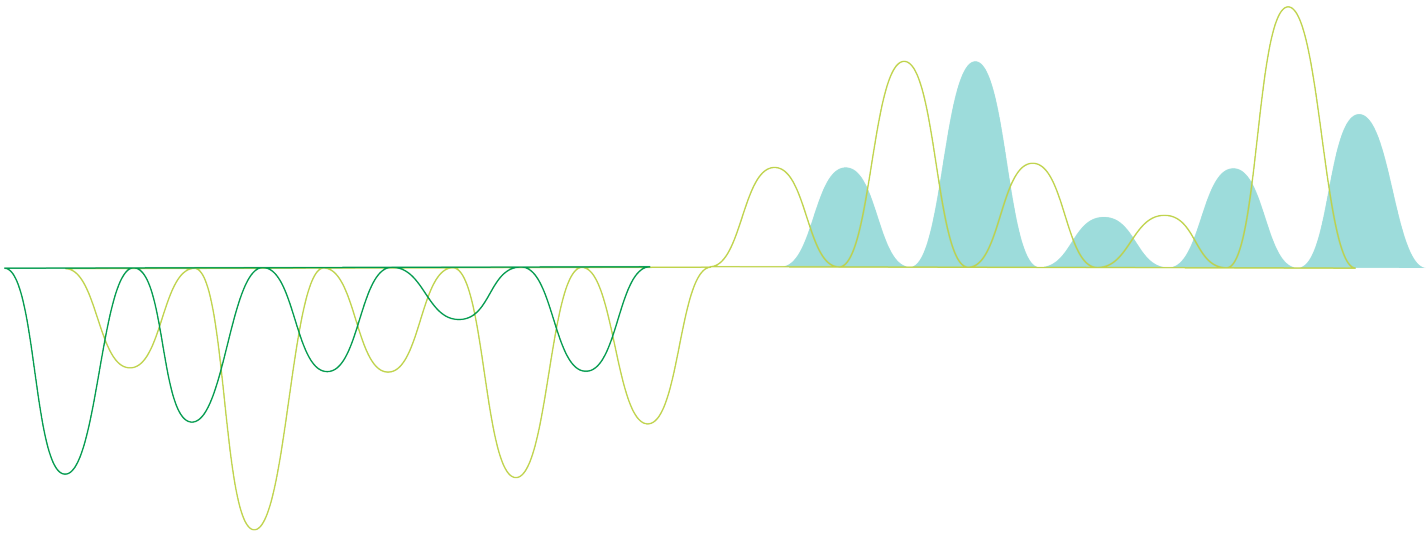
Sintassi dello script e funzioni grafiche

Qlik Sense®
February 2023



Contents

1 Che cos'è Qlik Sense?

Qlik Sense è una piattaforma per l'analisi dei dati. Con Qlik Sense è possibile analizzare i dati ed effettuare rilevazioni dati per proprio conto. È quindi possibile condividere conoscenze e analizzare i dati in gruppi e tra più organizzazioni. Qlik Sense consente di porre domande e trovare risposte oltre che seguire i propri percorsi personali per giungere alle proprie conclusioni. Qlik Sense consente agli utenti e ai loro colleghi di prendere decisioni in modo collaborativo.

1.1 Operazioni che è possibile eseguire in Qlik Sense

La maggior parte dei prodotti di BI (Business Intelligence) consente di rispondere a domande che possono essere formulate in anticipo. Ma come è possibile gestire le domande di follow-up, ossia quelle che possono sorgere dopo che un utente ha letto un report o ha esaminato una visualizzazione? Grazie all'esperienza associativa di Qlik Sense, è possibile rispondere a una domanda dopo l'altra completando i propri percorsi personali per giungere alle proprie conclusioni. Con Qlik Sense, è possibile esplorare liberamente i dati con pochi clic, apprendere in ogni fase del processo e individuare i passi successivi in base a quanto rilevato in precedenza.

1.2 Funzionamento di Qlik Sense

Qlik Sense genera in tempo reale viste di informazioni per l'utente. Qlik Sense non richiede report statici e predefiniti, né impone la dipendenza da altri utenti: è sufficiente un clic per apprendere. Ogni volta che si fa clic, Qlik Sense risponde immediatamente, aggiornando ogni visualizzazione e vista di Qlik Sense nell'app con serie di dati appena calcolati e visualizzazioni specifiche per le proprie selezioni.

Il modello app

Anziché distribuire e gestire applicazioni aziendali complesse, è possibile creare le proprie app Qlik Sense che sarà possibile riutilizzare, modificare e condividere con altri utenti. Il modello app consente di chiedere e rispondere da soli alla domanda successiva senza la necessità di ricorrere a un esperto per creare un nuovo report o una nuova visualizzazione.

L'esperienza associativa

Qlik Sense gestisce in modo automatico tutte le relazioni tra i dati e presenta le informazioni mediante una visione metaforica basata sui colori **green/white/gray**. Le selezioni vengono evidenziate in verde, i dati associati vengono visualizzati in bianco e i dati esclusi (non associati) vengono visualizzati in grigio. Questo feedback immediato consente di formulare nuove domande oltre che di continuare a esplorare e ad acquisire nuove conoscenze.

Collaborazione e mobilità

Qlik Sense consente inoltre di collaborare con i colleghi indipendentemente dal momento e dal luogo in cui si trovano. Tutte le funzionalità di Qlik Sense, compresa l'esperienza associativa e di collaborazione, sono disponibili sui dispositivi mobili. Con Qlik Sense, è possibile porre domande, quindi reperire risposte anche per le domande di follow-up, con i colleghi, indipendentemente dall'ubicazione.

1.3 Come è possibile distribuire Qlik Sense?

Qlik Sense può essere distribuito in due versioni, Qlik Sense Desktop e Qlik Sense Enterprise.

Qlik Sense Desktop

Questa è una versione per utenti singoli di facile installazione, che viene generalmente installata su un computer locale.

Qlik Sense Enterprise

Questa versione viene utilizzata per distribuire i siti Qlik Sense. Un sito è una raccolta di uno o più computer server connessi a un repository logico comune o a un nodo centrale.

1.4 Come amministrare e gestire un sito Qlik Sense

Con Qlik Management Console, è possibile configurare, gestire e monitorare i siti Qlik Sense in modo semplice e intuitivo. È possibile gestire licenze, regole di accesso e sicurezza, configurare nodi e connessioni di sorgenti dati oltre che sincronizzare il contenuto e gli utenti tra numerose attività e risorse.

1.5 Estensione di Qlik Sense e adattamento in base alle esigenze dell'utente

Qlik Sense fornisce API e SDK flessibili per consentire agli utenti di sviluppare le proprie estensioni, quindi di adattare e integrare Qlik Sense per diversi scopi tra cui:

Creazione di estensioni e mashup

È possibile effettuare lo sviluppo Web utilizzando JavaScript per creare estensioni che rappresentano una visualizzazione personalizzata nelle app Qlik Sense oppure utilizzare API di mashup per creare siti Web con il contenuto Qlik Sense.

Creazione di client

È possibile distribuire client in .NET e incorporare oggetti Qlik Sense nelle proprie applicazioni. È inoltre possibile creare client nativi in qualsiasi linguaggio di programmazione in grado di gestire la comunicazione WebSocket mediante l'utilizzo del protocollo client di Qlik Sense.

Creazione di strumenti server

Con le API del servizio e della directory utente, è possibile creare i propri strumenti personalizzati per amministrare e gestire i siti Qlik Sense.

Connessione ad altre sorgenti dati

È possibile creare connettori Qlik Sense per recuperare i dati da sorgenti dati personalizzate.

2 Panoramica sulla sintassi dello script

2.1 Introduzione alla sintassi dello script

In uno script vengono definiti il nome della sorgente dati, i nomi delle tabelle e i nomi dei campi inclusi nella logica. Vengono definiti inoltre i campi con diritti di accesso specificati nello script. Uno script è costituito da un certo numero di istruzioni che vengono eseguite consecutivamente.

La sintassi dello script e la sintassi della riga di comando di Qlik Sense vengono descritte in una notazione denominata metasintassi Backus-Naur Form o codice BNF.

Le prime righe di codice vengono generate automaticamente quando si crea un nuovo file Qlik Sense. I valori predefiniti di queste variabili di interpretazione numerica derivano dalle impostazioni internazionali del sistema operativo.

Lo script è costituito da numerose istruzioni dello script e parole chiave che vengono eseguite consecutivamente. Tutte le istruzioni dello script devono terminare con un punto e virgola, **;**.

È possibile utilizzare espressioni e funzioni nelle istruzioni **LOAD** per trasformare i dati caricati.

Per i file tabella contenenti virgole, tabulazioni o punti e virgola come delimitatori, è possibile utilizzare un'istruzione **LOAD**. Per impostazione predefinita, un'istruzione **LOAD** carica tutti i campi del file.

È possibile accedere ai database generici utilizzando i connettori del database ODBC o OLE DB. In questo caso, vengono usate le istruzioni SQL standard. La sintassi SQL accettata varia a seconda dei diversi driver ODBC.

Inoltre, è possibile accedere ad altre sorgenti dati utilizzando i connettori personalizzati.

2.2 Che cos'è la metasintassi Backus-Naur Form?

La sintassi dello script e la sintassi della riga di comando di Qlik Sense vengono descritte in una notazione denominata metasintassi Backus-Naur Form o codice BNF.

Nella tabella indicata di seguito viene fornito un elenco di simboli utilizzati nel codice BNF, con una descrizione di come vengono interpretati:

Simboli

Simbolo	Descrizione
|	Operatore OR logico: il simbolo può essere utilizzato da entrambi i lati.
( )	Parentesi che stabiliscono la precedenza: vengono utilizzate per strutturare la sintassi BNF.
[ ]	Parentesi quadre: gli elementi racchiusi sono opzionali.
{ }	Parentesi graffe: gli elementi racchiusi possono essere ripetuti zero o più volte.

Simbolo	Descrizione
Simbolo	Categoria sintattica nonterminale: può essere suddivisa ulteriormente in altri simboli. Ad esempio, composizione dei simboli precedenti, altri simboli nonterminali, stringhe di testo e così via.
::=	Contrassegna l'inizio di un blocco che definisce un simbolo.
LOAD	Simbolo terminale che consiste in una stringa di testo. Deve essere scritto nello script così come è visualizzato.

Tutti i simboli terminali sono stampati con carattere **bold face**. Ad esempio, "(" va interpretata come parentesi che stabilisce una precedenza, mentre "**(**" va interpretata come carattere che deve apparire nello script.

Esempio:

La descrizione dell'istruzione alias è:

alias fieldname **as** aliasname { , fieldname **as** aliasname}

Questo deve essere interpretato come la stringa testo "alias", seguita da un nome campo arbitrario, seguito dalla stringa di testo "as", seguita da un nome alias arbitrario. È possibile inserire qualsiasi numero di combinazioni aggiuntive di "fieldname as alias", separate da virgole.

Le seguenti istruzioni sono corrette:

```
alias a as first;
alias a as first, b as second;
alias a as first, b as second, c as third;
```

Le seguenti istruzioni non sono corrette:

```
alias a as first b as second;
alias a as first { , b as second };
```

2 Istruzioni e parole chiave dello script

Lo script di Qlik Sense è costituito da una serie di istruzioni. Un'istruzione può essere un'istruzione di script regolare o un'istruzione di controllo dello script. Alcune istruzioni possono essere precedute da prefissi.

Le istruzioni regolari vengono generalmente utilizzate per la manipolazione dei dati. Queste istruzioni possono essere scritte su un qualsiasi numero di righe nello script e devono sempre terminare con un punto e virgola, ";".

In genere, le istruzioni di controllo vengono utilizzate per controllare il flusso di esecuzione dello script. Ogni clausola di un'istruzione di controllo deve essere mantenuta in una singola riga dello script e può terminare con un punto e virgola oppure con un fine riga.

I prefissi possono essere applicati alle istruzioni regolari pertinenti, ma mai a istruzioni di controllo. I prefissi **when** e **unless** possono comunque essere utilizzati come suffissi per alcune specifiche clausole di istruzioni di controllo.

Nel seguente sottocapitolo, è riportato un elenco in ordine alfabetico di tutte le istruzioni di script, le istruzioni di controllo e di tutti i prefissi.

Tutte le parole chiave dello script possono essere immesse con qualsiasi combinazione di caratteri maiuscoli e minuscoli. I nomi dei campi e delle variabili utilizzati nelle istruzioni possono essere immessi indipendentemente dal formato del carattere.

2.3 Istruzioni di controllo dello script

Lo script di Qlik Sense è costituito da una serie di istruzioni. Un'istruzione può essere un'istruzione di script regolare o un'istruzione di controllo dello script.

In genere, le istruzioni di controllo vengono utilizzate per controllare il flusso di esecuzione dello script. Ogni clausola di un'istruzione di controllo deve essere inserita in una singola riga nello script e può terminare con un punto e virgola o con un carattere di fine riga.

I prefissi non vengono mai utilizzati nelle istruzioni di controllo, con l'eccezione dei prefissi **when** e **unless**, che possono essere utilizzati con alcune istruzioni di controllo specifiche.

Tutte le parole chiave dello script possono essere immesse con qualsiasi combinazione di caratteri maiuscoli e minuscoli.

Prospetto delle istruzioni di controllo dello script

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Call

L'istruzione di controllo **call** consente di chiamare una subroutine che deve essere definita da un'istruzione **sub** precedente.

```
Call name ( [ paramlist ])
```

Do..loop

L'istruzione di controllo **do..loop** è un costrutto per la ripetizione di script che esegue una o più istruzioni finché non incontra una condizione logica.

```
Do..loop [ ( while | until ) condition ] [statements]
[exit do [ ( when | unless ) condition ] [statements]
loop [ ( while | until ) condition ]
```

Exit script

Questa istruzione di controllo interrompe l'esecuzione dello script. Può essere inserita in un punto qualsiasi dello script.

```
Exit script[ (when | unless) condition ]
```

For each ..next

L'istruzione di controllo **for each..next** è un costrutto per la ripetizione di script che esegue una o più istruzioni per ogni valore in un elenco le cui voci sono separate da virgole. Le istruzioni incluse nel ciclo fra **for** e **next** verranno eseguite per ogni valore nell'elenco.

```
For each..next var in list
[statements]
[exit for [ ( when | unless ) condition ]
[statements]
next [var]
```

For..next

L'istruzione di controllo **for..next** è un costrutto per la ripetizione di script con un contatore. Le istruzioni all'interno del ciclo incluso tra **for** e **next** verranno eseguite per ogni valore del contatore in base ai limiti inferiore e superiore specificati.

```
For..next counter = expr1 to expr2 [ stepexpr3 ]
[statements]
[exit for [ ( when | unless ) condition ]
[statements]
Next [counter]
```

If..then

L'istruzione di controllo **if..then** è un costrutto per la selezione di script che forza l'esecuzione dello script su percorsi diversi in base a una o più condizioni logiche.

> *Poiché **if..then** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, ciascuna delle quattro possibili clausole corrispondenti (**if..then**, **elseif..then**, **else** e **end if**) deve essere contenuta in una sola riga.*

```
If..then..elseif..else..end if condition then
  [ statements ]
{ elseif condition then
  [ statements ] }
[ else
```

```
  [ statements ] ]
end if
```

Sub

L'istruzione di controllo **sub..end sub** definisce una subroutine che può essere richiamata da un'istruzione **call**.

```
Sub..end sub name [ ( paramlist )] statements end sub
```

Switch

L'istruzione di controllo **switch** è un costrutto per la selezione di script che forza l'esecuzione dello script su percorsi diversi, in base al valore di un'espressione.

```
Switch..case..default..end switch expression {case valuelist [ statements ]}
[default statements] end switch
```

Call

L'istruzione di controllo **call** consente di chiamare una subroutine che deve essere definita da un'istruzione **sub** precedente.

Sintassi:

```
Call name ( [ paramlist ])
```

Argomenti:

Argomenti

Argomento	Descrizione
name	Il nome della subroutine.
paramlist	Un elenco separato da virgole di parametri effettivi da inviare alla subroutine. Ogni voce dell'elenco può essere un nome di campo, una variabile o un'espressione arbitraria.

La subroutine chiamata da un'istruzione **call** deve essere definita da un'istruzione **sub** rilevata precedentemente durante l'esecuzione dello script.

I parametri vengono copiati nella subroutine e, se il parametro nell'istruzione **call** è una variabile e non un'espressione, verranno copiati nuovamente all'uscita dalla subroutine.

Limiti:

- Poiché **call** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, non deve superare un limite di riga.
- Quando si definisce una routine secondaria con `Sub..end sub` all'interno di un'istruzione di controllo, ad esempio `if..then`, è possibile richiamare la routine secondaria solo dall'interno della stessa istruzione di controllo.

Esempio:

In questo esempio sono mostrati tutti i file correlati a Qlik all'interno di una cartella e delle relative sottocartelle e le informazioni dei file vengono memorizzate in una tabella. Si presuppone che sia stata creata una connessione dati alla cartella denominata Apps.

La subroutine DoDir viene chiamata con il riferimento alla cartella, 'lib://Apps', come parametro. All'interno della subroutine, è presente una chiamata ricorrente `Call DoDir (Dir)`, che indica alla funzione di ricercare in modo ricorrente i file nelle sottocartelle.

```
sub DoDir (Root)    For Each Ext in 'qvw', 'qvo', 'qvs', 'qvt', 'qvd', 'qvc', 'qvf'       For
Each File in filelist (Root&'\*.' &Ext)           LOAD              '$(File)' as Name,
      FileSize( '$(File)' ) as Size,               FileTime( '$(File)' ) as FileTime
autogenerate 1;       Next File    Next Ext       For Each Dir in dirlist (Root&'\*' )
Call DoDir (Dir)    Next Dir End Sub  Call DoDir ('lib://Apps')
```

Do..loop

L'istruzione di controllo **do..loop** è un costrutto per la ripetizione di script che esegue una o più istruzioni finché non incontra una condizione logica.

Sintassi:

```
Do [ ( while | until ) condition ] [statements]
[exit do [ ( when | unless ) condition ] [statements]
loop[ ( while | until ) condition ]
```

> *Poiché **do..loop** è un'istruzione di controllo che termina con un punto e virgola o con un carattere di fine riga, ciascuna delle tre possibili clausole corrispondenti (**do**, **exit do** e **loop**) deve essere contenuta in una sola riga.*

Argomenti:

Argomenti

Argomento	Descrizione
condition	Un'espressione logica che restituisce un valore True o False.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.
while / until	La clausola condizionale **while** o **until** deve comparire una sola volta in ciascuna istruzione **do..loop**, dopo **do** o dopo **loop**. Ogni espressione condition verrà interpretata solo al primo rilevamento, ma verrà valutata ogni volta che sarà rilevata nel ciclo.
exit do	Se all'interno del ciclo è presente una clausola **exit do**, l'esecuzione dello script verrà trasferita alla prima istruzione dopo la clausola **loop** indicando quindi la fine del ciclo. Una clausola **exit do** può essere resa condizionale dall'utilizzo opzionale di un suffisso **when** o **unless**.

Esempio:

```
// LOAD files file1.csv..file9.csv
Set a=1;
Do while a<10
LOAD * from file$(a).csv;
Let a=a+1;
Loop
```

End

La parola chiave dello script **End** viene utilizzata per chiudere le clausole **If**, **Sub** e **Switch**.

Exit

La parola chiave dello script **Exit** fa parte dell'istruzione **Exit Script**, ma può essere utilizzata anche per uscire dalle clausole **Do**, **For** o **Sub**.

Exit script

Questa istruzione di controllo interrompe l'esecuzione dello script. Può essere inserita in un punto qualsiasi dello script.

Sintassi:

```
Exit Script [ (when | unless) condition ]
```

Poiché **exit script** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, non deve superare un limite di riga.

Argomenti:

Argomenti

Argomento	Descrizione
condition	Un'espressione logica che restituisce un valore True o False.
when / unless	Un'istruzione **exit script** può essere resa condizionale dall'utilizzo opzionale della clausola **when** o **unless**.

Esempi:

```
//Exit script
Exit Script;

//Exit script when a condition is fulfilled
Exit Script when a=1
```

For..next

L'istruzione di controllo **for..next** è un costrutto per la ripetizione di script con un contatore. Le istruzioni all'interno del ciclo incluso tra **for** e **next** verranno eseguite per ogni valore del contatore in base ai limiti inferiore e superiore specificati.

Sintassi:

```
For counter = expr1 to expr2 [ step expr3 ]
[statements]
[exit for [ ( when | unless ) condition ]
[statements]
Next [counter]
```

Le espressioni *expr1*, *expr2* ed *expr3* vengono valutate solo la prima volta che il ciclo viene eseguito. Il valore della variabile counter può essere modificato dalle istruzioni all'interno del ciclo, tuttavia l'utilizzo di questa procedura di programmazione non è consigliato.

Se all'interno del ciclo è presente una clausola **exit for**, l'esecuzione dello script verrà trasferita alla prima istruzione dopo la clausola **next** indicando quindi la fine del ciclo. Una clausola **exit for** può essere resa condizionale dall'utilizzo opzionale di un suffisso **when** o **unless**.

*Poiché **for..next** è un'istruzione di controllo che termina con un punto e virgola o con un carattere di fine riga, ciascuna delle tre possibili clausole corrispondenti (**for..to..step**, **exit for** e **next**) deve essere contenuta in una sola riga.*

Argomenti:

Argomenti

Argomento	Descrizione
counter	Un nome di variabile. Se *counter* viene specificato dopo **next**, deve avere lo stesso nome di variabile rilevato dopo l'istruzione **for** corrispondente.
expr1	Un'espressione che determina il primo valore della variabile*counter* per cui deve essere eseguito il ciclo.
expr2	Un'espressione che determina l'ultimo valore della variabile *counter* per cui deve essere eseguito il ciclo.
expr3	Un'espressione che determina il valore che indica l'incremento della variabile *counter* ogni volta che il ciclo è stato eseguito.
condition	Un'espressione logica che restituisce un valore True o False.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.

Example 1: Caricamento di una sequenza di file

```
// LOAD files file1.csv..file9.csv
for a=1 to 9

      LOAD * from file$(a).csv;

next
```

Example 2: Caricamento di un numero casuale di file

In questo esempio si presuppone l'utilizzo dei file di dati *x1.csv, x3.csv, x5.csv, x7.csv* e *x9.csv*. Il caricamento viene interrotto in un punto casuale mediante la condizione `if rand( )<0.5 then`.

```
for counter=1 to 9 step 2

      set filename=x$(counter).csv;
      if rand( )<0.5 then

            exit for unless counter=1

      end if
      LOAD a,b from $(filename);

next
```

For each..next

L'istruzione di controllo **for each..next** è un costrutto per la ripetizione di script che esegue una o più istruzioni per ogni valore in un elenco le cui voci sono separate da virgole. Le istruzioni incluse nel ciclo fra **for** e **next** verranno eseguite per ogni valore nell'elenco.

Sintassi:

Una sintassi speciale consente di generare elenchi contenenti nomi di file e di directory nella directory attuale.

```
for each var in list
[statements]
[exit for [ ( when | unless ) condition ]
[statements]
next [var]
```

Argomenti:

Argomenti

Argomento	Descrizione
var	Il nome di una variabile di script che acquisisce un nuovo valore dall'elenco a ogni esecuzione del ciclo. Se **var** viene specificato dopo **next**, deve avere lo stesso nome di variabile rilevato dopo l'istruzione **for each** corrispondente.

Il valore della variabile **var** può essere modificato dalle istruzioni all'interno del ciclo, tuttavia l'utilizzo di questa procedura di programmazione non è consigliato.

Se all'interno del ciclo è presente una clausola **exit for**, l'esecuzione dello script verrà trasferita alla prima istruzione dopo la clausola **next** indicando quindi la fine del ciclo. Una clausola **exit for** può essere resa condizionale dall'utilizzo opzionale di un suffisso **when** o **unless**.

> *Poiché **for each..next** è un'istruzione di controllo che termina con un punto e virgola o con un carattere di fine riga, ciascuna delle tre possibili clausole corrispondenti (**for each**, **exit for** e **next**) deve essere contenuta in una sola riga.*

Sintassi:

```
list := item { , item }
item := constant | (expression) | filelist mask | dirlist mask |
fieldvaluelist mask
```

Argomenti

Argomento	Descrizione
constant	Qualsiasi numero o stringa. Tenere presente che una stringa inserita direttamente nello script deve essere racchiusa tra virgolette singole. Se la stringa non viene racchiusa tra virgolette singole, verrà interpretata come una variabile, pertanto verrà utilizzato il valore della variabile. Non è necessario che i numeri siano racchiusi tra virgolette singole.
expression	Un'espressione arbitraria.
mask	Una maschera di un nome di file o di cartella che può includere un carattere qualsiasi di nome di file valido, così come i caratteri speciali standard, quali ***** e **?**. È possibile utilizzare percorsi di file assoluti o percorsi lib://.
condition	Un'espressione logica che restituisce un valore True o False.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.
filelist mask	Questa sintassi restituisce un elenco con valori separati da virgole di tutti i file presenti nella directory attuale che presentano una corrispondenza con la maschera del nome di file. *Questo argomento supporta esclusivamente le connessioni alla libreria in modalità standard.*

Argomento	Descrizione
dirlist mask	Questa sintassi restituisce un elenco con valori separati da virgole di tutte le cartelle incluse nella cartella attuale che presentano una corrispondenza con la maschera del nome di file. *Questo argomento supporta esclusivamente le connessioni alla libreria in modalità standard.*
fieldvaluelist mask	Questa sintassi ripete i valori di un campo già caricato in Qlik Sense.

*Qlik Connettori provider di archiviazione Web e altre connessioni DataFiles non supportano le maschere di filtro che utilizzano caratteri speciali (***** e **?**).*

Example 1: Caricamento di un elenco di file

```
// LOAD the files 1.csv, 3.csv, 7.csv and xyz.csv for each a in 1,3,7,'xyz'    LOAD * from
file$(a).csv; next
```

Example 2: Creazione di un elenco di file sul disco

In questo esempio viene caricato un elenco di tutti i file correlati a Qlik Sense in una cartella.

```
 sub DoDir (Root)    for each Ext in 'qvw', 'qva', 'qvo', 'qvs', 'qvc', 'qvf', 'qvd'
for each File in filelist (Root&'/*.' &Ext)           LOAD              '$(File)' as Name,
         FileSize( '$(File)' ) as Size,              FileTime( '$(File)' ) as FileTime
 autogenerate 1;        next File     next Ext    for each Dir in dirlist (Root&'/*' )
call DoDir (Dir)     next Dir  end sub  call DoDir ('lib://DataFiles')
```

Example 3: Ripetizione dei valori di un campo

In questo esempio viene ripetuto l'elenco di valori caricati di FIELD e viene generato un nuovo campo NEWFIELD. Per ciascun valore di FIELD, verranno creati due record NEWFIELD.

```
 load * inline [ FIELD one two three ];  FOR Each a in FieldValueList('FIELD') LOAD '$(a)' &'-
'&RecNo() as NEWFIELD AutoGenerate 2; NEXT a
```

La tabella risultante avrà l'aspetto seguente:

Example table

NEWFIELD
one-1
one-2
two-1

NEWFIELD
two-2
three-1
three-2

If..then..elseif..else..end if

L'istruzione di controllo **if..then** è un costrutto per la selezione di script che forza l'esecuzione dello script su percorsi diversi in base a una o più condizioni logiche.

In genere, le istruzioni di controllo vengono utilizzate per controllare il flusso di esecuzione dello script. In un'espressione del grafico, utilizzare invece la funzione condizionale **if**.

Sintassi:

```
If  condition then
  [ statements ]
{ elseif condition then
  [ statements ] }
[ else
  [ statements ] ]
end if
```

Poiché **if..then** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, ciascuna delle quattro possibili clausole corrispondenti (**if..then**, **elseif..then**, **else** e **end if**) deve essere contenuta in una sola riga.

Argomenti:

Argomenti

Argomento	Descrizione
condition	Un'espressione logica che può restituire un valore True o False.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.

Example 1:

```
if a=1 then

      LOAD * from abc.csv;
      SQL SELECT e, f, g from tab1;

end if
```

Example 2:

```
if a=1 then; drop table xyz; end if;
```

Example 3:

```
if x>0 then
      LOAD * from pos.csv;
elseif x<0 then
      LOAD * from neg.csv;
else
      LOAD * from zero.txt;
end if
```

Next

La parola chiave dello script **Next** consente di chiudere i loop **For**.

Sub..end sub

L'istruzione di controllo **sub..end sub** definisce una subroutine che può essere richiamata da un'istruzione **call**.

Sintassi:

```
Sub name [ ( paramlist )] statements end sub
```

Gli argomenti vengono copiati nella subroutine e, se i relativi parametri reali nell'istruzione **call** corrispondono a un nome di variabile, vengono copiati nuovamente quando si chiude la subroutine.

Se una subroutine presenta più parametri formali di quelli effettivi passati da un'istruzione **call**, i parametri extra vengono inizializzati su NULL e possono essere utilizzati come variabili locali all'interno della subroutine.

Argomenti:

Argomenti

Argomento	Descrizione
name	Il nome della subroutine.
paramlist	Un elenco separato da virgole di nomi di variabili per i parametri formali della subroutine. Può essere utilizzato come qualsiasi variabile all'interno della subroutine.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.

Limiti:

- Poiché **sub** è un'istruzione di controllo che termina con un punto e virgola o con un carattere di fine riga, ciascuna delle due clausole corrispondenti (**sub** e **end sub**) deve essere contenuta in una sola riga.
- Quando si definisce una routine secondaria con `Sub..end sub` all'interno di un'istruzione di controllo, ad esempio `if..then`, è possibile richiamare la routine secondaria solo dall'interno della stessa istruzione di controllo.

Example 1:

```
Sub INCR (I,J)
I = I + 1
Exit Sub when I < 10
J = J + 1
End Sub
Call INCR (X,Y)
```

Example 2: - trasferimento parametri

```
Sub ParTrans (A,B,C)
A=A+1
B=B+1
C=C+1
End Sub
A=1
X=1
C=1
Call ParTrans (A, (X+1)*2)
```

Dall'esempio precedente risulta che localmente, all'interno della subroutine, A verrà inizializzato su 1, B verrà inizializzato su 4 e C verrà inizializzato su NULL.

Quando si chiude la subroutine, la variabile globale A otterrà 2 come valore (ricopiato dalla subroutine). Il secondo parametro reale “(X+1)*2” non verrà ricopiato dato che non si tratta di una variabile. Infine, la variabile globale C non verrà influenzata dalla chiamata della subroutine.

Switch..case..default..end switch

L'istruzione di controllo **switch** è un costrutto per la selezione di script che forza l'esecuzione dello script su percorsi diversi, in base al valore di un'espressione.

Sintassi:

```
Switch expression {case valuelist [ statements ]} [default statements] end switch
```

*Poiché **switch** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, ciascuna delle quattro possibili clausole corrispondenti (**switch**, **case**, **default** e **end switch**) deve essere contenuta in una sola riga.*

Argomenti:

Argomenti

Argomento	Descrizione
expression	Un'espressione arbitraria.
valuelist	Un elenco separato da virgole dei valori con i quali viene confrontato il valore dell'espressione. L'esecuzione dello script continua con le istruzioni del primo gruppo in cui il valore di valuelist è pari al valore nell'espressione. Ciascun valore in valuelist può essere un'espressione arbitraria. Se non viene individuata alcuna corrispondenza in nessuna clausola **case**, vengono eseguite le eventuali istruzioni della clausola **default**.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.

Esempio:

```
Switch I
Case 1
LOAD '$(I): CASE 1' as case autogenerate 1;
Case 2
LOAD '$(I): CASE 2' as case autogenerate 1;
Default
LOAD '$(I): DEFAULT' as case autogenerate 1;
End Switch
```

To

La parola chiave dello script **To** viene utilizzata in diverse istruzioni dello script.

2.4 Prefissi dello script

I prefissi possono essere applicati alle istruzioni regolari pertinenti, ma mai a istruzioni di controllo. I prefissi **when** e **unless** possono comunque essere utilizzati come suffissi per alcune specifiche clausole di istruzioni di controllo.

Tutte le parole chiave dello script possono essere immesse con qualsiasi combinazione di caratteri maiuscoli e minuscoli. I nomi dei campi e delle variabili utilizzati nelle istruzioni possono essere immessi indipendentemente dal formato del carattere.

Prospetto dei prefissi dello script

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Add

Il prefisso **Add** può essere aggiunto a qualsiasi istruzione **LOAD** o **SELECT** nello script per specificare che dovrebbe aggiungere record a un'altra tabella. Specifica anche che questa istruzione dovrebbe essere eseguita in un ricaricamento parziale. Il prefisso **Add** può essere usato anche in un'istruzione **Map**.

```
Add [only] [Concatenate[(tablename )]] (loadstatement | selectstatement)
Add [ Only ] mapstatement
```

Buffer

È possibile creare e gestire automaticamente i file QVD mediante il prefisso **buffer**. Questo prefisso può essere utilizzato in quasi tutte le istruzioni **LOAD** e **SELECT** di uno script e indica che i file QVD vengono utilizzati per memorizzare nella cache/nel buffer il risultato dell'istruzione.

```
Buffer[(option [ , option])] ( loadstatement | selectstatement )
option::= incremental | stale [after] amount [(days | hours)]
```

Concatenate

Se due tabelle da concatenare contengono gruppi differenti di campi, è tuttavia possibile imporre la concatenazione di due tabelle utilizzando il prefisso **Concatenate**.

```
Concatenate[ (tablename ) ] ( loadstatement | selectstatement )
```

Crosstable

Il prefisso di caricamento **crosstable** viene utilizzato per trasporre i dati strutturati in “tabella incrociata” o in "tabella pivot". I dati strutturati in questo modo si trovano comunemente quando si lavora con le sorgenti dei fogli di calcolo. L'output e lo scopo del prefisso di caricamento **crosstable** è di trasporre tali strutture nell'equivalente di una normale tabella in un formato a colonne, poiché questa struttura è generalmente più adatta per l'analisi in Qlik Sense.

```
Crosstable (attribute field name, data field name [ , n ] ) ( loadstatement |
selectstatement )
```

First

Il prefisso **First** aggiunto a un'istruzione **LOAD** o **SELECT (SQL)** viene utilizzato per caricare un numero di record massimo impostato dalla tabella di origine dei dati.

```
First n( loadstatement | selectstatement )
```

Generic

Il prefisso di caricamento **Generic** consente la conversione dei dati modellati su entità-attributo-valore (EAV) in una struttura di tabella relazionale tradizionale e normalizzata. La modellazione EAV in alternativa è denominata "modellazione di dati generici" o "schema aperto".

```
Generic ( loadstatement | selectstatement )
```

Hierarchy

Il prefisso **hierarchy** viene utilizzato per trasformare una tabella gerarchica padre-figlio in una tabella utile in un modello dati Qlik Sense. Può essere inserito prima di un'istruzione **LOAD** o **SELECT** e utilizzerà i risultati dell'istruzione di caricamento come input per la trasformazione della tabella.

```
Hierarchy (NodeID, ParentID, NodeName, [ParentName], [PathSource],
[PathName], [PathDelimiter], [Depth])(loadstatement | selectstatement)
```

HierarchBelongsTo

Questo prefisso viene utilizzato per trasformare una tabella gerarchica padre-figlio in una tabella utile in un modello dati Qlik Sense. Può essere inserito prima di un'istruzione **LOAD** o **SELECT** e utilizzerà i risultati dell'istruzione di caricamento come input per la trasformazione della tabella.

```
HierarchyBelongsTo (NodeID, ParentID, NodeName, AncestorID, AncestorName,
[DepthDiff])(loadstatement | selectstatement)
```

Inner

I prefissi **join** e **keep** possono essere preceduti dal prefisso **inner**.

Se viene inserito prima di **join**, specifica che occorre utilizzare un'unione interna. La tabella risultante contiene solo le combinazioni di valori di campo estratte dalle tabelle di dati non elaborati dove i valori di campo di collegamento vengono rappresentati in entrambe le tabelle. Se utilizzato prima di **keep**, specifica che entrambe le tabelle di dati non elaborati devono essere ridotte alla loro intersezione comune prima di essere memorizzate in Qlik Sense.

.

```
Inner ( Join | Keep) [ (tablename) ](loadstatement |selectstatement )
```

IntervalMatch

Il prefisso **IntervalMatch** consente di creare una tabella che corrisponde sia ai valori numerici discreti su uno o più intervalli numerici che, in modo opzionale, ai valori di una o più chiavi aggiuntive.

```
IntervalMatch (matchfield)(loadstatement | selectstatement )
IntervalMatch (matchfield,keyfield1 [ , keyfield2, ... keyfield5 ] )
(loadstatement | selectstatement )
```

Join

Il prefisso **join** unisce la tabella caricata a una tabella denominata esistente oppure all'ultima tabella di dati creata in precedenza.

```
[Inner | Outer | Left | Right ] Join [ (tablename ) ]( loadstatement |
selectstatement )
```

Keep

Il prefisso **keep** è simile al prefisso **join**. Analogamente al prefisso **join**, confronta la tabella caricata con una tabella denominata esistente o con l'ultima tabella dati creata in precedenza, tuttavia, invece di unire la tabella caricata alla tabella esistente, riduce una o entrambe le due tabelle prima che vengano memorizzate in Qlik Sense, in base all'intersezione dei dati della tabella. Il confronto effettuato equivale a un'unione naturale effettuata su tutti i campi comuni, ad esempio nello stesso modo di un'unione corrispondente. In ogni modo, le due tabelle non vengono unite e verranno conservate in Qlik Sense come due tabelle denominate separatamente.

```
(Inner | Left | Right) Keep [(tablename ) ]( loadstatement | selectstatement
)
```

Left

I prefissi **Join** e **Keep** possono essere preceduti dal prefisso **left**.

Se viene inserito prima di **join**, specifica che occorre utilizzare un'unione sinistra. La tabella risultante conterrà solo le combinazioni di valori di campo estratte dalle tabelle di dati non elaborati, dove i valori di campo di collegamento vengono rappresentati nella prima tabella. Se viene utilizzato prima di **keep**, specifica che la seconda tabella di dati non elaborati deve essere ridotta alla sua intersezione comune con la prima tabella prima di essere memorizzata in Qlik Sense.

```
Left ( Join | Keep) [ (tablename) ](loadstatement |selectstatement )
```

Mapping

Il prefisso **mapping** consente di creare una tabella di mapping che può essere utilizzata, ad esempio, per sostituire i valori di campo e i nomi di campo durante l'esecuzione dello script.

```
Mapping  ( loadstatement | selectstatement )
```

Merge

Il prefisso **Merge** può essere aggiunto a qualsiasi istruzione **LOAD** o **SELECT** nello script per specificare che dovrebbe aggiungere record a un'altra tabella. Specifica anche che questa istruzione dovrebbe essere eseguita in un ricaricamento parziale.

```
Unisci [only] [(SequenceNoField [, SequenceNoVar])] On ListOfKeys
[Concatenate [(TableName)]] (loadstatement | selectstatement)
```

NoConcatenate

Con il prefisso **NoConcatenate**, due tabelle che vengono caricate con gruppi di campo identici verranno considerate come due tabelle interne separate, invece di venire concatenate automaticamente.

```
NoConcatenate( loadstatement | selectstatement )
```

Outer

Il prefisso **Join** esplicito può essere preceduto dal prefisso **Outer** per specificare un'unione esterna. In un'unione esterna vengono generate tutte le combinazioni tra le due tabelle. La tabella risultante conterrà quindi le combinazioni di valori di campo provenienti dalle tabelle di dati non elaborati dove i valori di campo di collegamento vengono rappresentati in una o entrambe le tabelle. La parola chiave **Outer** è facoltativa ed è il tipo di unione predefinito utilizzato quando non viene specificato un prefisso di unione.

```
Outer Join [ (tablename) ](loadstatement |selectstatement )
```

Partial reload

Un ricaricamento completo inizia sempre eliminando tutte le tabelle nel modello dati esistente, dopodiché esegue lo script di caricamento.

Un *Caricamento parziale (page 99)* non effettuerà tale operazione. Mantiene invece tutte le tabelle nel modello dati ed esegue solo le istruzioni **Load** e **Select** precedute da un prefisso **Aggiungi**, **Unisci** o **Sostituisci**. Altre tabelle di dati non sono interessate dal comando. L'argomento **solo** indica che l'istruzione deve essere eseguita solo durante i caricamenti parziali e deve essere ignorata durante i caricamenti completi. La tabella seguente riepiloga l'esecuzione dell'istruzione per i ricaricamenti parziali e completi.

Replace

Il prefisso **Replace** può essere aggiunto a qualsiasi istruzione **LOAD** o **SELECT** nello script per specificare che la tabella caricata dovrebbe sostituire un'altra tabella. Specifica anche che questa istruzione dovrebbe essere eseguita in un ricaricamento parziale. Il prefisso **Replace** può essere usato anche in un'istruzione **Map**.

```
Replace [only] [Concatenate[(tablename) ]] (loadstatement | selectstatement)
Replace [only] mapstatement
```

Right

I prefissi **Join** e **Keep** possono essere preceduti dal prefisso **right**.

Se viene inserito prima di **join**, specifica che occorre utilizzare un'unione destra. La tabella risultante contiene solo le combinazioni di valori di campo estratte dalle tabelle di dati non elaborati, dove i valori di campo di collegamento vengono rappresentati nella seconda tabella. Se viene utilizzato prima di **keep**, specifica che la prima tabella di dati non elaborati deve essere ridotta alla sua intersezione comune con la seconda tabella prima di essere memorizzata in Qlik Sense.

```
Right (Join | Keep) [(tablename)](loadstatement |selectstatement )
```

Sample

Il prefisso **sample** aggiunto a un'istruzione **LOAD** o **SELECT** viene utilizzato per caricare un campione casuale di record dalla sorgente dati.

```
Sample p ( loadstatement | selectstatement )
```

Semantic

È possibile caricare le tabelle che contengono relazioni fra i record utilizzando il prefisso **semantic**. Si può trattare ad esempio di auto-riferimenti all'interno di una tabella, in cui un record punta a un altro, come ad esempio padre, appartenenza o predecessore.

```
Semantic ( loadstatement | selectstatement)
```

Unless

Il prefisso e suffisso **unless** viene utilizzato per creare una clausola condizionale che determina se valutare o meno un'istruzione oppure una clausola exit. Può essere considerato come un'alternativa compatta all'istruzione completa **if..end if**.

```
(Unless condition statement | exitstatement Unless condition )
```

When

Il prefisso e suffisso **when** viene utilizzato per creare una clausola condizionale che determina se eseguire o meno un'istruzione oppure una clausola exit. Può essere considerato come un'alternativa compatta all'istruzione completa **if..end if**.

```
( When condition statement | exitstatement when condition )
```

Add

Il prefisso **Add** può essere aggiunto a qualsiasi istruzione **LOAD** o **SELECT** nello script per specificare che dovrebbe aggiungere record a un'altra tabella. Specifica anche che questa istruzione dovrebbe essere eseguita in un ricaricamento parziale. Il prefisso **Add** può essere usato anche in un'istruzione **Map**.

Affinché il ricaricamento parziale funzioni correttamente, aprire l'app con i dati prima di attivare un ricaricamento parziale.

Eseguire un ricaricamento parziale usando il pulsante **Ricarica**. È anche possibile utilizzare Qlik Engine JSON API.

Sintassi:

```
Add [only] [Concatenate[(tablename)]] (loadstatement | selectstatement)
```

```
Add [only] mapstatement
```

Durante un caricamento normale (non parziale), la costruzione **Add LOAD** funzionerà come normale istruzione **LOAD**. I record verranno generati e archiviati in una tabella.

Se viene utilizzato il prefisso **Concatenate**, o se esiste una tabella con lo stesso set di campi, i record verranno aggiunti alla tabella esistente rilevante. Altrimenti, la costruzione **Add LOAD** creerà una nuova tabella.

Un caricamento parziale otterrà lo stesso risultato. L'unica differenza è che la costruzione **Add LOAD** non creerà mai una nuova tabella. Esiste sempre una tabella pertinente dall'esecuzione script precedente a cui aggiungere i record.

Non viene eseguito alcun controllo di duplicati. Pertanto, un'istruzione che utilizza il prefisso **Add** spesso includerà un qualificatore distinto o una clausola dove a protezione dei duplicati.

L'istruzione **Add Map...Using** determina l'esecuzione del mapping anche durante l'esecuzione parziale dello script.

Argomenti:

Argomenti

Argomento	Descrizione
only	Un qualificatore opzionale che denota che l'istruzione dovrebbe essere eseguita solo durante i caricamenti parziali. Deve essere ignorata durante i caricamenti normali (non parziali).

Esempi e risultati:

Esempio	Risultato

`Tab1:`
 	
`LOAD Name, Number FROM Persons.csv;`
 	
`Add LOAD Name, Number FROM newPersons.csv;`
	Durante il ricaricamento normale, i dati vengono caricati dal file *Persons.csv* e memorizzati nella tabella di Qlik Sense Tab1. I dati in *NewPersons.csv* vengono quindi concatenati alla stessa tabella Qlik Sense.
	
	Durante il ricaricamento parziale, i dati vengono caricati dal file *NewPersons.csv* e aggiunti alla tabella di Qlik Sense Tab1. Non viene eseguito alcun controllo di duplicati.

`Tab1:`
 	
`SQL SELECT Name, Number FROM Persons.csv;`
 	
`Add LOAD Name, Number FROM NewPersons.csv where not exists(Name);`
	Viene eseguito un controllo di duplicati mediante la verifica dell'esistenza di Name nei dati della tabella caricati in precedenza.
	
	Durante il ricaricamento normale, i dati vengono caricati dal file *Persons.csv* e memorizzati nella tabella di Qlik Sense Tab1. I dati in *NewPersons.csv* vengono quindi concatenati alla stessa tabella Qlik Sense.
	
	Durante il ricaricamento parziale, i dati vengono caricati dal file *NewPersons.csv*, che viene aggiunto alla tabella Qlik Sense Tab1. Viene eseguito un controllo di duplicati che verifica l'esistenza di Name nei dati della tabella precedentemente caricata.

`Tab1:`
 	
`LOAD Name, Number FROM Persons.csv;`
 	
`Add Only LOAD Name, Number FROM NewPersons.csv where not exists(Name);`
	Durante il ricaricamento normale, i dati vengono caricati dal file *Persons.csv* e memorizzati nella tabella di Qlik Sense Tab1. L'istruzione che carica *NewPersons.csv* viene ignorata.
	
	Durante il ricaricamento parziale, i dati vengono caricati dal file *NewPersons.csv*, che viene aggiunto alla tabella Qlik Sense Tab1. Viene eseguito un controllo di duplicati che verifica l'esistenza di Name nei dati della tabella precedentemente caricata.

Buffer

È possibile creare e gestire automaticamente i file QVD mediante il prefisso **buffer**. Questo prefisso può essere utilizzato in quasi tutte le istruzioni **LOAD** e **SELECT** di uno script e indica che i file QVD vengono utilizzati per memorizzare nella cache/nel buffer il risultato dell'istruzione.

Sintassi:

```
Buffer [(option [ , option])] ( loadstatement | selectstatement )
option::= incremental | stale [after] amount [(days | hours)]
```

Se nessuna opzione viene utilizzata, il buffer QVD creato alla prima esecuzione dello script verrà utilizzato indefinitamente.

Il file del buffer viene salvato nella sottocartella *Buffers* solitamente in *C:\ProgramData\Qlik\Sense\Engine\Buffers* (installazione sul server) o *C:\Utenti\{user}\Documenti\Qlik\Sense\Buffers* (Qlik Sense Desktop).

Il nome del file QVD è un nome calcolato, un hash esadecimale a 160 bit di tutta l'istruzione **LOAD** o **SELECT** successiva e delle altre informazioni discriminanti. Questo significa che il buffer QVD verrà invalidato da qualsiasi modifica apportata all'istruzione **LOAD** o **SELECT** seguente.

I buffer QVD vengono normalmente rimossi quando non esistono più riferimenti a essi durante l'intera esecuzione di uno script nell'app che li ha creati oppure quando l'app che li ha creati non esiste più.

Argomenti:

Argomenti

Argomento	Descrizione
incremental	L'opzione incremental consente di leggere solo parte di un file sottostante. La dimensione precedente del file viene salvata nell'intestazione XML del file QVD. Queste informazioni risultano particolarmente utili con i file di registro. Tutti i record caricati nell'occasione precedente vengono letti dal file QVD, mentre i nuovi record seguenti vengono letti dalla sorgente originale, quindi viene creato un file QVD aggiornato. L'opzione incremental può essere utilizzata solo con le istruzioni **LOAD** e i file di testo. Il carico incrementale non può essere utilizzato quando i precedenti dati vengono cambiati o eliminati.
stale [after] amount [(days | hours)]	amount è un numero che specifica l'intervallo di tempo. Possono essere utilizzati valori decimali. Se omessa, verrà utilizzata l'unità di misura giorni. In genere, l'opzione stale after viene utilizzata con sorgenti DB i cui dati originali non dispongono di alcun indicatore temporale semplice. In alternativa, è possibile specificare per quanto tempo conservare lo snapshot QVD. Una clausola stale after dichiara semplicemente l'intervallo di tempo a partire dalla creazione del buffer QVD, trascorso il quale non verrà più considerato valido. Prima di quel tempo, il buffer QVD verrà utilizzato come sorgente dei dati e, trascorso l'intervallo specificato, verrà utilizzata la sorgente dati iniziale. Il file del buffer QVD verrà aggiornato automaticamente, quindi avrà inizio un nuovo intervallo.

Limiti:

Esistono numerose limitazioni, la più importante delle quali stabilisce che dovrà esistere necessariamente un'istruzione **LOAD** o **SELECT** per file alla base di qualsiasi istruzione complessa.

Example 1:

```
Buffer SELECT * from MyTable;
```

Example 2:

```
Buffer (stale after 7 days) SELECT * from MyTable;
```

Example 3:

```
Buffer (incremental) LOAD * from MyLog.log;
```

Concatenate

`Concatenate` è un prefisso di caricamento dello script che consente di aggiungere un set di dati a una tabella in memoria già esistente. Viene spesso utilizzato per aggiungere diversi set di dati transazionali a un'unica tabella dei fatti centrale, o per costruire insiemi di dati di riferimento comuni di un tipo specifico che provengono da più fonti. La sua funzionalità è simile a quella dell'operatore UNION di SQL.

La tabella risultante da un'operazione `concatenate` conterrà il set di dati originale con le nuove righe di dati aggiunte in fondo alla tabella. Le tabelle di origine e di destinazione possono presentare campi diversi. Se i campi sono diversi, la tabella risultante sarà ampliata per rappresentare il risultato combinato di tutti i campi presenti sia nella tabella di origine che in quella di destinazione.

Sintassi:

```
Concatenate[ (tablename ) ] ( loadstatement | selectstatement )
```

Argomenti

Argomento	Descrizione
`tablename`	Il nome di una tabella esistente. La tabella denominata sarà l'obiettivo dell'operazione `Concatenate` e tutti i record di dati caricati saranno aggiunti a quella tabella. Se il parametro `tablename` non viene utilizzato, la tabella di destinazione sarà l'ultima tabella caricata prima di questa istruzione.
`loadstatement/selectstatement`	L'argomento `loadstatement/selectstatement` che segue l'argomento `tablename` sarà concatenato alla tabella specificata.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio di funzione

Esempio	Risultato
`Concatenate (Transactions) Load …. ;`	I dati caricati nell'istruzione LOAD sotto il prefisso `Concatenate` verranno aggiunti alla tabella esistente in memoria denominata `Transactions` (supponendo che una tabella denominata `Transactions` sia stata caricata prima di questo punto dello script di caricamento).

Esempio 1 - Aggiunta di più set di dati a una tabella di destinazione con il prefisso di caricamento Concatenate

Script di caricamento e risultati

Panoramica

In questo esempio verranno caricati due script in ordine sequenziale.

- Il primo script di caricamento contiene un set di dati iniziale con date e importi che viene inviato a una tabella denominata `Transactions`.
- Il secondo script di caricamento contiene:
  - Un secondo set di dati che viene aggiunto al set di dati iniziale utilizzando il prefisso `Concatenate`. Questo set di dati contiene un campo aggiuntivo, `type`, che non è presente nel set di dati iniziale.
  - Il prefisso `Concatenate`.

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Primo script di caricamento

```
Transactions:
Load * Inline [

id, date, amount
3750, 08/30/2018, 23.56
3751, 09/07/2018, 556.31
3752, 09/16/2018, 5.75
3753, 09/22/2018, 125.00
3754, 09/22/2018, 484.21
3756, 09/22/2018, 59.18
3757, 09/23/2018, 177.42
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `amount`

Tabella dei risultati del primo script di caricamento

id	date	importo
3750	08/30/2018	23.56
3751	09/07/2018	556.31
3752	09/16/2018	5.75
3753	09/22/2018	125.00
3754	09/22/2018	484.21
3756	09/22/2018	59.18
3757	09/23/2018	177.42

La tabella mostra il set di dati iniziale.

Secondo script di caricamento

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto.

```
Concatenate(Transactions)
Load * Inline [
id, date, amount, type
3758, 10/01/2018, 164.27, Internal
3759, 10/03/2018, 384.00, External
3760, 10/06/2018, 25.82, Internal
3761, 10/09/2018, 312.00, Internal
3762, 10/15/2018, 4.56, Internal
3763, 10/16/2018, 90.24, Internal
3764, 10/18/2018, 19.32, External
];
```

Risultati

Caricare i dati e passare al foglio. Creare questo campo come una dimensione:

- `type`

Tabella dei risultati del secondo script di caricamento

id	date	importo	tipo
3750	08/30/2018	23.56	-
3751	09/07/2018	556.31	-
3752	09/16/2018	5.75	-
3753	09/22/2018	125.00	-
3754	09/22/2018	484.21	-

id	date	importo	tipo
3756	09/22/2018	59.18	-
3757	09/23/2018	177.42	-
3758	10/01/2018	164.27	Interne
3759	10/03/2018	384.00	Esterni
3760	10/06/2018	25.82	Interne
3761	10/09/2018	312.00	Interne
3762	10/15/2018	4.56	Interne
3763	10/16/2018	90.24	Interne
3764	10/18/2018	19.32	Esterni

Si notino i valori null nel campo `type` per i primi sette record caricati in cui `type` non era stato definito.

Esempio 2 - Aggiunta di più set di dati a una tabella di destinazione utilizzando la concatenazione implicita

Script di caricamento e risultati

Panoramica

Un caso tipico di applicazione implicita dei dati si ha quando si caricano diversi file di dati strutturati in modo identico e si desidera aggiungerli tutti a una tabella di destinazione.

Ad esempio, utilizzando `wildcards` nei nomi dei file con una sintassi come:

```
myTable:
Load * from [myFile_*.qvd] (qvd);
```

o in cicli che utilizzano costrutti come:

```
for each file in filelist('myFile_*.qvd')

myTable:
Load * from [$(file)] (qvd);

next file
```

La concatenazione implicita avverrà tra due tabelle caricate con campi di nome identico, anche se non sono definite una dopo l'altra nello script. Ciò può portare all'aggiunta involontaria di dati alle tabelle. Se non si vuole che una tabella secondaria con campi identici venga aggiunta in questo modo, utilizzare il prefisso load `NoConcatenate`. Rinominare la tabella con un tag nome alternativo non è sufficiente a impedire che si verifichi una concatenazione implicita. Per ulteriori informazioni, vedere NoConcatenate (page 90).

In questo esempio verranno caricati due script in ordine sequenziale.

- Il primo script di caricamento contiene un set di dati iniziale con quattro campi che viene inviato a una tabella denominata `Transactions`.
- Il secondo script di caricamento contiene un set di dati con gli stessi campi del primo set di dati.

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Primo script di caricamento

```
Transactions:
Load * Inline [
id, date, amount, type
3758, 10/01/2018, 164.27, Internal
3759, 10/03/2018, 384.00, External
3760, 10/06/2018, 25.82, Internal
3761, 10/09/2018, 312.00, Internal
3762, 10/15/2018, 4.56, Internal
3763, 10/16/2018, 90.24, Internal
3764, 10/18/2018, 19.32, External
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `amount`
- `type`

Tabella dei risultati del primo script di caricamento

id	date	tipo	importo
3758	10/01/2018	Interne	164.27
3759	10/03/2018	Esterni	384.00
3760	10/06/2018	Interne	25.82
3761	10/09/2018	Interne	312.00
3762	10/15/2018	Interne	4.56
3763	10/16/2018	Interne	90.24
3764	10/18/2018	Esterni	19.32

La tabella mostra il set di dati iniziale.

Secondo script di caricamento

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto.

```
Load * Inline [
id, date, amount, type
```

```
3765, 11/03/2018, 129.40, Internal
3766, 11/05/2018, 638.50, External
];
```

Risultati

Caricare i dati e passare al foglio.

Tabella dei risultati del secondo script di caricamento

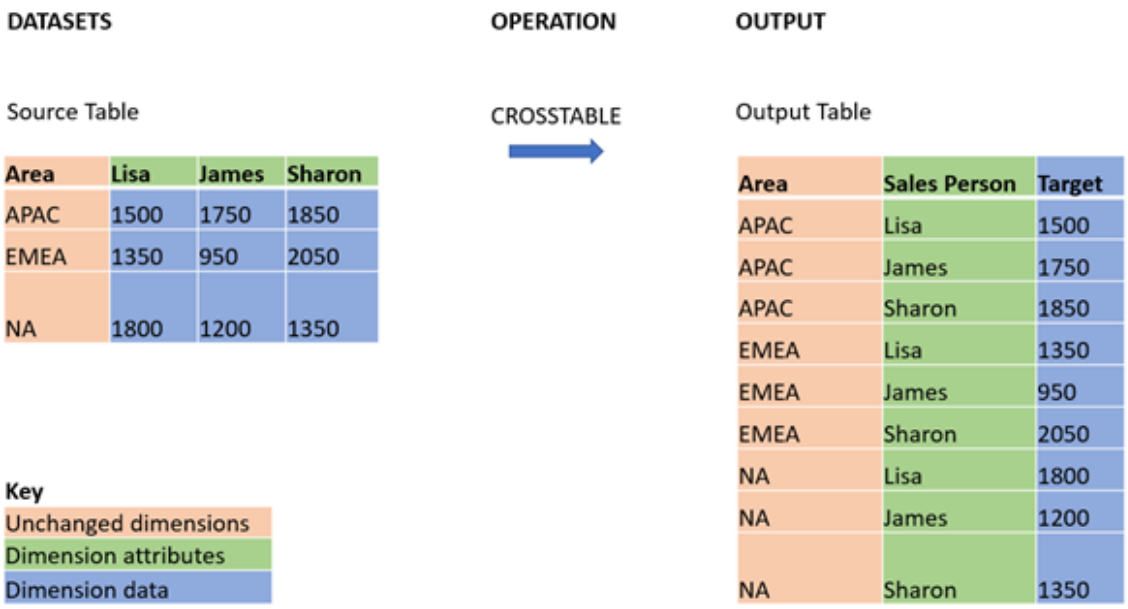
id	date	tipo	importo
3758	10/01/2018	Interne	164.27
3759	10/03/2018	Esterni	384.00
3760	10/06/2018	Interne	25.82
3761	10/09/2018	Interne	312.00
3762	10/15/2018	Interne	4.56
3763	10/16/2018	Interne	90.24
3764	10/18/2018	Esterni	19.32
3765	11/03/2018	Interne	129.40
3766	11/05/2018	Esterni	638.50

Il secondo set di dati è stato concatenato implicitamente al set di dati iniziale, poiché i campi sono identici.

Crosstable

Il prefisso di caricamento **crosstable** viene utilizzato per trasporre i dati strutturati in “tabella incrociata” o in "tabella pivot". I dati strutturati in questo modo si trovano comunemente quando si lavora con le sorgenti dei fogli di calcolo. L'output e lo scopo del prefisso di caricamento **crosstable** è di trasporre tali strutture nell'equivalente di una normale tabella in un formato a colonne, poiché questa struttura è generalmente più adatta per l'analisi in Qlik Sense.

Esempio di dati strutturati come tabella incrociata e struttura equivalente dopo una trasformazione crosstable



Sintassi:

```
crosstable (attribute field name, data field name [ , n ] ) ( loadstatement | selectstatement )
```

Argomenti

Argomento	Descrizione
attribute field name	Il nome del campo di output desiderato che descrive la dimensione orientata orizzontalmente da trasporre (la riga di intestazione).
data field name	Il nome del campo di output desiderato che descrive i dati orientati orizzontalmente della dimensione da trasporre (ma matrice dei valori di dati sotto la riga dell'intestazione).
n	Il numero di campi qualificatori, o le dimensioni invariate, che precedono la tabella da trasformare in un formato generico. Il valore predefinito è 1.

Questa funzione di script è correlata alle funzioni seguenti:

Funzioni correlate

Funzione	Interazione
Generic (page 58)	Un prefisso del carico di trasformazione che prende un set di dati strutturati entity-attribute-value e lo trasforma nella struttura di una normale tabella relazionale, separando ogni attributo rilevato in un nuovo campo o colonna di dati.

Esempio 1 – Trasformazione dei dati di vendita pivot (semplice)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere il primo script di caricamento di seguito in una nuova scheda.

Il primo script di caricamento contiene un set di dati a cui verrà successivamente applicato il prefisso dello script `crosstable`, con la sezione che applica `crosstable` con commenti. Ciò significa che la sintassi dei commenti è stata utilizzata per disabilitare questa sezione nello script di caricamento.

Il secondo script di caricamento è uguale al primo, ma con l'applicazione di `crosstable` senza commenti (abilitato rimuovendo la sintassi del commento). Gli script vengono mostrati in questo modo per evidenziare il valore di questa funzione di script nella trasformazione dei dati.

Primo script di caricamento (funzione non applicata)

```
tmpData:
//Crosstable (MonthText, Sales)
Load * inline [
Product, Jan 2021, Feb 2021, Mar 2021, Apr 2021, May 2021, Jun 2021
A, 100, 98, 103, 63, 108, 82
B, 284, 279, 297, 305, 294, 292
C, 50, 53, 50, 54, 49, 51];

//Final:
//Load Product,
//Date(Date#(MonthText,'MMM YYYY'),'MMM YYYY') as Month,
//Sales

//Resident tmpData;

//Drop Table tmpData;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `Product`
- `Jan 2021`
- `Feb 2021`
- `Mar 2021`
- `Apr 2021`
- `May 2021`
- `Jun 2021`

Tabella dei risultati

Prodotto	Gen 2021	Feb 2021	Mar 2021	Apr 2021	Mag 2021	Giu 2021
A	100	98	103	63	108	82
B	284	279	297	305	294	292
C	50	53	50	54	49	51

Questo script consente la creazione di una tabella incrociata con una colonna per ogni mese e una riga per prodotto. Nel suo formato attuale, questi dati non sono facili da analizzare. Sarebbe preferibile disporre di tutti i numeri in un campo e tutti i mesi in un altro, in una tabella a tre colonne. La sezione successiva spiega come eseguire questa trasformazione nella tabella incrociata.

Secondo script di caricamento (funzione non applicata)

Cancellare i commenti nello script rimuovendo `//`. Lo script di caricamento dovrebbe avere l'aspetto seguente:

```
tmpData:
Crosstable (MonthText, Sales)
Load * inline [
Product, Jan 2021, Feb 2021, Mar 2021, Apr 2021, May 2021, Jun 2021
A, 100, 98, 103, 63, 108, 82
B, 284, 279, 297, 305, 294, 292
C, 50, 53, 50, 54, 49, 51];

Final:
Load Product,
Date(Date#(MonthText,'MMM YYYY'),'MMM YYYY') as Month,
Sales

Resident tmpData;

Drop Table tmpData;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `Product`
- `Month`
- `Sales`

Tabella dei risultati

Prodotto	Month	Vendite
A	Jan 2021	100

Prodotto	Month	Vendite
A	Feb 2021	98
A	Mar 2021	103
A	Apr 2021	63
A	May 2021	108
A	Jun 2021	82
B	Jan 2021	284
B	Feb 2021	279
B	Mar 2021	297
B	Apr 2021	305
B	May 2021	294
B	Jun 2021	292
C	Jan 2021	50
C	Feb 2021	53
C	Mar 2021	50
C	Apr 2021	54
C	May 2021	49
C	Jun 2021	51

Una volta applicato il prefisso dello script, la tabella incrociata viene trasformata in una tabella lineare con una colonna per il valore `Month` e un'altra per `Sales`. Ciò migliora la leggibilità dei dati.

Esempio 2 – Trasformazione dei dati sugli obiettivi di vendita pivot in una tabella con struttura verticale (livello intermedio)

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella denominata Target.
- Il prefisso di caricamento `crosstable`, che traspone i nomi degli agenti dedicati alle vendite pivot in un campo a sé stante con l'etichetta `Sales Person`.
- I dati degli obiettivi di vendita associati, sono strutturati in un campo denominato `Target`.

Script di caricamento

```
SalesTargets:
CROSSTABLE([Sales Person],Target,1)
LOAD
*
INLINE [
Area, Lisa, James, Sharon
APAC, 1500, 1750, 1850
EMEA, 1350, 950, 2050
NA, 1800, 1200, 1350
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `Area`
- `Sales Person`

Aggiungere questa misura:

```
=Sum(Target)
```

Tabella dei risultati

Area	Venditore	=Sum(Target)
APAC	James	1750
APAC	Lisa	1500
APAC	Sharon	1850
EMEA	James	950
EMEA	Lisa	1350
EMEA	Sharon	2050
N/D	James	1200
N/D	Lisa	1800
N/D	Sharon	1350

Se si desidera replicare la visualizzazione dei dati come tabella pivot di input, è possibile creare una tabella pivot equivalente in un foglio.

Procedere come indicato di seguito:

1. Copiare e incollare la tabella appena creata nel foglio.
2. Trascinare l'oggetto grafico **Tabella pivot** sopra la copia della tabella appena creata. Selezionare **Converti**.

3. Fare clic su ✔ **Termina modifica**.
4. Trascinare il campo `Sales Person` dalla casella della colonna verticale alla casella della colonna orizzontale.

La tabella seguente mostra i dati nella forma della tabella iniziale, come è visualizzato in Qlik Sense:

Tabella dei risultati originale, come mostrato in Qlik Sense

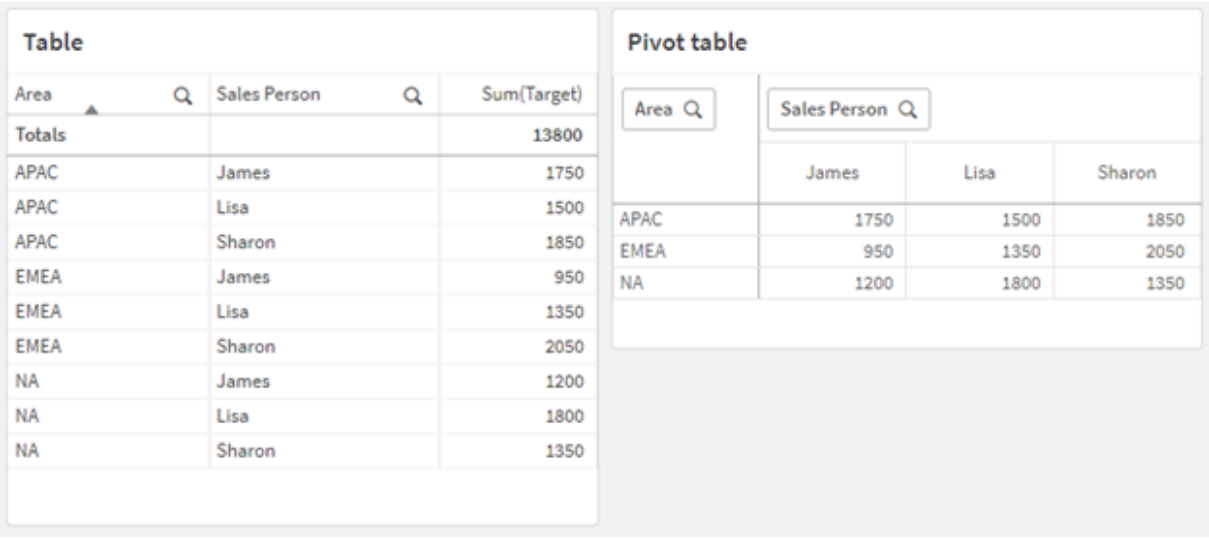
Area	Venditore	=Sum(Target)
Totali	-	13800
APAC	James	1750
APAC	Lisa	1500
APAC	Sharon	1850
EMEA	James	950
EMEA	Lisa	1350
EMEA	Sharon	2050
N/D	James	1200
N/D	Lisa	1800
N/D	Sharon	1350

La tabella pivot equivalente è simile a quella seguente, con la colonna per il nome di ogni dipendente addetto alle vendite contenuta all'interno della riga più grande per `Sales Person`:

Tabella pivot equivalente con il campo `Sales Person` ruotato orizzontalmente

Area	James	Lisa	Sharon
APAC	1750	1500	1850
EMEA	950	1350	2050
N/D	1350	1350	1350

Esempio di dati visualizzati come una tabella e una tabella pivot equivalente con il campo `Sales Person` ruotato orizzontalmente



Table

Area	Sales Person	Sum(Target)
Totals		13800
APAC	James	1750
APAC	Lisa	1500
APAC	Sharon	1850
EMEA	James	950
EMEA	Lisa	1350
EMEA	Sharon	2050
NA	James	1200
NA	Lisa	1800
NA	Sharon	1350

Pivot table

Area	James	Lisa	Sharon
APAC	1750	1500	1850
EMEA	950	1350	2050
NA	1200	1800	1350

Esempio 3 – Trasformazione dei dati e degli obiettivi di vendita pivot in una tabella con struttura verticale (avanzata)

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che rappresenta i dati sulle vendite e sugli obiettivi, organizzati per area e mese dell'anno. Questo viene caricato in una tabella denominata `SalesAndTargets`.
- Il prefisso di caricamento `crosstable`. Questo viene utilizzato per annullare il pivot della dimensione `Month Year` in un campo dedicato, nonché per trasporre la matrice delle vendite e degli importi target in un campo dedicato chiamato `Amount`.
- Conversione del campo `Month Year` da testo in una data corretta, utilizzando la funzione di conversione text-to-date `date#`. Il campo `Month Year` convertito in data viene unito nuovamente alla tabella `SalesAndTarget` tramite un prefisso di caricamento `Join`.

Script di caricamento

```
SalesAndTargets:
CROSSTABLE(MonthYearAsText,Amount,2)
LOAD
      *
INLINE [
Area   Type    Jan-22  Feb-22  Mar-22  Apr-22  May-22  Jun-22  Jul-22  Aug-22  Sep-22  Oct-22  Nov-22  Dec-22
APAC   Target  425     425     425     425     425     425     425     425     425     425     425     425
APAC   Actual  435     434     397     404     458     447     413     458     385     421     448     397
```

```
EMEA    Target  362.5   362.5   362.5   362.5   362.5   362.5   362.5   362.5   362.5   362.5   362.5   362.5
EMEA    Actual  363.5   359.5   337.5   361.5   341.5   337.5   379.5   352.5   327.5   337.5   360.5   334.5
NA      Target  375     375     375     375     375     375     375     375     375     375     375     375
NA      Actual  378     415     363     356     403     343     401     365     393     340     360     405
] (delimiter is '\t');

tmp:
LOAD DISTINCT MonthYearAsText,date#(MonthYearAsText,'MMM-YY') AS [Month Year]
RESIDENT SalesAndTargets;

JOIN (SalesAndTargets)
LOAD * RESIDENT tmp;

DROP TABLE tmp;
DROP FIELD MonthYearAsText;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `Area`
- `Month Year`

Creare le seguenti misure, con l'etichetta `Actual`:

```
=Sum({<Type={'Actual'}>} Amount)
```

Inoltre, creare questa misura, con l'etichetta `Target`:

```
=Sum({<Type={'Target'}>} Amount)
```

Tabella dei risultati (ritagliata)

Area	Anno mese	Effettivo	Target
APAC	Jan-22	435	425
APAC	Feb-22	434	425
APAC	Mar-22	397	425
APAC	Apr-22	404	425
APAC	May-22	458	425
APAC	Jun-22	447	425
APAC	Jul-22	413	425
APAC	Aug-22	458	425
APAC	Sep-22	385	425
APAC	Oct-22	421	425
APAC	Nov-22	448	425

Area	Anno mese	Effettivo	Target
APAC	Dec-22	397	425
EMEA	Jan-22	363.5	362.5
EMEA	Feb-22	359.5	362.5

Se si desidera replicare la visualizzazione dei dati come tabella pivot di input, è possibile creare una tabella pivot equivalente in un foglio.

Procedere come indicato di seguito:

1. Copiare e incollare la tabella appena creata nel foglio.
2. Trascinare l'oggetto grafico **Tabella pivot** sopra la copia della tabella appena creata. Selezionare **Converti**.
3. Fare clic su ✔ **Termina modifica**.
4. Trascinare il campo `Month Year` dalla casella della colonna verticale alla casella della colonna orizzontale.
5. Trascinare l'elemento `Values` dalla casella della colonna verticale alla casella della colonna orizzontale.

La tabella seguente mostra i dati nella forma della tabella iniziale, come è visualizzato in Qlik Sense:

Tabella dei risultati originale (ritagliata), come mostrata in Qlik Sense

Area	Anno mese	Effettivo	Target
Totali	-	13812	13950
APAC	Jan-22	435	425
APAC	Feb-22	434	425
APAC	Mar-22	397	425
APAC	Apr-22	404	425
APAC	May-22	458	425
APAC	Jun-22	447	425
APAC	Jul-22	413	425
APAC	Aug-22	458	425
APAC	Sep-22	385	425
APAC	Oct-22	421	425
APAC	Nov-22	448	425
APAC	Dec-22	397	425

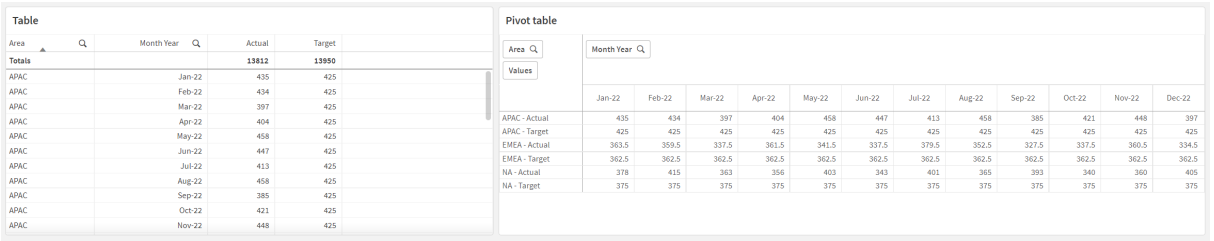
Area	Anno mese	Effettivo	Target
EMEA	Jan-22	363.5	362.5
EMEA	Feb-22	359.5	362.5

La tabella pivot equivalente è simile a quella seguente, con la colonna per ogni mese dell'anno contenuta all'interno della riga più grande per `Month Year`:

Tabella pivot (ritagliata) equivalente con il campo `Month Year` ruotato orizzontalmente

Area (valori)	Jan-22	Feb-22	Mar-22	Apr-22	May-22	Jun-22	Jul-22	Aug-22	Sep-22	Oct-22	Nov-22	Dec-22
APAC - Effettivo	435	434	397	404	458	447	413	458	385	421	448	397
APAC - Target	425	425	425	425	425	425	425	425	425	425	425	425
EMEA - Effettivo	363.5	359.5	337.5	361.5	341.5	337.5	379.5	352.5	327.5	337.5	360.5	334.5
EMEA - Target	362.5	362.5	362.5	362.5	362.5	362.5	362.5	362.5	362.5	362.5	362.5	362.5
N/D - Effettivo	378	415	363	356	403	343	401	365	393	340	360	405
N/D - Target	375	375	375	375	375	375	375	375	375	375	375	375

Esempio di dati visualizzati come una tabella e una tabella pivot equivalente con il campo `Month Year` ruotato orizzontalmente



First

Il prefisso `First` aggiunto a un'istruzione `LOAD` o `SELECT` (SQL) viene utilizzato per caricare un numero di record massimo impostato dalla tabella di origine dei dati. Un tipico caso d'uso per l'utilizzo del prefisso `First` è quando si desidera recuperare un piccolo sottoinsieme di record da una fase di caricamento dei dati di grandi dimensioni e/o lenta. Non appena il numero di record definito "`n`" viene caricato, la fase di caricamento termina prematuramente e il resto dell'esecuzione dello script continua normalmente.

Sintassi:

```
First n ( loadstatement | selectstatement )
```

Argomenti

Argomento	Descrizione
`n`	Un'espressione arbitraria che restituisce un numero intero indicante il numero massimo di record da leggere. È possibile anche racchiudere `n` tra parentesi: `(n)`.
`loadstatement | selectstatement`	Il valore `load statement/select statement` che segue l'argomento n definirà la tabella specificata che deve essere caricata con il numero massimo di record impostato.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`FIRST 10 LOAD * from abc.csv;`	Questo esempio recupererà le prime dieci righe da un file Excel.
`FIRST (1) SQL SELECT * from Orders;`	Questo esempio recupererà la prima riga selezionata dal set di dati `Orders`.

Esempio: Caricamento delle prime cinque righe

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati delle prime due settimane del 2020.
- La variabile `First` che indica all'applicazione di caricare solo i primi cinque record.

Script di caricamento

```
Sales:
FIRST 5
LOAD
*
Inline [
date,sales
01/01/2020,6000
01/02/2020,3000
01/03/2020,6000
01/04/2020,8000
01/05/2020,5000
01/06/2020,7000
01/07/2020,3000
01/08/2020,5000
01/09/2020,9000
01/10/2020,5000
01/11/2020,7000
01/12/2020,7000
01/13/2020,7000
01/14/2020,7000
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere `Date` e `sum(sales)` come misura.

Tabella dei risultati

Date	sum(sales)
01/01/2020	6000
01/02/2020	3000
01/03/2020	6000
01/04/2020	8000
01/05/2020	5000

Lo script carica solo i primi cinque record della tabella `Sales`.

Generic

Il prefisso di caricamento **Generic** consente la conversione dei dati modellati su entità-attributo-valore (EAV) in una struttura di tabella relazionale tradizionale e normalizzata. La modellazione EAV in alternativa è denominata "modellazione di dati generici" o "schema aperto".

Esempio di dati EAV modellati e tabella relazionale denormalizzata equivalente

Product ID	Attribute	Value
13	Status	Discontinued
13	Colour	Brown
20	Colour	White
13	Size	13-15
20	Size	16-18

Product ID	Status	Colour	Size
13	Discontinued	Brown	13-15
20		White	16-18

Esempio di dati EAV modellati e tabella relazionale denormalizzata equivalente

Product ID	Attribute	Value
13	Status	Discontinued
13	Colour	Brown
20	Colour	White
13	Size	13-15
20	Size	16-18

Product ID	Status
13	Discontinued

Product ID	Colour
13	Brown
20	White

Product ID	Size
13	13-15
20	16-18

Sebbene sia tecnicamente possibile caricare e analizzare i dati modellati EAV in Qlik, è spesso più semplice lavorare con una struttura di dati relazionale tradizionale equivalente.

Sintassi:

```
Generic( loadstatement | selectstatement )
```

I seguenti argomenti possono aiutarti a lavorare con questa funzione:

Argomenti correlati

Argomento	Descrizione
Crosstable (page 45)	Il prefisso di caricamento `Crosstable` trasforma i dati orientati orizzontalmente in dati orientati verticalmente. Da una prospettiva puramente funzionale, esegue la trasformazione opposta al prefisso di caricamento `Generic`, sebbene i prefissi servano in genere per casi d'uso completamente diversi.
Database generici in *Gestione dei dati*	I modelli di dati strutturati EAV sono ulteriormente descritti qui.

Esempio 1 – Trasformazione di dati strutturati EAV con il prefisso di caricamento generico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene un set di dati che viene caricato in una tabella denominata `Transactions`. Il set di dati include un campo data. Viene utilizzata la definizione predefinita `MonthNames`.

Script di caricamento

```
Products:
Generic
Load * inline [
Product ID, Attribute, Value
13, Status, Discontinued
13, Color, Brown
20, Color, White
13, Size, 13-15
20, Size, 16-18
2, Status, Discontinued
5, Color, Brown
2, Color, White
44, Color, Brown
45, Size, 16-18
45, Color, Brown
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `Color`.

Aggiungere questa misura:

```
=Count([Product ID])
```

Ora è possibile controllare il numero di prodotti per colore.

Tabella dei risultati

Colore	=Count([Product ID])
Marrone	4
Bianco	2

Si noti la forma del modello dati, in cui ogni attributo è stato suddiviso in una tabella separata denominata in base al contrassegno della tabella di destinazione originale `Product`. Ogni tabella ha l'attributo come suffisso. Un esempio è `Product.Color`. I record di output dell'Attributo prodotto risultanti sono associati per `Product ID`.

Rappresentazione dei risultati del sistema di visualizzazione modello dati

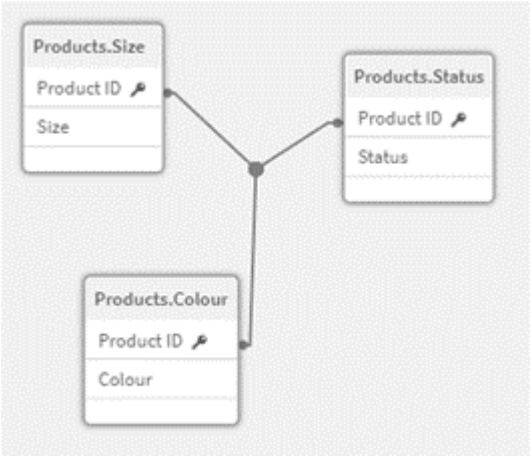


Tabella dei record risultante: Products.Status

Product ID	Stato
13	Interrotto
2	Interrotto

Tabella dei record risultante: Products.Size

Product ID	Dimensioni
13	13-15
20	16-18
45	16-18

Tabella dei record risultante: Products.Color

Product ID	Colore
13	Marrone
5	Marrone
44	Marrone
45	Marrone
20	Bianco
2	Bianco

Esempio 2 – Analisi dei dati strutturati EAV senza il prefisso di caricamento generico

Script di caricamento ed espressione del grafico

Panoramica

Questo esempio mostra come analizzare i dati strutturati EAV nella loro forma originale.

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene un set di dati che viene caricato in una tabella denominata `Products` in una struttura EAV.

In questo esempio, i prodotti continuano ad essere contati per attributo colore. Per analizzare i dati strutturati in questo modo, è necessario applicare un filtro a livello di espressione per i prodotti con il valore Attributo `Color`.

Inoltre, i singoli attributi non sono disponibili per la selezione come dimensioni o campi, rendendo più difficile determinare come creare visualizzazioni efficaci.

Script di caricamento

```
Products:
Load * Inline
[
Product ID, Attribute, Value
13, Status, Discontinued
13, Color, Brown
20, Color, White
13, Size, 13-15
20, Size, 16-18
2, Status, Discontinued
5, Color, Brown
2, Color, White
44, Color, Brown
45, Size, 16-18
```

```
45, Color, Brown
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `Value`.

Creare la seguente misura:

```
=Count({<Attribute={'Color'}>} [Product ID])
```

Ora è possibile controllare il numero di prodotti per colore.

Tabella dei record risultante: Products.Status

Value	=Count({<Attribute={'Color'}>} [Product ID])
Marrone	4
Bianco	2

Esempio 3 – Denormalizzazione delle tabelle di output risultanti da un caricamento generico (avanzato)

Script di caricamento ed espressione del grafico

Panoramica

In questo esempio, viene mostrato come la struttura dati normalizzata prodotta dal prefisso di caricamento `Generic` può essere denormalizzata nuovamente in una tabella dimensionale `Product` consolidata. Questa è una tecnica di modellazione avanzata che può essere impiegata come parte dell'ottimizzazione delle prestazioni del modello dati.

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Script di caricamento

```
Products:

Generic
Load * inline [
Product ID, Attribute, Value
13, Status, Discontinued
13, Color, Brown
20, Color, White
13, Size, 13-15
20, Size, 16-18
2, Status, Discontinued
5, Color, Brown
2, Color, White
44, Color, Brown
45, Size, 16-18
```

```
45, Color, Brown
];

RENAME TABLE Products.Color TO Products;

OUTER JOIN (Products)
LOAD * RESIDENT Products.Size;

OUTER JOIN (Products)
LOAD * RESIDENT Products.Status;
DROP TABLES Products.Size,Products.Status;
```

Risultati

Aprire il sistema di visualizzazione modello dati e annotare la forma del modello di dati risultante. È presente una sola tabella denormalizzata. Questa è una combinazione delle tre tabelle di output intermedie: `Products.Size`, `Products.Status` e `Products.Color`.

Modello di dati interni risultante

Prodotti
Product ID
Stato
Colore
Dimensioni

Tabella dei record risultante: Prodotti

Product ID	Stato	Colore	Dimensioni
13	Interrotto	Marrone	13-15
20	-	Bianco	16-18
2	Interrotto	Bianco	-
5	-	Marrone	-
44	-	Marrone	-
45	-	Marrone	16-18

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `Color`.

Aggiungere questa misura:

```
=Count([Product ID])
```

Tabella dei risultati

Colore	=Count([Product ID])
Marrone	4
Bianco	2

Hierarchy

Il prefisso **hierarchy** viene utilizzato per trasformare una tabella gerarchica padre-figlio in una tabella utile in un modello dati Qlik Sense. Può essere inserito prima di un'istruzione **LOAD** o **SELECT** e utilizzerà i risultati dell'istruzione di caricamento come input per la trasformazione della tabella.

Il prefisso crea una tabella di nodi espansi che, in generale, presenta lo stesso numero di record della tabella di input, ma dove ogni livello all'interno della gerarchia viene memorizzato in un campo separato. Il campo del percorso può essere utilizzato in una struttura ad albero.

Sintassi:

```
Hierarchy (NodeID, ParentID, NodeName, [ParentName, [PathSource, [PathName,
[PathDelimiter, Depth]]]])(loadstatement | selectstatement)
```

La tabella di input deve essere una tabella di nodi adiacenti. Le tabelle di nodi adiacenti sono tabelle in cui ogni record corrisponde a un nodo e presenta un campo contenente un riferimento al nodo padre. In questa tabella il nodo è salvato solamente su un record, anche se può presentare un qualsiasi numero di figli. Ovviamente la tabella può contenere campi aggiuntivi che descrivono gli attributi dei nodi.

Il prefisso crea una tabella di nodi espansi che, in generale, presenta lo stesso numero di record della tabella di input, ma dove ogni livello all'interno della gerarchia viene memorizzato in un campo separato. Il campo del percorso può essere utilizzato in una struttura ad albero.

In generale, la tabella di input presenta esattamente un record per nodo; in questi casi, la tabella di output contiene lo stesso numero di record. Tuttavia, a volte esistono nodi con più padri, ad esempio un nodo è rappresentato da più record nella tabella di input. In questo caso, la tabella di output potrà avere più record della tabella di input.

Tutti i nodi con un ID padre non presente nella colonna dell'ID del nodo (inclusi i nodi con ID padre mancanti) verranno considerati nodi radice. Inoltre, verranno caricati solo i nodi con una connessione, diretta o indiretta, al nodo radice, in modo da evitare riferimenti circolari.

È anche possibile creare campi aggiuntivi contenenti il nome del nodo padre, il percorso del nodo e la relativa profondità.

Argomenti:

Argomenti

Argomento	Descrizione
NodeID	Il nome del campo contenente l'ID del nodo. Questo campo deve esistere nella tabella di input.
ParentID	Il nome del campo contenente l'ID nodo del nodo padre. Questo campo deve esistere nella tabella di input.
NodeName	Il nome del campo contenente il nome del nodo. Questo campo deve esistere nella tabella di input.
ParentName	Una stringa utilizzata per assegnare un nome al nuovo campo **ParentName**. Se omessa, il campo non verrà creato.
ParentSource	Il nome del campo contenente il nome del nodo utilizzato per creare il percorso del nodo. È un parametro opzionale. Se omesso, verrà utilizzato **NodeName**.
PathName	Una stringa utilizzata per assegnare un nome al nuovo campo **Path**, che contiene il percorso dalla radice al nodo. È un parametro opzionale. Se omessa, il campo non verrà creato.
PathDelimiter	Una stringa utilizzata come delimitatore nel nuovo campo **Path**. È un parametro opzionale. Se omesso, sarà utilizzato il simbolo '/'.
Depth	Una stringa utilizzata per assegnare un nome al nuovo campo **Depth**, il quale contiene la profondità del nodo nella gerarchia. È un parametro opzionale. Se omessa, il campo non verrà creato.

Esempio:

```
Hierarchy(NodeID, ParentID, NodeName, ParentName, NodeName, PathName, '\', Depth) LOAD *
inline [
NodeID, ParentID, NodeName
1, 4, London
2, 3, Munich
3, 5, Germany
4, 5, UK
5, , Europe
];
```

NodeID	ParentID	NodeName	NodeName1	NodeName2	NodeName3	ParentName	PathName	Depth
1	4	London	Europe	UK	London	UK	Europe\UK\London	3
2	3	Munich	Europe	Germany	Munich	Germany	Europe\Germany\Munich	3
3	5	German	Europe	Germany	-	Europe	Europe\Germany	2

		y						
4	5	UK	Europe	UK	-	Europe	Europe\UK	2
5		Europe	Europe	-	-	-	Europe	1

HierarchyBelongsTo

Questo prefisso viene utilizzato per trasformare una tabella gerarchica padre-figlio in una tabella utile in un modello dati Qlik Sense. Può essere inserito prima di un'istruzione **LOAD** o **SELECT** e utilizzerà i risultati dell'istruzione di caricamento come input per la trasformazione della tabella.

Il prefisso consente di creare una tabella contenente tutte le relazioni padre-figlio della gerarchia. I campi padre possono essere quindi utilizzati per selezionare intere sezioni di tale gerarchia. Nella maggior parte dei casi, la tabella di output contiene più record per nodo.

Sintassi:

```
HierarchyBelongsTo (NodeID, ParentID, NodeName, AncestorID, AncestorName,
[DepthDiff])(loadstatement | selectstatement)
```

La tabella di input deve essere una tabella di nodi adiacenti. Le tabelle di nodi adiacenti sono tabelle in cui ogni record corrisponde a un nodo e presenta un campo contenente un riferimento al nodo padre. In questa tabella il nodo è salvato solamente su un record, anche se può presentare un qualsiasi numero di figli. Ovviamente la tabella può contenere campi aggiuntivi che descrivono gli attributi dei nodi.

Il prefisso consente di creare una tabella contenente tutte le relazioni padre-figlio della gerarchia. I campi padre possono essere quindi utilizzati per selezionare intere sezioni di tale gerarchia. Nella maggior parte dei casi, la tabella di output contiene più record per nodo.

Può essere creato un campo aggiuntivo contenente la differenza di profondità dei nodi.

Argomenti:

Argomenti

Argomento	Descrizione
NodeID	Il nome del campo contenente l'ID del nodo. Questo campo deve esistere nella tabella di input.
ParentID	Il nome del campo contenente l'ID nodo del nodo padre. Questo campo deve esistere nella tabella di input.
NodeName	Il nome del campo contenente il nome del nodo. Questo campo deve esistere nella tabella di input.
AncestorID	Una stringa utilizzata per assegnare un nome al nuovo campo ID padre, contenente l'ID del nodo padre.

Argomento	Descrizione
AncestorName	Una stringa utilizzata per assegnare un nome al nuovo campo padre, contenente il nome del nodo padre.
DepthDiff	Una stringa utilizzata per assegnare un nome al nuovo campo **DepthDiff**, contenente la profondità del nodo nella gerarchia relativa al nodo padre. È un parametro opzionale. Se omessa, il campo non verrà creato.

Esempio:

```
HierarchyBelongsTo (NodeID, AncestorID, NodeName, AncestorID, AncestorName, DepthDiff) LOAD *
inline [
NodeID, AncestorID, NodeName
1, 4, London
2, 3, Munich
3, 5, Germany
4, 5, UK
5, , Europe
];
```

Results

NodeID	AncestorID	NodeName	AncestorName	DepthDiff
1	1	London	London	0
1	4	London	UK	1
1	5	London	Europe	2
2	2	Munich	Munich	0
2	3	Munich	Germany	1
2	5	Munich	Europe	2
3	3	Germany	Germany	0
3	5	Germany	Europe	1
4	4	UK	UK	0
4	5	UK	Europe	1
5	5	Europe	Europe	0

Inner

I prefissi **join** e **keep** possono essere preceduti dal prefisso **inner**. Se viene inserito prima di **join**, specifica che occorre utilizzare un'unione interna. La tabella risultante contiene solo le combinazioni di valori di campo estratte dalle tabelle di dati non elaborati dove i valori di campo di collegamento vengono rappresentati in entrambe le tabelle. Se utilizzato prima di **keep**, specifica che entrambe le tabelle di dati non elaborati devono essere ridotte alla loro intersezione comune prima di essere memorizzate in Qlik Sense.

Sintassi:

```
Inner ( Join | Keep) [ (tablename) ](loadstatement |selectstatement )
```

Argomenti:

Argomenti

Argomento	Descrizione
tablename	La tabella denominata da confrontare con la tabella caricata.
loadstatementoppure selectstatement	L'istruzione **LOAD** o **SELECT** per la tabella caricata.

Esempio

Script di caricamento

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
Table1:
Load * inline [
Column1, Column2
A, B
1, aa
2, cc
3, ee ];

Table2:
Inner Join Load * inline [
Column1, Column3
A, C
1, xx
4, yy ];
```

Risultato

Tabella risultante

Column1	Column2	Column3
A	B	C
1	aa	xx

Spiegazione

Questo esempio dimostra l'output Inner Join dove vengono uniti solo i valori presenti sia nella prima (sinistra) che nella seconda (destra) tabella.

IntervalMatch

Il prefisso **IntervalMatch** consente di creare una tabella che corrisponde sia ai valori numerici discreti su uno o più intervalli numerici che, in modo opzionale, ai valori di una o più chiavi aggiuntive.

Sintassi:

```
IntervalMatch (matchfield)(loadstatement | selectstatement )
IntervalMatch (matchfield,keyfield1 [ , keyfield2, ... keyfield5 ] )
(loadstatement | selectstatement )
```

Il prefisso **IntervalMatch** deve essere inserito prima di un'istruzione **LOAD** o **SELECT** che carica gli intervalli. Il campo che contiene i punti dati discreti (Time nell'esempio seguente) e chiavi aggiuntive deve essere già stato caricato in Qlik Sense prima dell'istruzione con il prefisso **IntervalMatch**. Il prefisso non è in grado di leggere questo campo dalla tabella del database, pertanto trasforma la tabella caricata degli intervalli e delle chiavi in una tabella contenente una colonna aggiuntiva: i punti dati numerici discreti. Inoltre, espande il numero di record in modo che la nuova tabella disponga di un record per ogni combinazione possibile di punti dati discreti, intervallo e valore dei campi chiave.

Gli intervalli possono sovrapporsi e i valori discreti saranno collegati a tutti gli intervalli corrispondenti.

Quando il prefisso IntervalMatch viene esteso con i campi chiave, consente di creare una tabella che corrisponde sia ai valori numerici discreti presenti su uno o più intervalli numerici sia ai valori di una o più chiavi aggiuntive.

Per evitare che i limiti degli intervalli non definiti vengano ignorati, potrebbe essere necessario consentire il mapping dei valori NULL sugli altri campi che costituiscono i limiti inferiore e superiore dell'intervallo. Questa operazione può essere eseguita dall'istruzione **NullAsValue** o da un test esplicito che sostituisce i valori NULL con un valore numerico prima o dopo qualsiasi dei punti dati numerici discreti.

Argomenti:

Argomenti

Argomento	Descrizione
matchfield	Il campo contenente i valori numerici discreti da collegare agli intervalli.
keyfield	I campi contenenti gli attributi aggiuntivi da associare nella trasformazione.
loadstatement orselectstatement	Il risultato deve essere una tabella in cui il primo campo contiene il limite inferiore di ciascun intervallo, il secondo campo contiene il limite superiore di ciascun intervallo e, nel caso di utilizzo di una corrispondenza chiave, il terzo campo e quelli successivi contengono gli elementi keyfield presenti nell'istruzione **IntervalMatch**. Gli intervalli sono sempre chiusi, ossia i punti di fine sono inclusi nell'intervallo. I limiti non numerici fanno in modo che l'intervallo venga ignorato (non definito).

Example 1:

Nelle due tabelle seguenti, la prima tabella contiene un elenco di eventi discreti, mentre la seconda definisce l'ora di inizio e l'ora di fine relative alla produzione di ordini differenti. Utilizzando il prefisso **IntervalMatch**, è possibile eseguire il collegamento logico delle due tabelle in modo da poter individuare, ad esempio, gli ordini che hanno subito interruzioni e gli ordini elaborati in base a turni specifici.

```
EventLog:
LOAD * Inline [
Time, Event, Comment
00:00, 0, Start of shift 1
01:18, 1, Line stop
02:23, 2, Line restart 50%
04:15, 3, Line speed 100%
08:00, 4, Start of shift 2
11:43, 5, End of production
];

OrderLog:
LOAD * INLINE [
Start, End, Order
01:00, 03:35, A
02:30, 07:58, B
03:04, 10:27, C
07:23, 11:43, D
];

//Link the field Time to the time intervals defined by the fields Start and End.
Inner Join IntervalMatch ( Time )
LOAD Start, End
Resident OrderLog;
```

La tabella **OrderLog** contiene ora una colonna aggiuntiva: *Time*. Anche il numero di record risulta espanso.

Table with additional column

Time	Start	End	Order
00:00	-	-	-
01:18	01:00	03:35	A
02:23	01:00	03:35	A
04:15	02:30	07:58	B
04:15	03:04	10:27	C
08:00	03:04	10:27	C
08:00	07:23	11:43	D
11:43	07:23	11:43	D

Example 2: (mediante keyfield)

Lo stesso esempio illustrato in precedenza, con l'aggiunta di *ProductionLine* come campo chiave.

```
EventLog:
LOAD * Inline [
Time, Event, Comment, ProductionLine
00:00, 0, Start of shift 1, P1
01:00, 0, Start of shift 1, P2
```

```
01:18, 1, Line stop, P1
02:23, 2, Line restart 50%, P1
04:15, 3, Line speed 100%, P1
08:00, 4, Start of shift 2, P1
09:00, 4, Start of shift 2, P2
11:43, 5, End of production, P1
11:43, 5, End of production, P2
];

OrderLog:
LOAD * INLINE [
Start, End, Order, ProductionLine
01:00, 03:35, A, P1
02:30, 07:58, B, P1
03:04, 10:27, C, P1
07:23, 11:43, D, P2
];

//Link the field Time to the time intervals defined by the fields Start and End and match the
values
// to the key ProductionLine.
Inner Join
IntervalMatch ( Time, ProductionLine )
LOAD Start, End, ProductionLine
Resident OrderLog;
```

È ora possibile creare una tabella come quella seguente:

Tablebox example

ProductionLine	Time	Event	Comment	Order	Start	End
P1	00:00	0	Start of shift 1	-	-	-
P2	01:00	0	Start of shift 1	-	-	-
P1	01:18	1	Line stop	A	01:00	03:35
P1	02:23	2	Line restart 50%	A	01:00	03:35
P1	04:15	3	Line speed 100%	B	02:30	07:58
P1	04:15	3	Line speed 100%	C	03:04	10:27
P1	08:00	4	Start of shift 2	C	03:04	10:27
P2	09:00	4	Start of shift 2	D	07:23	11:43
P1	11:43	5	End of production	-	-	-
P2	11:43	5	End of production	D	07:23	11:43

Join

Il prefisso **join** unisce la tabella caricata a una tabella denominata esistente oppure all'ultima tabella di dati creata in precedenza.

L'effetto dell'unione dei dati è quello di estendere la tabella di destinazione di un insieme aggiuntivo di campi o attributi, vale a dire quelli che non sono già presenti nella tabella di destinazione. Qualsiasi nome di campo comune tra il set di dati di origine e la tabella di destinazione viene utilizzato per determinare come associare i nuovi record in ingresso. Questa viene comunemente indicata come "unione naturale". Un'operazione Join Qlik può fare in moco che la tabella di destinazione risultante abbia più o meno record di quelli con cui è stata creata, a seconda dell'unicità dell'associazione join e del tipo di operazione join utilizzato.

Esistono quattro tipi di join:

Left join

Le operazioni Left join sono il tipo più utilizzato. Ad esempio, se si dispone di un set di dati per una transazione e si desidera combinarlo con un set di dati di riferimento, in genere si utilizza `Left Join`. Per prima cosa è necessario caricare la tabella delle transazioni, quindi caricare il set di dati di riferimento e unirlo tramite un prefisso `Left Join` alla tabella delle transazioni già caricata. `Left Join` consente di mantenere tutte le transazioni così come sono e di aggiungere i campi dati di riferimento supplementari in cui viene trovata una corrispondenza.

Inner join

Quando si dispone di due set di dati in cui sono importanti solo i risultati in cui è presente un'associazione di corrispondenza, considerare l'utilizzo di `Inner Join`. Ciò consente di eliminare tutti i record sia dai dati di origine caricati che dalla tabella di destinazione se non viene trovata alcuna corrispondenza. Di conseguenza, l'operazione può la tabella di destinazione con meno record rispetto a prima che venisse completata l'operazione join.

Outer join

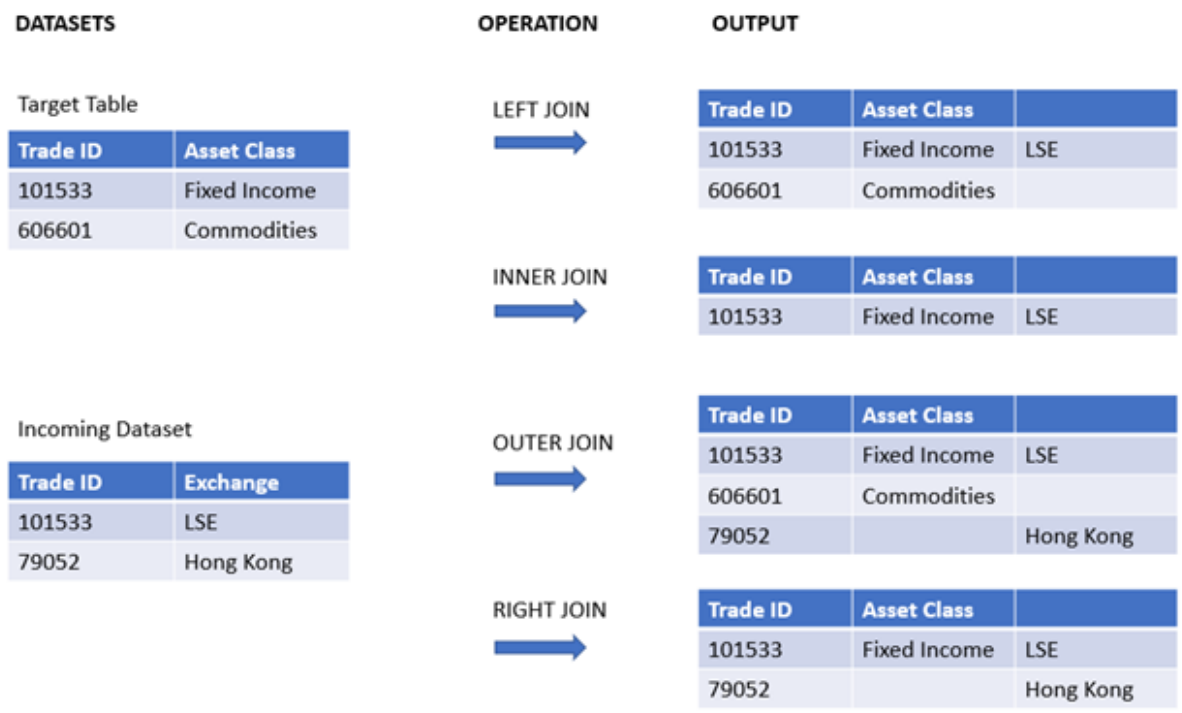
Quando è necessario conservare sia i record di destinazione che tutti i record in entrata, utilizzare `Outer Join`. Se non viene trovata alcuna corrispondenza, ogni set di record viene comunque conservato, mentre i campi dal lato opposto del JOIN non verranno popolati (null).

Se la parola chiave del tipo viene omessa, il tipo di join predefinito è un outer join.

Right join

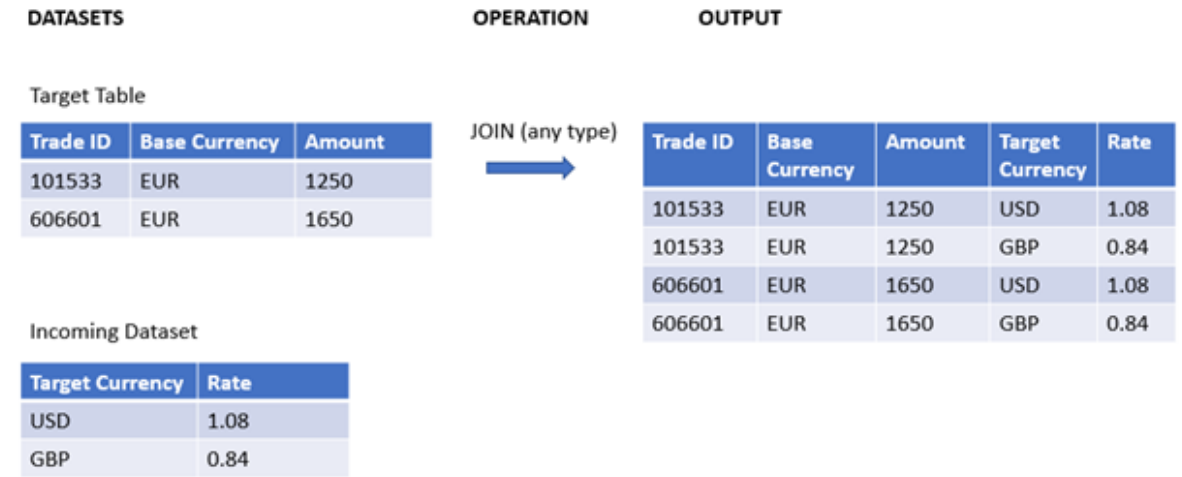
Questo tipo di join mantiene tutti i record che stanno per essere caricati, riducendo i record nella tabella di destinazione tramite l'operazione di join solo ai record in cui è presente una corrispondenza di associazione nei record in entrata. Questo è un tipo di join di nicchia che viene talvolta utilizzato come mezzo per ridurre una tabella di record già caricata precedentemente in un sottoinsieme richiesto.

Esempi di set di risultati da diversi tipi di operazioni join



Se non ci sono nomi di campo in comune tra l'origine e la destinazione di un'operazione join, questa darà come risultato un prodotto cartesiano di tutte le righe, che viene definito "cross join".

Esempio di set di risultati da un'operazione di "cross join".



Sintassi:

```
[inner | outer | left | right ]Join [ (tablename ) ]( loadstatement | selectstatement )
```

Argomenti

Argomento	Descrizione
tablename	La tabella denominata da confrontare con la tabella caricata.
loadstatementoppure selectstatement	L'istruzione **LOAD** o **SELECT** per la tabella caricata.

I seguenti argomenti possono aiutarti a lavorare con questa funzione:

Argomenti correlati

Argomento	Descrizione
Combinazione di tabelle con Join e Keep in *Gestione dei dati*	Questo argomento fornisce un'ulteriore spiegazione dei concetti di "unione" e "mantenimento" dei set di dati.
Keep (page 81)	Il prefisso di caricamento `Keep` è simile al prefisso `Join`, ma non combina i set di dati di origine e di destinazione. Invece, ritaglia ogni set di dati in base al tipo di operazione adottata (inner, outer, left o right).

Esempio 1 - Left join: Miglioramento dei dati di una tabella di destinazione con un set di dati di riferimento

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che rappresenta i record delle modifiche, che viene caricato in una tabella denominata `Changes`. Questa include il campo chiave "Status ID".
- Un secondo set di dati che rappresenta gli stati di modifica, che viene caricato e combinato con i record di modifica originali unendoli con un prefisso di caricamento `Join` a sinistra.

Questa operazione di left join assicura che i record di modifica rimangano intatti durante l'aggiunta di attributi di stato in cui viene trovata una corrispondenza nei record di stato in entrata in base a un ID di stato comune.

Script di caricamento

```
Changes:
Load * inline [
Change ID      Status ID       Scheduled Start Date    Scheduled End Date      Business Impact
10030  4       19/01/2022      23/02/2022      None
10015  3       04/01/2022      15/02/2022      Low
```

```
10103  1       02/04/2022      29/05/2022      Medium
10185  2       23/06/2022      08/09/2022      None
10323  1       08/11/2022      26/11/2022      High
10326  2       11/11/2022      05/12/2022      None
10138  2       07/05/2022      03/08/2022      None
10031  3       20/01/2022      25/03/2022      Low
10040  1       29/01/2022      22/04/2022      None
10134  1       03/05/2022      08/07/2022      Low
10334  2       19/11/2022      06/02/2023      Low
10220  2       28/07/2022      06/09/2022      None
10264  1       10/09/2022      17/10/2022      Medium
10116  1       15/04/2022      24/04/2022      None
10187  2       25/06/2022      24/08/2022      Low
] (delimiter is '\t');

Status:
Join (Changes)
Load * inline [
Status ID      Status  Sub Status
1      Open    Not Started
2      Open    Started
3      Closed  Complete
4      Closed  Cancelled
] (delimiter is '\t');
```

Risultati

Aprire il sistema di visualizzazione modello dati e annotare la forma del modello di dati. È presente una sola tabella denormalizzata. Questa comprende una combinazione di tutti i record di modifica originali, con gli attributi di stato corrispondenti uniti a ciascun record di modifica.

Modello di dati interni risultante

Modifiche
Cambia ID
ID stato
Data di inizio programmata
Data di fine programmata
Impatto aziendale
Stato
Stato secondario

Se si espande la finestra di anteprima nel sistema di visualizzazione modello dati, viene visualizzata una parte di questo set di risultati completo organizzato in una tabella:

Anteprima della tabella Modifiche nel sistema di visualizzazione modello dati

Cambia ID	ID stato	Data di inizio programmata	Data di fine programmata	Impatto aziendale	Stato	Stato secondario
10015	3	04/01/2022	15/02/2022	Scarso	Chiuso	Completato
10030	4	19/01/2022	23/02/2022	Nessuno	Chiuso	Annullato
10031	3	20/01/2022	25/03/2022	Scarso	Chiuso	Completato
10040	1	29/01/2022	22/04/2022	Nessuno	Apri	Non avviato
10103	1	02/04/2022	29/05/2022	Medio	Aperto	Non avviato
10116	1	15/04/2022	24/04/2022	Nessuno	Apri	Non avviato
10134	1	03/05/2022	08/07/2022	Scarso	Apri	Non avviato
10138	2	07/05/2022	03/08/2022	Nessuno	Aperto	Avviato
10185	2	23/06/2022	08/09/2022	Nessuno	Aperto	Avviato
10187	2	25/06/2022	24/08/2022	Scarso	Aperto	Avviato
10220	2	28/07/2022	06/09/2022	Nessuno	Aperto	Avviato
10264	1	10/09/2022	17/10/2022	Medio	Aperto	Non avviato
10323	1	08/11/2022	26/11/2022	Elevati	Aperto	Non avviato
10326	2	11/11/2022	05/12/2022	Nessuno	Aperto	Avviato
10334	2	19/11/2022	06/02/2023	Scarso	Aperto	Avviato

Tornare all'Editor caricamento dati. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `Status`.

Aggiungere questa misura:

```
=Count([Change ID])
```

Ora è possibile verificare il numero di modifiche per stato.

Tabella dei risultati

Stato	=Count([Change ID])
Aperto	12
Chiuso	3

Esempio 2 – Inner join: Combinazione solo dei record corrispondenti

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che rappresenta i record delle modifiche, che viene caricato in una tabella denominata `Changes`.
- Un secondo set di dati che rappresenta i record delle modifiche originate dal sistema di origine `JIRA`. Questo viene caricato e combinato con i record originali unendoli con un prefisso di caricamento `Inner Join`.

`Inner Join` garantisce che vengano mantenuti solo i cinque record delle modifiche che si trovano in entrambi i set di dati.

Script di caricamento

```
Changes:
Load * inline [
Change ID      Status ID       Scheduled Start Date    Scheduled End Date      Business Impact
10030  4       19/01/2022      23/02/2022      None
10015  3       04/01/2022      15/02/2022      Low
10103  1       02/04/2022      29/05/2022      Medium
10185  2       23/06/2022      08/09/2022      None
10323  1       08/11/2022      26/11/2022      High
10326  2       11/11/2022      05/12/2022      None
10138  2       07/05/2022      03/08/2022      None
10031  3       20/01/2022      25/03/2022      Low
10040  1       29/01/2022      22/04/2022      None
10134  1       03/05/2022      08/07/2022      Low
10334  2       19/11/2022      06/02/2023      Low
10220  2       28/07/2022      06/09/2022      None
10264  1       10/09/2022      17/10/2022      Medium
10116  1       15/04/2022      24/04/2022      None
10187  2       25/06/2022      24/08/2022      Low
] (delimiter is '\t');

JIRA_changes:
Inner Join (Changes)
Load
  [Ticket ID] AS [Change ID],
  [Source System]
inline
[
Ticket ID      Source System
10030  JIRA
10323  JIRA
10134  JIRA
10334  JIRA
10220  JIRA
10187  JIRA
] (delimiter is '\t');
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `Source System`
- `Change ID`
- `Business Impact`

Ora è possibile esaminare i cinque record risultanti.

Tabella dei risultati

Sistema di origine	Cambia ID	Impatto aziendale
JIRA	10030	Nessuno
JIRA	10134	Scarso
JIRA	10220	Nessuno
JIRA	10323	Elevati
JIRA	10334	Scarso

Esempio 3 – Outer join: Combinazione di set di record sovrapposti

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che rappresenta i record delle modifiche, che viene caricato in una tabella denominata `Changes`.
- Un secondo set di dati che rappresenta i record delle modifiche originate dal sistema di origine `JIRA`, che viene caricato e combinato con i record originali unendoli con un prefisso di caricamento `Outer Join`.

Ciò garantisce che tutti i record delle modifiche sovrapposte di entrambi i set di dati vengano mantenuti.

Script di caricamento

```
// 8 Change records

Changes:
Load * inline [
Change ID      Status ID       Scheduled Start Date    Scheduled End Date      Business Impact
10030  4       19/01/2022      23/02/2022      None
10015  3       04/01/2022      15/02/2022      Low
10138  2       07/05/2022      03/08/2022      None
10031  3       20/01/2022      25/03/2022      Low
10040  1       29/01/2022      22/04/2022      None
10134  1       03/05/2022      08/07/2022      Low
```

```
10334  2       19/11/2022      06/02/2023      Low
10220  2       28/07/2022      06/09/2022      None
] (delimiter is '\t');

// 6 Change records

JIRA_changes:
Outer Join (Changes)
Load
  [Ticket ID] AS [Change ID],
  [Source System]
inline
[
Ticket ID      Source System
10030  JIRA
10323  JIRA
10134  JIRA
10334  JIRA
10220  JIRA
10597  JIRA
] (delimiter is '\t');
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `Source System`
- `Change ID`
- `Business Impact`

Ora è possibile esaminare i 10 record risultanti.

Tabella dei risultati

Sistema di origine	Cambia ID	Impatto aziendale
JIRA	10030	Nessuno
JIRA	10134	Scarso
JIRA	10220	Nessuno
JIRA	10323	-
JIRA	10334	Scarso
JIRA	10597	-
-	10015	Scarso
-	10031	Scarso
-	10040	Nessuno
-	10138	Nessuno

Esempio 4 – Right join: Semplificazione di una tabella di destinazione da un set di dati master secondario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che rappresenta i record delle modifiche, che viene caricato in una tabella denominata `Changes`.
- Un secondo set di dati che rappresenta i record di modifica provenienti dal sistema di origine `Teamwork`. Questo viene caricato e combinato con i record originali unendolo con un prefisso di caricamento `Right Join`.

Ciò garantisce che vengano mantenuti solo i record delle modifiche `Teamwork`, senza perdere alcun record `Teamwork` se la tabella di destinazione non ha nessuna corrispondenza `Change ID`.

Script di caricamento

```
Changes:

Load * inline [
Change ID      Status ID       Scheduled Start Date    Scheduled End Date      Business Impact
10030  4       19/01/2022      23/02/2022      None
10015  3       04/01/2022      15/02/2022      Low
10103  1       02/04/2022      29/05/2022      Medium
10185  2       23/06/2022      08/09/2022      None
10323  1       08/11/2022      26/11/2022      High
10326  2       11/11/2022      05/12/2022      None
10138  2       07/05/2022      03/08/2022      None
10031  3       20/01/2022      25/03/2022      Low
10040  1       29/01/2022      22/04/2022      None
10134  1       03/05/2022      08/07/2022      Low
10334  2       19/11/2022      06/02/2023      Low
10220  2       28/07/2022      06/09/2022      None
10264  1       10/09/2022      17/10/2022      Medium
10116  1       15/04/2022      24/04/2022      None
10187  2       25/06/2022      24/08/2022      Low
] (delimiter is '\t');

Teamwork_changes:
Right Join (Changes)
Load
  [Ticket ID] AS [Change ID],
  [Source System]
inline
[
Ticket ID      Source System
10040  Teamwork
```

```
10015  Teamwork
10103  Teamwork
10031  Teamwork
50231  Teamwork
] (delimiter is '\t');
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `Source System`
- `Change ID`
- `Business Impact`

Ora è possibile esaminare i cinque record risultanti.

Tabella dei risultati

Sistema di origine	Cambia ID	Impatto aziendale
Lavoro in team	10015	Scarso
Lavoro in team	10031	Scarso
Lavoro in team	10040	Nessuno
Lavoro in team	10103	Medio
Lavoro in team	50231	-

Keep

Il prefisso **keep** è simile al prefisso **join**. Analogamente al prefisso **join**, confronta la tabella caricata con una tabella denominata esistente o con l'ultima tabella dati creata in precedenza, tuttavia, invece di unire la tabella caricata alla tabella esistente, riduce una o entrambe le due tabelle prima che vengano memorizzate in Qlik Sense, in base all'intersezione dei dati della tabella. Il confronto effettuato equivale a un'unione naturale effettuata su tutti i campi comuni, ad esempio nello stesso modo di un'unione corrispondente. In ogni modo, le due tabelle non vengono unite e verranno conservate in Qlik Sense come due tabelle denominate separatamente.

Sintassi:

```
(inner | left | right) keep [(tablename ) ]( loadstatement | selectstatement
)
```

Il prefisso **keep** deve essere preceduto da uno dei prefissi seguenti: **inner**,**left** o **right**.

Il prefisso esplicito **join** nel linguaggio di script di Qlik Sense consente di eseguire un'unione completa delle due tabelle. Il risultato è una sola tabella. In alcuni casi tale operazione di unione produce tabelle di dimensioni notevoli. Una delle principali funzioni di Qlik Sense è la capacità di generare associazioni tra più tabelle invece di unirle, un'operazione che riduce notevolmente l'utilizzo della memoria, aumenta le

prestazioni e offre una notevole flessibilità. In generale, si sconsiglia di utilizzare operazioni di unione esplicite negli script di Qlik Sense. La funzionalità keep è stata studiata proprio per ridurre le situazioni in cui occorre utilizzare operazioni di unione esplicite.

Argomenti:

Argomenti

Argomento	Descrizione
tablename	La tabella denominata da confrontare con la tabella caricata.
loadstatementoppure selectstatement	L'istruzione **LOAD** o **SELECT** per la tabella caricata.

Esempio:

```
Inner Keep LOAD * from abc.csv;
Left Keep SELECT * from table1;
tab1:
LOAD * from file1.csv;
tab2:
LOAD * from file2.csv;
.. .. ..
Left Keep (tab1) LOAD * from file3.csv;
```

Left

I prefissi **Join** e **Keep** possono essere preceduti dal prefisso **left**.

Se viene inserito prima di **join**, specifica che occorre utilizzare un'unione sinistra. La tabella risultante conterrà solo le combinazioni di valori di campo estratte dalle tabelle di dati non elaborati, dove i valori di campo di collegamento vengono rappresentati nella prima tabella. Se viene utilizzato prima di **keep**, specifica che la seconda tabella di dati non elaborati deve essere ridotta alla sua intersezione comune con la prima tabella prima di essere memorizzata in Qlik Sense.

Si stava cercando la funzione di stringa con lo stesso nome? Vedere: Left (page 1447)

Sintassi:

```
Left ( Join | Keep) [ (tablename) ](loadstatement | selectstatement)
```

Argomenti:

Argomenti

Argomento	Descrizione
tablename	La tabella denominata da confrontare con la tabella caricata.
loadstatementoppure selectstatement	L'istruzione **LOAD** o **SELECT** per la tabella caricata.

Esempio

Script di caricamento

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
Table1:
Load * inline [
Column1, Column2
A, B
1, aa
2, cc
3, ee ];

Table2:
Left Join Load * inline [
Column1, Column3
A, C
1, xx
4, yy ];
```

Risultato

Tabella risultante

Column1	Column2	Column3
A	B	C
1	aa	xx
2	cc	-
3	ee	-

Spiegazione

Questo esempio dimostra l'output Left Join dove vengono uniti solo i valori presenti nella prima tabella (sinistra).

Mapping

Il prefisso **mapping** consente di creare una tabella di mapping che può essere utilizzata, ad esempio, per sostituire i valori di campo e i nomi di campo durante l'esecuzione dello script.

Sintassi:

```
Mapping( loadstatement | selectstatement )
```

Il prefisso **mapping** può essere inserito prima di un'istruzione **LOAD** o **SELECT** e consente di memorizzare i risultati dell'istruzione di caricamento come tabella di mapping. Il mapping rappresenta un metodo efficiente per la sostituzione dei valori di campo durante l'esecuzione dello script, ad esempio la sostituzione di US, U.S. o America con USA. Una tabella di mapping è composta da due colonne: la prima

contiene dei valori di confronto, mentre la seconda contiene i valori di mapping desiderati. Le tabelle di mapping vengono salvate temporaneamente in memoria e vengono eliminate automaticamente una volta eseguito lo script.

È possibile accedere al contenuto della tabella di mapping utilizzando l'istruzione **Map … Using**, l'istruzione **Rename Field**, la funzione **Applymap()** o la funzione **Mapsubstring()**.

Esempio:

In questo esempio viene caricato un elenco del personale addetto alle vendite con un codice paese che ne identifica il paese di residenza. Viene utilizzata una tabella per il mapping di un codice paese a un paese al fine di per sostituire il codice paese con il relativo nome. Nella tabella di mapping vengono definiti solo tre paesi, mentre gli altri codici paese vengono mappati a 'Rest of the world'.

```
// Load mapping table of country codes:
map1:
mapping LOAD *
Inline [
CCode, Country
Sw, Sweden
Dk, Denmark
No, Norway
] ;
// Load list of salesmen, mapping country code to country
// If the country code is not in the mapping table, put Rest of the world
Salespersons:
LOAD *,
ApplyMap('map1', CCode,'Rest of the world') As Country
Inline [
CCode, Salesperson
Sw, John
Sw, Mary
Sw, Per
Dk, Preben
Dk, Olle
No, Ole
Sf, Risttu] ;
// We don't need the CCode anymore
Drop Field 'CCode';
```

La tabella risultante avrà l'aspetto seguente:

Mapping table

Salesperson	Country
John	Sweden
Mary	Sweden
Per	Sweden
Preben	Denmark
Olle	Denmark

Salesperson	Country
Ole	Norway
Risttu	Rest of the world

Unisci

Il prefisso **Merge** può essere aggiunto a qualsiasi istruzione **LOAD** o **SELECT** nello script per specificare che dovrebbe aggiungere record a un'altra tabella. Specifica anche che questa istruzione dovrebbe essere eseguita in un ricaricamento parziale.

Il caso tipico di utilizzo riguarda il caricamento di un registro dei cambiamenti che si desidera utilizzare per applicare inserts, updates e deletes a una tabella esistente.

Affinché il ricaricamento parziale funzioni correttamente, aprire l'app con i dati prima di attivare un ricaricamento parziale.

Eseguire un ricaricamento parziale usando il pulsante **Ricarica**. È anche possibile utilizzare Qlik Engine JSON API.

Sintassi:

```
Merge [only] [(SequenceNoField [, SequenceNoVar])] On ListOfKeys [Concatenate [(TableName)]] (loadstatement | selectstatement)
```

Argomenti:

Argomenti

Argomento	Descrizione
only	Un qualificatore opzionale che denota che l'istruzione dovrebbe essere eseguita solo durante i caricamenti parziali. L'istruzione viene ignorata durante i ricaricamenti normali (non parziali).
SequenceNoField	Il nome del campo contenente una data e ora o un numero di sequenza che definisce l'ordine delle operazioni.
SequenceNoVar	Il nome della variabile che viene assegnata al valore massimo per SequenceNoField della tabella unita.
ListOfKeys	Un elenco separato da virgole di nomi di campo che specifica la chiave primaria.
Operation	Il primo campo dell'istruzione LOAD deve contenere l'operazione come una stringa di testo: sono accettati anche 'Insert', 'Update' o 'Delete'. ‘i’, ‘u’ e ‘d’.

Funzionalità generale

Durante un caricamento normale (non parziale), la costruzione **Merge LOAD** funziona come una normale istruzione **Load** ma con la funzionalità aggiuntiva di rimuovere i record vecchi e obsoleti e i record contrassegnati per l'eliminazione. Il primo campo dell'istruzione **Load** deve presentare informazioni sull'operazione: Insert, Update o Delete.

Per ciascun record caricato, l'identificatore record verrà confrontato con i record precedentemente caricati e verrà conservato solo il record più recente (in base al numero di sequenza). Se il record più recente è contrassegnato con Delete, non ne verrà conservato nessuno.

Tabella di destinazione

Quale tabella modificare è determinato dal set di campi. Se una tabella con lo stesso set di campi (tranne il primo campo; l'operazione) esiste già, sarà questa la tabella da modificare. In alternativa, un prefisso **Concatenate** può essere utilizzato per specificare la tabella. Se la tabella di destinazione non viene determinata, il risultato della costruzione **Merge LOAD** viene archiviato in una nuova tabella.

Se viene utilizzato il prefisso Concatena, la tabella risultante ha un insieme di campi corrispondenti all'unione della tabella esistente e l'input per unire. Pertanto, la tabella di destinazione può avere più campi rispetto al registro delle modifiche utilizzato come input per l'unione.

Un ricaricamento parziale consente di ottenere lo stesso risultato di un ricaricamento completo. Una differenza è che un ricaricamento parziale raramente crea una nuova tabella. A meno che non sia stata utilizzata la clausola **Only**, esiste sempre una tabella di destinazione con lo stesso set di campi dell'esecuzione precedente dello script.

Numero di sequenza

Se il registro delle modifiche caricato è di tipo accumulato, ovvero contiene modifiche già caricate, il parametro SequenceNoVar può essere utilizzato in una clausola **Where** per limitare la quantità di dati di input. **Merge LOAD** può quindi essere realizzata per caricare soltanto i record in cui il campo SequenceNoField è superiore a SequenceNoVar. Al completamento, la costruzione **Merge LOAD** assegna un nuovo valore a SequenceNoVar con il valore massimo visto nel campo SequenceNoField.

Operazioni

Merge LOAD può avere meno campi rispetto alla tabella di destinazione. Le operazioni diverse trattano i campi mancati in modo diverso:

Insert: I campi mancanti in **Merge LOAD**, ma esistenti nella tabella di destinazione, ottengono un NULL nella tabella di destinazione.

Delete: i campi mancanti non influenzano il risultato. I record rilevanti vengono eliminati comunque.

Update: I campi elencati in **Merge LOAD** sono aggiornati nella tabella di destinazione. I campi mancanti non vengono modificati. Ciò significa che le due istruzioni seguenti non sono identiche:

- Merge on Key Concatenate Load 'U' as Operation, Key, F1, Null() as F2 From ...;
- Merge on Key Concatenate Load 'U' as Operation, Key, F1 From ...;

La prima istruzione aggiorna i record elencati e cambia F2 in NULL. La seconda non cambia F2, ma lascia i valori nella tabella di destinazione.

Esempi

Esempio 1: Unione semplice con tabella specificata

In questo esempio, una tabella inline denominata Persons è caricata con tre righe. **Merge** cambia quindi la tabella nel modo seguente:

- Aggiunge la riga, *Mary, 4*.
- Elimina la riga, *Steven, 3*.
- Assegna il numero *5* a *Jake*.

La variabile *LastChangeDate* è impostata al valore massimo nella colonna *ChangeDate* dopo l'esecuzione di **Merge**.

Script di caricamento

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
Set DateFormat='D/M/YYYY';
Persons:
load * inline [
Name, Number
Jake, 3
Jill, 2
Steven, 3
];

Merge (ChangeDate, LastChangeDate) on Name Concatenate(Persons)
LOAD * inline [
Operation, ChangeDate,   Name,     Number
Insert,    1/1/2021,     Mary,     4
Delete,    1/1/2021,     Steven,
Update,    2/1/2021,     Jake,     5
];
```

Risultato

Prima di **Merge Load**, la tabella risultante appare come segue:

Resulting table

Name	Number
Jake	3
Jill	2
Steven	3

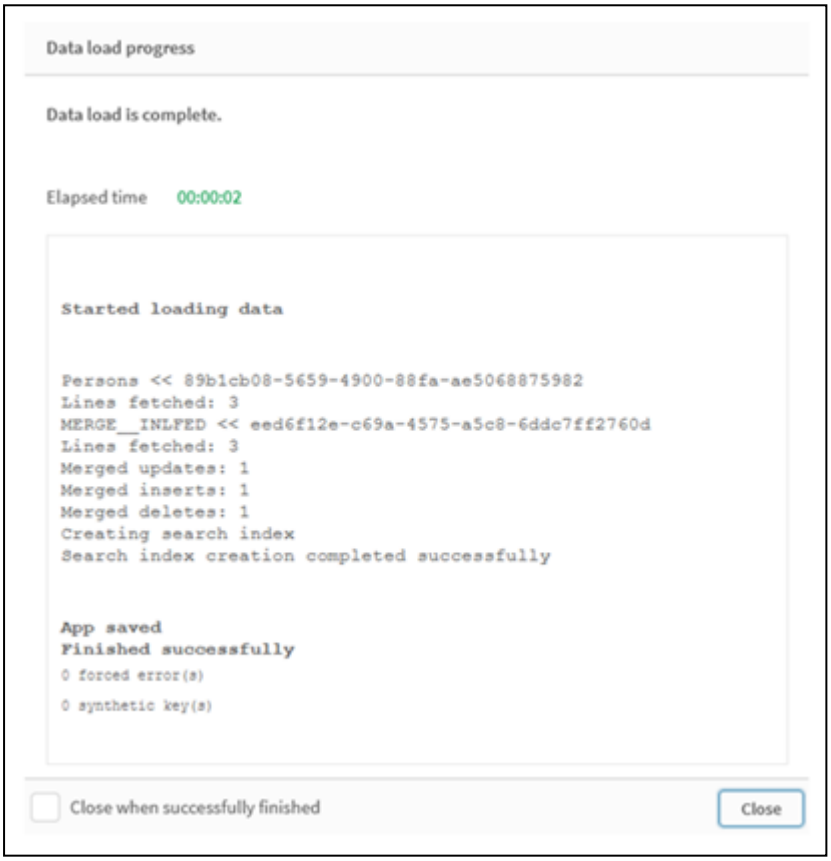
A seguito del comando **Merge Load**, la tabella appare come segue:

Resulting table

ChangeDate	Name	Number
2/1/2021	Jake	5
-	Jill	2
1/1/2021	Mary	4

Quando i dati vengono caricati, la finestra di dialogo **Avanzamento caricamento dati** mostra le operazioni che vengono eseguite:

Finestra di dialogo Avanzamento caricamento dati



Esempio 2: Script di caricamento dei dati con campi mancanti

In questo esempio, vengono caricati gli stessi dati di cui sopra, ma ora con un ID per ciascuna persona.

Merge cambia la tabella nel modo seguente:

- Aggiunge la riga, *Mary, 4*.
- Elimina la riga, *Steven, 3*.
- Assegna il numero *5* a *Jake*.
- Assegna il numero *6* a *Jill*.

Script di caricamento

Qui utilizziamo due istruzioni **Merge Load**, una per 'Inserisci' ed 'Elimina' e una per 'Aggiorna'.

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
Set DateFormat='D/M/YYYY';
Persons:
Load * Inline [
PersonID, Name, Number
1, Jake, 3
2, Jill, 2
3, Steven, 3
];

Merge (ChangeDate, LastChangeDate) on PersonID Concatenate(Persons)
Load * Inline [
Operation, ChangeDate,   PersonID, Name,     Number
Insert,    1/1/2021,     4,        Mary,     4
Delete,    1/1/2021,     3,        Steven,
];

Merge (ChangeDate, LastChangeDate) on PersonID Concatenate(Persons)
Load * Inline [
Operation, ChangeDate,   PersonID, Number
Update,    2/1/2021,     1,        5
Update,    3/1/2021,     2,        6
];
```

Risultato

Seguendo le istruzioni **Merge Load**, la tabella avrà il seguente aspetto:

Resulting table

PersonID	ChangeDate	Name	Number
1	2/1/2021	Jake	5
2	3/1/2021	Jill	6
4	1/1/2021	Mary	4

Notare che la seconda istruzione **Merge** non include il campo **Name**, e come conseguenza, i nomi non sono stati cambiati.

Esempio 3: Script di caricamento dei dati - Ricaricamento parziale usando un'istruzione Where con ChangeDate

Nell'esempio seguente, l'argomento **Only** specifica che il comando **Merge** viene eseguito solo durante un ricaricamento parziale. Gli aggiornamenti vengono filtrati in base al valore LastChangeDate precedentemente acquisito. Al termine dell'operazione **Merge**, alla variabile LastChangeDate è assegnato il valore massimo della colonna ChangeDate elaborato durante l'unione.

Script di caricamento

```
Merge Only (ChangeDate, LastChangeDate) on Name Concatenate(Persons)
LOAD Operation, ChangeDate,  Name, Number
from [lib://ChangeFilesFolder/BulkChangesInPersonsTable.csv] (txt)
where ChangeDate >= $(LastChangeDate);
```

NoConcatenate

Con il prefisso **NoConcatenate**, due tabelle che vengono caricate con gruppi di campo identici verranno considerate come due tabelle interne separate, invece di venire concatenate automaticamente.

Sintassi:

```
NoConcatenate( loadstatement | selectstatement )
```

Per impostazione predefinita, se viene caricata una tabella che contiene un numero identico di campi e nomi di campi corrispondenti a una tabella caricata in precedenza nello script, queste due tabelle verranno concatenate automaticamente da Qlik Sense. Questo accade anche se la seconda tabella ha un nome diverso.

Tuttavia, se il prefisso dello script `NoConcatenate` viene incluso prima dell'istruzione LOAD o SELECT della seconda tabella, le due tabelle verranno caricate separatamente.

Un caso d'uso tipico per `NoConcatenate` è quello in cui è necessario creare una copia temporanea di una tabella per eseguire alcune trasformazioni temporanee su tale copia, conservando una copia dei dati originali. `NoConcatenate` assicura che sia possibile creare tale copia senza aggiungerla implicitamente alla tabella di origine.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio di funzione

Esempio	Risultato
`Source: LOAD A,B from file1.csv; CopyOfSource: NoConcatenate LOAD A,B resident Source;`	Viene caricata una tabella con `A` e `B` come misure. Una seconda tabella con gli stessi campi viene caricata separatamente utilizzando la variabile `NoConcatenate`.

Esempio 1 - Concatenazione implicita

Script di caricamento e risultati

Panoramica

In questo esempio verranno aggiunti due script di caricamento in ordine sequenziale.

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati iniziale con date e importi che viene inviato a una tabella denominata `Transactions`.

Primo script di caricamento

```
Transactions:
LOAD
*
Inline [
id, date, amount
1, 08/30/2018, 23.56
2, 09/07/2018, 556.31
3, 09/16/2018, 5.75
4, 09/22/2018, 125.00
5, 09/22/2018, 484.21
6, 09/22/2018, 59.18
7, 09/23/2018, 177.42
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `amount`

Prima tabella dei risultati

id	date	importo
1	08/30/2018	23.56
2	09/07/2018	556.31
3	09/16/2018	5.75
4	09/22/2018	125.00
5	09/22/2018	484.21
6	09/22/2018	59.18
7	09/23/2018	177.42

Secondo script di caricamento

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un secondo set di dati con campi identici viene inviato a una tabella denominata `Sales`.

```
Sales:
LOAD
*
Inline [
id, date, amount
8, 10/01/2018, 164.27
9, 10/03/2018, 384.00
10, 10/06/2018, 25.82
11, 10/09/2018, 312.00
12, 10/15/2018, 4.56
13, 10/16/2018, 90.24
14, 10/18/2018, 19.32
];
```

Risultati

Caricare i dati e passare alla tabella.

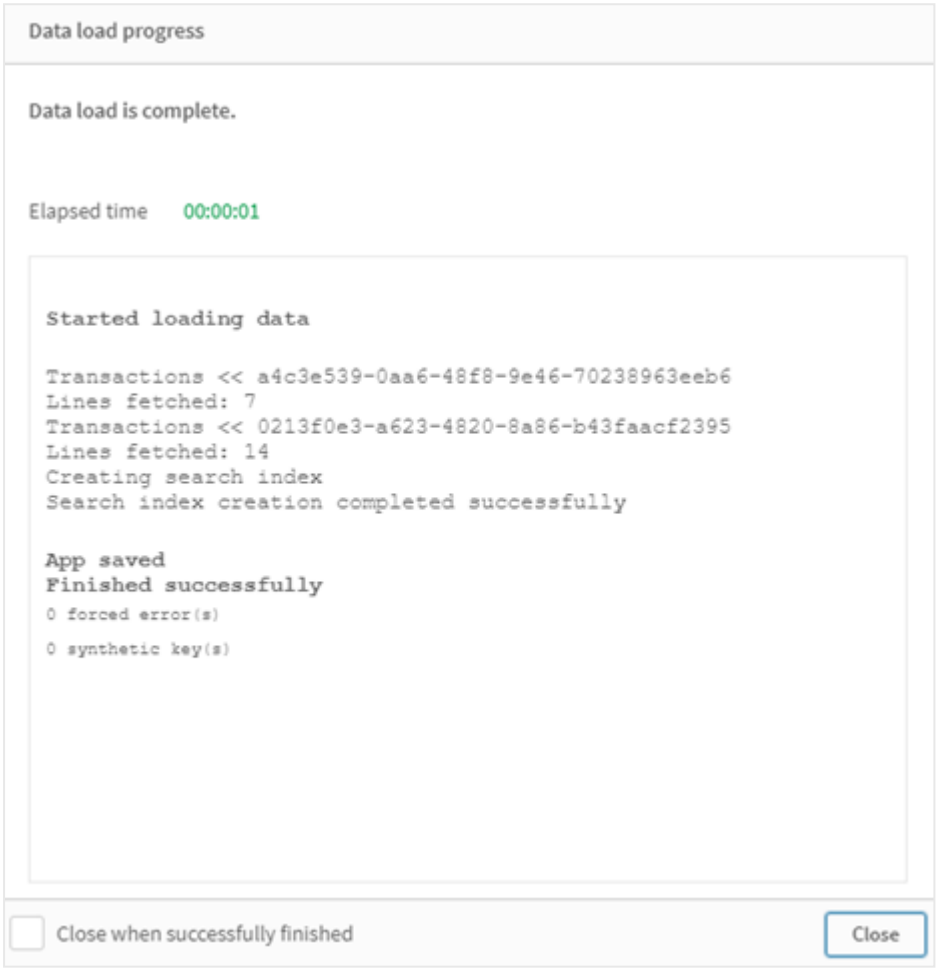
Seconda tabella dei risultati

id	date	importo
1	08/30/2018	23.56
2	09/07/2018	556.31
3	09/16/2018	5.75
4	09/22/2018	125.00
5	09/22/2018	484.21
6	09/22/2018	59.18
7	09/23/2018	177.42
8	10/01/2018	164.27
9	10/03/2018	384.00
10	10/06/2018	25.82
11	10/09/2018	312.00
12	10/15/2018	4.56
13	10/16/2018	90.24
14	10/18/2018	19.32

Quando viene eseguito lo script, la tabella `Sales` viene concatenata implicitamente alla tabella `Transactions` esistente, poiché i due set di dati condividono un numero identico di campi, con nomi di campo identici. Questo accade nonostante il secondo tag di nome della tabella tenti di dare un nome al set di risultati `'Sales'`.

Si può vedere che il set di dati Vendite è concatenato implicitamente osservando il registro di **Avanzamento del caricamento dei dati**.

Registro di avanzamento del caricamento dei dati che mostra la concatenazione implicita dei dati delle transazioni.



Esempio 2 - Scenario d'uso

Script di caricamento e risultati

Panoramica

In questo scenario d'uso si ha:

- Un set di dati sulle transazioni con:
    - id
    - date
    - importo (in GBP)
- Una tabella delle valute con:
    - Tassi di conversione da USD a GBP
- Un secondo set di dati sulle transazioni con:
    - id
    - date
    - importo (in USD)

Verranno caricati cinque script in ordine sequenziale.

- Il primo script di caricamento contiene un set di dati iniziale con date e importi in GBP che viene inviato a una tabella denominata `Transactions`.
- Il secondo script di caricamento contiene:
    - Un secondo set di dati con date e importi in USD viene inviato a una tabella denominata `Transactions_in_USD`.
    - Il prefisso `noconcatenate` che viene posto prima dell'istruzione LOAD del set di dati `Transactions_in_USD` per evitare la concatenazione implicita.
- Il terzo script di caricamento contiene il prefisso `join` che verrà utilizzato per creare un tasso di cambio tra GBP e USD nella tabella `Transactions_in_USD`.
- Il quarto script di caricamento contiene il prefisso `concatenate` che aggiungerà `Transactions_in_USD` alla tabella iniziale `Transactions`.
- Il quinto script di caricamento contiene l'istruzione `drop table` che rimuove la tabella `Transactions_in_USD` i cui dati sono stati concatenati alla tabella `Transactions`.

Primo script di caricamento

```
Transactions:

Load * Inline [
id, date, amount
1, 12/30/2018, 23.56
2, 12/07/2018, 556.31
3, 12/16/2018, 5.75
4, 12/22/2018, 125.00
5, 12/22/2018, 484.21
6, 12/22/2018, 59.18
7, 12/23/2018, 177.42
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- date
- amount

Risultati del primo script di caricamento

id	date	importo
1	12/30/2018	23.56
2	12/07/2018	556.31
3	12/16/2018	5.75
4	12/22/2018	125.00
5	12/22/2018	484.21
6	12/22/2018	59.18
7	12/23/2018	177.42

La tabella mostra il set di dati iniziale con importi in GBP.

Secondo script di caricamento

```
Transactions_in_USD:
NoConcatenate
Load * Inline [
id, date, amount
8, 01/01/2019, 164.27
9, 01/03/2019, 384.00
10, 01/06/2019, 25.82
11, 01/09/2019, 312.00
12, 01/15/2019, 4.56
13, 01/16/2019, 90.24
14, 01/18/2019, 19.32
];
```

Risultati

Caricare i dati e passare alla tabella.

Risultati del secondo script di caricamento

id	date	importo
1	12/30/2018	23.56
2	12/07/2018	556.31
3	12/16/2018	5.75
4	12/22/2018	125.00

id	date	importo
5	12/22/2018	484.21
6	12/22/2018	59.18
7	12/23/2018	177.42
8	01/01/2019	164.27
9	01/03/2019	384.00
10	01/06/2019	25.82
11	01/09/2019	312.00
12	01/15/2019	4.56
13	01/16/2019	90.24
14	01/18/2019	19.32

Si vedrà che il secondo set di dati della tabella `Transactions_in_USD` è stato aggiunto.

Terzo script di caricamento

Questo script di caricamento aggiunge alla tabella `Transactions_in_USD` un tasso di cambio da USD a GBP.

```
Join (Transactions_in_USD)
Load * Inline [
rate
0.7
];
```

Risultati

Caricare i dati e accedere al sistema di visualizzazione modello dati. Selezionare la tabella `Transactions_in_USD` per vedere che ogni record esistente ha un valore di 0,7 nel campo "tasso".

Quarto script di caricamento

Utilizzando il caricamento residente, questo script di caricamento concatenerà la tabella `Transactions_in_USD` alla tabella `Transactions` dopo aver convertito gli importi in USD.

```
Concatenate (Transactions)
LOAD
id,
date,
amount * rate as amount
Resident Transactions_in_USD;
```

Risultati

Caricare i dati e passare alla tabella. Dalle righe otto a quattordici vengono visualizzate nuove voci con importi in GBP.

Risultati del quarto script di caricamento

id	date	importo
1	12/30/2018	23.56
2	12/07/2018	556.31
3	12/16/2018	5.75
4	12/22/2018	125.00
5	12/22/2018	484.21
6	12/22/2018	59.18
7	12/23/2018	177.42
8	01/01/2019	114.989
8	01/01/2019	164.27
9	01/03/2019	268.80
9	01/03/2019	384.00
10	01/06/2019	18.074
10	01/06/2019	25.82
11	01/09/2019	218.40
11	01/09/2019	312.00
12	01/15/2019	3.192
12	01/15/2019	4.56
13	01/16/2019	63.168
13	01/16/2019	90.24
14	01/18/2019	13.524
14	01/18/2019	19.32

Quinto script di caricamento

Questo script di caricamento eliminerà le voci duplicate dalla tabella dei risultati del quarto script di caricamento, lasciando solo le voci con importi in GBP.

```
drop tables Transactions_in_USD;
```

Risultati

Caricare i dati e passare alla tabella.

Risultati del quinto script di caricamento

id	date	importo
1	12/30/2018	23.56
2	12/07/2018	556.31
3	12/16/2018	5.75
4	12/22/2018	125.00
5	12/22/2018	484.21
6	12/22/2018	59.18
7	12/23/2018	177.42
8	01/01/2019	114.989
9	01/03/2019	268.80
10	01/06/2019	18.074
11	01/09/2019	218.40
12	01/15/2019	3.192
13	01/16/2019	63.168
14	01/18/2019	13.524

Dopo aver caricato il quinto script di caricamento, la tabella dei risultati mostra tutte le quattordici transazioni esistenti in entrambi i set di dati di transazioni; tuttavia, le transazioni 8-14 sono state convertite in GBP.

Se si rimuove il prefisso `NoConcatenate` utilizzato prima di `Transactions_in_USD` nel secondo script di caricamento, lo script non riuscirà e mostrerà l'errore: "Tabella '`Transactions_in_USD`' non trovata". Questo perché la tabella `Transactions_in_USD` sarebbe stata concatenata automaticamente alla tabella originale `Transactions`.

Only

La parola chiave di script **Only** viene utilizzata come funzione di aggregazione o come parte della sintassi nei prefissi di ricaricamento parziale **Add**, **Replace** e **Merge**.

Outer

Il prefisso **Join** esplicito può essere preceduto dal prefisso **Outer** per specificare un'unione esterna. In un'unione esterna vengono generate tutte le combinazioni tra le due tabelle. La tabella risultante conterrà

quindi le combinazioni di valori di campo provenienti dalle tabelle di dati non elaborati dove i valori di campo di collegamento vengono rappresentati in una o entrambe le tabelle. La parola chiave **Outer** è facoltativa ed è il tipo di unione predefinito utilizzato quando non viene specificato un prefisso di unione.

Sintassi:

```
Outer Join [ (tablename) ](loadstatement |selectstatement )
```

Argomenti:

Argomenti

Argomento	Descrizione
tablename	La tabella denominata da confrontare con la tabella caricata.
loadstatementoppure selectstatement	L'istruzione **LOAD** o **SELECT** per la tabella caricata.

Esempio

Script di caricamento

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
Table1: Load * inline [ Column1, Column2 A, B 1, aa 2, cc 3, ee ];  Table2: Outer Join Load * inline [ Column1, Column3 A, C 1, xx 4, yy ];
```

Tabella risultante

Column1	Column2	Column3
A	B	C
1	aa	xx
2	cc	-
3	ee	-
4	-	yy

Spiegazione

In questo esempio, le due tabelle, Table1 e Table2, sono fuse in una singola tabella etichettata Table1. In casi come questo, il prefisso **outer** è spesso utilizzato per unire diverse tabelle in una singola tabella per eseguire aggregazioni sui valori di una singola tabella.

Caricamento parziale

Un ricaricamento completo inizia sempre eliminando tutte le tabelle nel modello dati esistente, dopodiché esegue lo script di caricamento.

Con un caricamento parziale questo non è possibile. Mantiene invece tutte le tabelle nel modello dati ed esegue solo le istruzioni **Load** e **Select** precedute da un prefisso **Aggiungi**, **Unisci** o **Sostituisci**. Altre

tabelle di dati non sono interessate dal comando. L'argomento **solo** indica che l'istruzione deve essere eseguita solo durante i caricamenti parziali e deve essere ignorata durante i caricamenti completi. La tabella seguente riepiloga l'esecuzione dell'istruzione per i ricaricamenti parziali e completi.

Istruzione	Ricaricamento completo	Caricamento parziale
Load ...	Verrà eseguita l'istruzione	Non verrà eseguita l'istruzione
Aggiungi/Sostituisci/Unisci LOAD...	Verrà eseguita l'istruzione	Verrà eseguita l'istruzione
Aggiungi/Sostituisci/Unisci solo LOAD...	Non verrà eseguita l'istruzione	Verrà eseguita l'istruzione

I ricaricamenti parziali comportano svariati benefici rispetto ai ricaricamenti completi:

- Sono più rapidi, perché solo i dati modificati di recente devono essere caricati. Con grandi set di dati la differenza è considerevole.
- Viene consumata meno memoria, visto che vengono caricati meno dati.
- Aumenta l'affidabilità, dato che le query ai dati sorgente vengono eseguite più rapidamente, riducendo il rischio di problemi di rete.

Affinché il ricaricamento parziale funzioni correttamente, aprire l'app con i dati prima di attivare un ricaricamento parziale.

Eseguire un ricaricamento parziale usando il pulsante **Ricarica**. È anche possibile utilizzare Qlik Engine JSON API.

Limitazioni

Un ricaricamento parziale non verrà completato correttamente se vi sono comandi con riferimento alle tabelle che sono esistite durante il ricaricamento completo ma non durante il ricaricamento parziale.

Esempio

Comandi di esempio

```
LEFT JOIN(<Table_removed_after_full_reload>)
CONCATENATE(<Table_removed_after_full_reload>)
```

Dove <Table_removed_after_full_reload> è una tabella che esisteva a table nel ricaricamento completo, ma che non è presente in quello parziale.

Soluzione

Come soluzione il comando può essere racchiuso dalla seguente istruzione if:

```
IF NOT IsPartialReload() THEN ... ENDIF.
```

Un ricaricamento parziale può rimuovere valori dai dati. Tuttavia, ciò non verrà riflesso nell'elenco dei valori distinti, che rappresenta una tabella mantenuta internamente. Pertanto, dopo un ricaricamento parziale, l'elenco conterrà tutti i valori distinti esistiti nel campo dall'ultimo ricaricamento completo, che potrebbero essere superiori a quanto attualmente esistente dopo il ricaricamento parziale. Ciò influisce sull'output delle funzioni FieldValueCount() e FieldValue(). Il valore FieldValueCount() potrebbe potenzialmente restituire un numero superiore al numero corrente di valori di campo.

Esempio

Esempio 1

Script di caricamento

Aggiungere lo script di esempio alla propria app ed eseguire un ricaricamento parziale. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
T1:
Add only Load distinct recno()+10 as Num autogenerate 10;
```

Risultato

Resulting table

Num	Count(Num)
11	1
12	1
13	1
14	1
15	1
16	1
17	1
18	1
19	1
20	1

Spiegazione

L'istruzione viene eseguita solo durante un ricaricamento parziale. Se il prefisso "distinto" viene omesso, il conteggio del campo **Num** aumenterà con ciascun ricaricamento parziale successivo.

Esempio 2

Script di caricamento

Aggiungere lo script di esempio alla propria app. Eseguire un ricaricamento completo e visualizzare il risultato. Quindi, eseguire un ricaricamento parziale e visualizzare il risultato. Per visualizzare i risultati, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
T1:
Load recno() as ID, recno() as Value autogenerate 10;

T1:
Replace only Load recno() as ID, repeat(recno(),3) as Value autogenerate 10;
```

Risultato

Output table after full reload

ID	Value
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10

Output table after partial reload

ID	Value
1	111
2	222
3	333
4	444
5	555
6	666
7	777
8	888
9	999
10	101010

Spiegazione

La prima tabella viene caricata durante un ricaricamento completo, mentre la seconda tabella sostituisce semplicemente la prima tabella durante un ricaricamento parziale.

Replace

La parola chiave di script **Replace** viene utilizzata come funzione stringa o come prefisso nel ricaricamento parziale.

Replace

Il prefisso **Replace** può essere aggiunto a qualsiasi istruzione **LOAD** o **SELECT** nello script per specificare che la tabella caricata dovrebbe sostituire un'altra tabella. Specifica anche che questa istruzione dovrebbe essere eseguita in un ricaricamento parziale. Il prefisso **Replace** può essere usato anche in un'istruzione **Map**.

Affinché il ricaricamento parziale funzioni correttamente, aprire l'app con i dati prima di attivare un ricaricamento parziale.

Eseguire un ricaricamento parziale usando il pulsante **Ricarica**. È anche possibile utilizzare Qlik Engine JSON API.

Sintassi:

```
Replace [only] [Concatenate[(tablename)]] (loadstatement | selectstatement)
```

```
Replace [only] mapstatement
```

Durante un caricamento normale (non parziale), la costruzione **Replace LOAD** funzionerà come una normale istruzione **LOAD** ma sarà preceduta da un **Drop Table**. Verrà prima eliminata la vecchia tabella, quindi i record verranno generati e archiviati come una nuova tabella.

Se viene utilizzato il prefisso **Concatenate**, o se esiste una tabella con lo stesso set di campi, sarà questa la tabella pertinente da eliminare. Altrimenti, non vi sarà alcuna tabella da eliminare e la costruzione **Replace LOAD** sarà identica a un normale **LOAD**.

Un caricamento parziale otterrà lo stesso risultato. L'unica differenza è che c'è sempre una tabella da eliminare dalla precedente esecuzione script. La costruzione **Replace LOAD** eliminerà sempre prima la vecchia tabella, per poi creare la nuova.

L'istruzione **Replace Map...Using** determina l'esecuzione del mapping anche durante l'esecuzione parziale dello script.

Argomenti:

Argomenti

Argomento	Descrizione
only	Un qualificatore opzionale che denota che l'istruzione dovrebbe essere eseguita solo durante i caricamenti parziali. Deve essere ignorata durante i caricamenti normali (non parziali).

Esempi e risultati:

Esempio	Risultato
`Tab1: Replace LOAD * from File1.csv;`	Durante le operazioni di ricaricamento normale e parziale dei dati, la tabella Tab1 di Qlik Sense viene inizialmente rimossa. Successivamente, i nuovi dati vengono caricati dal file File1.csv e memorizzati nella tabella Tab1.
`Tab1: Replace only LOAD * from File1.csv;`	Durante le operazioni di ricaricamento normale, questa istruzione viene ignorata. Durante il ricaricamento parziale, ogni tabella di Qlik Sense precedentemente denominata Tab1 viene inizialmente rimossa. Successivamente, i nuovi dati vengono caricati dal file File1.csv e memorizzati nella tabella Tab1.
`Tab1: LOAD a,b,c from File1.csv; Replace LOAD a,b,c from File2.csv;`	Durante il ricaricamento normale il file File1.csv viene inizialmente letto nella tabella di Qlik Sense Tab1, ma successivamente viene immediatamente rimosso e sostituito dai nuovi dati caricati dal file File2.csv. Tutti i dati del file File1.csv vengono persi. Durante il caricamento parziale l'intera tabella di Qlik Sense Tab1 viene inizialmente rimossa. Successivamente, la tabella viene sostituita dai dati caricati dal file File2.csv.
`Tab1: LOAD a,b,c from File1.csv; Replace only LOAD a,b,c from File2.csv;`	Durante il ricaricamento normale, i dati vengono caricati dal file File1.csv e memorizzati nella tabella di Qlik Sense Tab1. Il file File2.csv viene ignorato. Durante il ricaricamento parziale l'intera tabella di Qlik Sense Tab1 viene inizialmente rimossa. Successivamente, la tabella viene sostituita dai dati caricati dal file File2.csv. Tutti i dati del file File1.csv vengono persi.

Right

I prefissi **Join** e **Keep** possono essere preceduti dal prefisso **right**.

Se viene inserito prima di **join**, specifica che occorre utilizzare un'unione destra. La tabella risultante contiene solo le combinazioni di valori di campo estratte dalle tabelle di dati non elaborati, dove i valori di campo di collegamento vengono rappresentati nella seconda tabella. Se viene utilizzato prima di **keep**, specifica che la prima tabella di dati non elaborati deve essere ridotta alla sua intersezione comune con la seconda tabella prima di essere memorizzata in Qlik Sense.

Si stava cercando la funzione di stringa con lo stesso nome? Vedere: Right (page 1456)

Sintassi:

```
Right (Join | Keep) [(tablename)](loadstatement |selectstatement )
```

Argomenti:

Argomenti

Argomento	Descrizione
tablename	La tabella denominata da confrontare con la tabella caricata.
loadstatementoppure selectstatement	L'istruzione **LOAD** o **SELECT** per la tabella caricata.

Esempio

Script di caricamento

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
Table1:
Load * inline [
Column1, Column2
A, B
1, aa
2, cc
3, ee ];

Table2:
Right Join Load * inline [
Column1, Column3
A, C
1, xx
4, yy ];
```

Risultato

Tabella risultante

Column1	Column2	Column3
A	B	C
1	aa	xx
4	-	yy

Spiegazione

Questo esempio dimostra l'output di Right Join dove vengono uniti solo i valori presenti nella seconda tabella (destra).

Sample

Il prefisso **sample** aggiunto a un'istruzione **LOAD** o **SELECT** viene utilizzato per caricare un campione casuale di record dalla sorgente dati.

Sintassi:

```
Sample  p ( loadstatement | selectstatement )
```

L'espressione che viene valutata non definisce la percentuale di record del set di dati che verranno caricati nell'applicazione Qlik Sense, ma la probabilità che ogni record letto venga caricato nell'applicazione. In altre parole, specificare un valore `p = 0.5` non significa che verrà caricato il 50% del numero totale di record, ma piuttosto che per ogni record ci sarà il 50% di possibilità che venga caricato nell'applicazione Qlik Sense.

Argomenti

Argomento	Descrizione
p	Un'espressione arbitraria che restituisce un numero maggiore di 0 e minore o uguale a 1. Il numero indica la probabilità di lettura di un determinato record. Anche se verranno letti tutti i record, solo alcuni verranno caricati in Qlik Sense.

Casi di utilizzo

Un campione è utile quando si desidera campionare i dati provenienti da una tabella di grandi dimensioni, per capire la natura dei dati, la distribuzione o il contenuto dei campi. Dato che porta un sottoinsieme di dati, il caricamento dei dati è più veloce e consente di testare più rapidamente gli script. A differenza di `First`, la funzione `Sample` riporta i dati dell'intera tabella, invece di limitarsi alle prime righe. In alcuni casi, ciò può fornire una rappresentazione più accurata dei dati.

Gli esempi seguenti mostrano due possibili usi del prefisso script `Sample`:

```
Sample 0.15 SQL SELECT * from Longtable;
Sample(0.15) LOAD * from Longtab.csv;
```

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Campione da una tabella in linea

Script di caricamento e risultati

Panoramica

In questo esempio, lo script carica un campione di dati da un set di dati contenente sette record in una tabella denominata `Transactions` da una tabella in linea.

Script di caricamento

```
Transactions:
SAMPLE 0.3
LOAD
*
Inline [
id, date, amount
1, 08/30/2018, 23.56
2, 09/07/2018, 556.31
3, 09/16/2018, 5.75
4, 09/22/2018, 125.00
5, 09/22/2018, 484.21
6, 09/22/2018, 59.18
7, 09/23/2018, 177.42
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `amount`

Aggiungere la seguente misura:

```
=sum(amount)8
```

Tabella dei risultati

id	date	=Sum(amount)
2	09/07/2018	556.31
4	09/22/2018	125
1	08/30/2018	23.56
3	09/16/2018	5.75

Nell'iterazione del caricamento utilizzata in questo esempio, sono stati letti tutti e sette i record, ma solo quattro sono stati caricati nella tabella dei dati. Qualsiasi nuova esecuzione di caricamento potrebbe risultare in un numero diverso e in un diverso set di record caricati nell'applicazione.

Esempio 2 - Esempio di tabella generata automaticamente

Script di caricamento e risultati

Panoramica

In questo esempio, utilizzando `Autogenerate`, viene creato un set di dati di 100 record con i campi `date`, `id`, e `amount`. Tuttavia, viene utilizzato il prefisso `Sample`, con un valore di 0,1.

Script di caricamento

```
SampleData:
Sample 0.1
LOAD
RecNo() AS id,
MakeDate(2013, Ceil(Rand() * 12), Ceil(Rand() * 29)) as date,
Rand() * 1000 AS amount

Autogenerate(100);
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `amount`

Aggiungere la seguente misura:

Tabella dei risultati

id	date	=Sum(amount)
48	9/28/2013	763
20	5/15/2013	752
19	11/8/2013	657
25	3/24/2013	522
27	8/23/2013	389
81	6/1/2013	53
100	8/15/2013	17

Nell'iterazione del caricamento utilizzata in questo esempio, sono stati caricati sette record dal set di dati creato. Di nuovo, qualsiasi nuova esecuzione di caricamento potrebbe risultare in un numero diverso e in un diverso set di record caricati nell'applicazione.

Semantic

Il prefisso dell'istruzione LOAD `semantic` crea un tipo speciale di campo che può essere utilizzato in Qlik Sense per connettere e gestire dati relazionali, come strutture ad albero, dati strutturati genitore-figlio autoreferenziali e/o dati che possono essere descritti come un grafico.

Si noti che il carico `semantic` può funzionare in modo simile ai prefissi *Hierarchy (page 64)* e *HierarchyBelongsTo (page 66)*. Tutti e tre i prefissi possono essere utilizzati come elementi costitutivi in efficaci soluzioni front-end per l'attraversamento dei dati relazionali.

Sintassi:

```
Semantic( loadstatement | selectstatement)
```

Un carico semantico prevede un input largo esattamente tre o quattro campi con una definizione rigorosa di ciò che rappresenta ciascun campo ordinato, come mostrato nella tabella seguente:

Campi di carico semantico

Nome campo	Descrizione del campo
1° campo:	questo tag è una rappresentazione del primo di due oggetti tra i quali esiste una relazione.
2° campo:	questo tag verrà utilizzato per descrivere la relazione "forward" tra il primo e il secondo oggetto. Se il primo è un oggetto figlio e il secondo è un oggetto padre, è possibile creare una scheda di relazione che indichi "padre" o "padre di", per fare in modo di seguire dall'oggetto figlio all'oggetto padre.
3° campo:	questo tag è una rappresentazione del primo di due oggetti tra i quali esiste una relazione.
4° campo:	questo campo è facoltativo. Questo tag descrive le relazioni "forward" o "inverse" tra il primo e il secondo oggetto. Se il primo è un oggetto figlio e il secondo è un oggetto padre, una scheda di relazione può indicare "figlio" o "figlio di", per fare in modo di seguire dall'oggetto padre all'oggetto figlio. Se non si aggiunge un quarto campo, il secondo tag di campo verrà utilizzato per descrivere la relazione in entrambe le direzioni. In tal caso, viene aggiunto automaticamente un simbolo di freccia come parte del tag.

Il codice seguente è un esempio del prefisso `semantic`.

```
Semantic
Load
Object,
‘Parent’ AS Relationship,
NeighbouringObject AS Object,
‘Child’ AS Relationship
from graphdata.csv;
```

È consentito ed è una pratica comune etichettare il terzo campo come il primo campo. Questo crea una ricerca autoreferenziale, in modo che si possa seguire gli oggetti verso gli oggetti correlati una fase alla volta. Se il 3° campo non ha lo stesso nome, il risultato finale sarà una semplice ricerca da uno o più oggetti ai suoi vicini relazionali diretti, con un output di scarsa utilità pratica.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Funzioni correlate

Funzioni	Interazione
Hierarchy (page 64)	Il prefisso di caricamento Hierarchy viene utilizzato per dividere e organizzare i nodi in strutture di dati padre-figlio e altre strutture di dati simili a grafici e trasformarli in tabelle.
HierarchyBelongsTo (page 66)	Il prefisso di caricamento HierarchyBelongsTo viene utilizzato per individuare e organizzare i dati predecessori dei dati padre-figlio e altre strutture di dati simili a grafici e trasformarli in tabelle.

Esempio: creazione di un campo speciale per collegare le relazioni utilizzando il prefisso semantico

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che rappresenta i record della relazione geografica, che viene caricato in una tabella denominata `GeographyTree`.

    - Ogni voce ha un ID all'inizio della riga e un ParentID alla fine della riga.
- Il prefisso `semantic` aggiungerà un campo di comportamento speciale con l'etichetta `Relation`.

Script di caricamento

```
GeographyTree:
LOAD
    ID,
    Geography,
    if(ParentID='',null(),ParentID) AS ParentID

INLINE [
ID,Geography,ParentID
1,World
2,Europe,1
3,Asia,1
4,North America,1
5,South America,1
6,UK,2
7,Germany,2
8,Sweden,2
9,South Korea,3
10,North Korea,3
11,China,3
12,London,6
13,Birmingham,6
];

SemanticTable:
Semantic Load
       ID as ID,
       'Parent' as Relation,
       ParentID as ID,
       'Child' as Relation
resident GeographyTree;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `Id`
- `Geography`

Quindi, creare una casella di filtro con `Relation` come dimensione. Fare clic su **Termina modifica**.

Tabella dei risultati

Id	Geografia
1	Mondo
2	Europa

Id	Geografia
3	Asia
4	Nord America
5	America Latina
6	GB
7	Germany
8	Svezia
9	Corea del Sud
10	Corea del Nord
11	Cina
12	Londra
13	Birmingham

Casella di filtro

Relazione
Figlio
Padre

Fare clic su **Europa** dalla dimensione `Geography` nella tabella e fare clic su **Figlio** dalla dimensione `Relation` nella casella di filtro. Notare il risultato atteso nella tabella:

Tabella dei risultati che mostra i risultati "figli" dell'Europa

Id	Geografia
6	GB
7	Germany
8	Svezia

Facendo nuovamente clic su **Figlio** , verranno mostrati i luoghi "figlio" del Regno Unito, un gradino più in basso.

Tabella dei risultati che mostra i risultati "figli" del Regno Unito

Id	Geografia
12	Londra
13	Birmingham

Unless

Il prefisso e suffisso **unless** viene utilizzato per creare una clausola condizionale che determina se valutare o meno un'istruzione oppure una clausola exit. Può essere considerato come un'alternativa compatta all'istruzione completa **if..end if**.

Sintassi:

```
(Unless condition statement | exitstatement Unless condition )
```

L'istruzione **statement** o **exitstatement** verrà eseguita solo se **condition** restituisce False.

Il prefisso **unless** può essere utilizzato con istruzioni che presentano già almeno un'altra istruzione, inclusi i prefissi addizionali **when** o **unless**.

Argomenti

Argomento	Descrizione
condition	Un'espressione logica che restituisce un valore True o False.
statement	Qualsiasi istruzione dello script di Qlik Sense, ad eccezione delle istruzioni di controllo.
exitstatement	Una clausola **exit for**, **exit do** o **exit sub** oppure un'istruzione **exit script**.

Casi di utilizzo

L'istruzione `Unless` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzata come condizione quando l'utente desidera caricare o escludere parti dello script in modo condizionato.

Le righe seguenti mostrano tre esempi di utilizzo della funzione `Unless`:

```
exit script unless A=1;
unless A=1 LOAD * from myfile.csv;
unless A=1 when B=2 drop table Tab1;
```

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Prefisso Unless

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- La creazione della variabile A, alla quale viene assegnato il valore 1.
- Un set di dati che viene caricato in una tabella denominata Transazioni, a meno che la variabile A = 2.

Script di caricamento

```
LET A = 1;

UNLESS A = 2

Transactions:
LOAD
*
Inline [
id, date, amount
1, 08/30/2018, 23.56
2, 09/07/2018, 556.31
3, 09/16/2018, 5.75
4, 09/22/2018, 125.00
5, 09/22/2018, 484.21
6, 09/22/2018, 59.18
7, 09/23/2018, 177.42
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `amount`

Tabella dei risultati

id	date	importo
1	08/30/2018	23.56
2	09/07/2018	556.31
3	09/16/2018	5.75

id	date	importo
4	09/22/2018	125.00
5	09/22/2018	484.21
6	09/22/2018	59.18
7	09/23/2018	177.42

Poiché alla variabile A viene assegnato il valore 1 all'inizio dello script, la condizione che segue il prefisso `Unless` viene valutata, restituendo un risultato di `FALSE`. Di conseguenza, lo script continua a eseguire l'istruzione `Load`. Nella tabella dei risultati sono visibili tutti i record della tabella `Transactions`.

Se il valore di questa variabile è impostato su 2, non verranno caricati dati nel modello dati.

Esempio 2 – Suffisso Unless

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento dati inizia caricando un set di dati iniziale in una tabella denominata `Transactions`. Lo script viene quindi terminato se non vi sono meno di 10 record nella tabella `Transactions`.

Se questa condizione non porta alla chiusura dello script, un altro set di transazioni viene concatenato nella tabella `Transactions` e il processo viene ripetuto.

Script di caricamento

```
Transactions:
LOAD
*
Inline [
id, date, amount
1, 08/30/2018, 23.56
2, 09/07/2018, 556.31
3, 09/16/2018, 5.75
4, 09/22/2018, 125.00
5, 09/22/2018, 484.21
6, 09/22/2018, 59.18
7, 09/23/2018, 177.42
];


exit script unless NoOfRows('Transactions') < 10 ;

Concatenate
LOAD
*
Inline [
```

```
id, date, amount
8, 10/01/2018, 164.27
9, 10/03/2018, 384.00
10, 10/06/2018, 25.82
11, 10/09/2018, 312.00
12, 10/15/2018, 4.56
13, 10/16/2018, 90.24
14, 10/18/2018, 19.32
];

exit script unless NoOfRows('Transactions') < 10 ;

Concatenate
LOAD
*
Inline [
id, date, amount
15, 10/01/2018, 164.27
16, 10/03/2018, 384.00
17, 10/06/2018, 25.82
18, 10/09/2018, 312.00
19, 10/15/2018, 4.56
20, 10/16/2018, 90.24
21, 10/18/2018, 19.32
];

exit script unless NoOfRows('Transactions') < 10 ;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- date
- amount

Tabella dei risultati

id	date	importo
1	08/30/2018	23.56
2	09/07/2018	556.31
3	09/16/2018	5.75
4	09/22/2018	125.00
5	09/22/2018	484.21
6	09/22/2018	59.18
7	09/23/2018	177.42
8	10/01/2018	164.27

id	date	importo
9	10/03/2018	384.00
10	10/06/2018	25.82
11	10/09/2018	312.00
12	10/15/2018	4.56
13	10/16/2018	90.24
14	10/18/2018	19.32

Sono presenti sette record in ciascuno dei tre set di dati dello script di caricamento.

Il primo set di dati (con le transazioni da `id` 1 a 7) viene caricato nell'applicazione. La condizione `Unless` valuta se nella tabella `Transactions` ci sono meno di 10 righe. La valutazione è pari a `TRUE`, quindi il secondo set di dati (con le transazioni `id` da 8 a 14) viene caricato nell'applicazione. La seconda condizione `Unless` viene valutata se nella tabella `Transactions` sono presenti meno di 10 record. La valutazione è pari a `FALSE`, quindi lo script termina.

Esempio 3 – Prefissi Unless multipli

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

In questo esempio, un set di dati contenente una transazione viene creato come tabella chiamata `Transactions`. Viene quindi attivato un ciclo "for", in cui vengono valutate due istruzioni "unless" nidificate:

1. A meno che non ci siano più di 100 record nella tabella `Transactions`
2. A meno che il numero di record nella tabella `Transactions` non sia un multiplo di 6

Se queste condizioni sono `FALSE`, vengono generati altri sette record e concatenati alla tabella `Transactions` esistente. Questo processo viene ripetuto finché una delle due transazioni non restituisce un valore di `TRUE`.

Script di caricamento

```
Transactions:
Load
        0 as id
Autogenerate 1;

For i = 1 to 100
        unless NoOfRows('Transactions') > 100 unless mod(NoOfRows('Transactions'),6) = 0
                Concatenate
      Load
if(isnull(Peek(id)),1,peek(id)+1) as id
```

```
                Autogenerate 7;
next i
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`id`.

Tabella dei risultati

id
0
1
2
3
4
5
+30 righe in più

Le istruzioni nidificate unless che si verificano nel ciclo 'for' valutano quanto segue:

1. Sono presenti più di 100 righe nella tabella `Transactions`?
2. Il numero totale di record nella tabella `Transactions` è un multiplo di 6?

Ogni volta che entrambe le istruzioni unless restituiscono un valore di `FALSE`, vengono generati altri sette record e concatenati alla tabella `Transactions` esistente.

Queste istruzioni restituiscono un valore di `FALSE` cinque volte, a questo punto è presente un totale di 36 righe di dati nella tabella `Transactions`.

Dopodiché, la seconda istruzione `Unless` restituisce un valore di `TRUE`, pertanto l'istruzione di caricamento successiva non verrà più eseguita.

When

Il prefisso e suffisso **when** viene utilizzato per creare una clausola condizionale che determina se eseguire o meno un'istruzione oppure una clausola exit. Può essere considerato come un'alternativa compatta all'istruzione completa **if..end if**.

Sintassi:

```
(when condition statement | exitstatement when condition )
```

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

L'istruzione **statement** o **exitstatement** verrà eseguita solo se la condizione restituisce TRUE.

Il prefisso `When` può essere utilizzato con istruzioni che presentano già almeno un'altra istruzione, inclusi i prefissi addizionali `When` o `Unless`.

Casi di utilizzo

L'istruzione `When` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzata come condizione quando l'utente desidera caricare o escludere parti di uno script.

Argomenti

Argomento	Descrizione
condition	Un'espressione logica che restituisce un valore TRUE o FALSE
statement	Qualsiasi istruzione dello script di Qlik Sense, ad eccezione delle istruzioni di controllo.
exitstatement	Una clausola **exit for**, **exit do** o **exit sub** oppure un'istruzione **exit script**.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`exit script when A=1;`	Quando l'istruzione `A=1` viene valutata come TRUE, lo script si interrompe.
`when A=1 LOAD * from myfile.csv;`	Quando l'istruzione `A=1` viene valutata come TRUE, viene caricato `myfile.csv`.
`when A=1 unless B=2 drop table Tab1;`	Quando l'istruzione `A=1` viene valutata come TRUE e se `B=2` viene valutato come FALSE, la tabella `Tab1` viene eliminata.

Esempio 1 – Prefisso When

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati con date e importi che viene inviato a una tabella denominata '`Transactions`'.
- L'istruzione `Let` che afferma che la variabile `A` è stata creata e ha il valore di `1`.
- La condizione `When` che fornisce la condizione che se `A` è uguale a `1`, allora lo script continuerà a essere caricato.

Script di caricamento

```
LET A = 1;

WHEN A = 1

Transactions:
LOAD
*
Inline [
id, date, amount
1, 08/30/2018, 23.56
2, 09/07/2018, 556.31
3, 09/16/2018, 5.75
4, 09/22/2018, 125.00
5, 09/22/2018, 484.21
6, 09/22/2018, 59.18
7, 09/23/2018, 177.42
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `amount`

Tabella dei risultati

id	date	importo
1	08/30/2018	23.56
2	09/07/2018	556.31

id	date	importo
3	09/16/2018	5.75
4	09/22/2018	125.00
5	09/22/2018	484.21
6	09/22/2018	59.18
7	09/23/2018	177.42

Poiché alla variabile `A` viene assegnato il valore di `1` all'inizio dello script, la condizione che segue il prefisso `When` viene valutata e restituisce un risultato di TRUE. Poiché il risultato è TRUE, lo script continua a eseguire l'istruzione LOAD. È possibile visualizzare tutti i record della tabella dei risultati.

Se il valore di questa variabile fosse impostato su un valore diverso da `1`, non verrebbero caricati dati nel modello dati.

Esempio 2 – Suffisso When

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Tre set di dati con date e importi che vengono inviati a una tabella denominata '`Transactions`'.
  - Il primo set di dati contiene le transazioni 1-7.
  - Il secondo set di dati contiene le transazioni 8-14.
  - Il terzo set di dati contiene le transazioni 15-21.
- Una condizione `When` che determina se la tabella '`Transactions`' contiene più di dieci righe. Se una qualsiasi delle istruzioni `When` viene valutata come TRUE, lo script di caricamento si interrompe. Questa condizione è posta alla fine di ciascuno dei tre set di dati.

Script di caricamento

```
Transactions:
LOAD
*
Inline [
id, date, amount
1, 08/30/2018, 23.56
2, 09/07/2018, 556.31
3, 09/16/2018, 5.75
4, 09/22/2018, 125.00
5, 09/22/2018, 484.21
6, 09/22/2018, 59.18
7, 09/23/2018, 177.42
];
```

```
exit script when NoOfRows('Transactions') > 10 ;

Concatenate
LOAD
*
Inline [
id, date, amount
8, 10/01/2018, 164.27
9, 10/03/2018, 384.00
10, 10/06/2018, 25.82
11, 10/09/2018, 312.00
12, 10/15/2018, 4.56
13, 10/16/2018, 90.24
14, 10/18/2018, 19.32
];

exit script when NoOfRows('Transactions') > 10 ;

Concatenate
LOAD
*
Inline [
id, date, amount
15, 10/01/2018, 164.27
16, 10/03/2018, 384.00
17, 10/06/2018, 25.82
18, 10/09/2018, 312.00
19, 10/15/2018, 4.56
20, 10/16/2018, 90.24
21, 10/18/2018, 19.32
];

exit script when NoOfRows('Transactions') > 10 ;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- date
- amount

Tabella dei risultati

id	date	importo
1	08/30/2018	23.56
2	09/07/2018	556.31
3	09/16/2018	5.75

id	date	importo
4	09/22/2018	125.00
5	09/22/2018	484.21
6	09/22/2018	59.18
7	09/23/2018	177.42
8	10/01/2018	164.27
9	10/03/2018	384.00
10	10/06/2018	25.82
11	10/09/2018	312.00
12	10/15/2018	4.56
13	10/16/2018	90.24
14	10/18/2018	19.32

Ciascuna delle tre serie di dati contiene sette transazioni. Il primo set di dati contiene le transazioni 1-7 e viene caricato nell'applicazione. La condizione `When` che segue l'istruzione di caricamento viene valutata come FALSE perché nella tabella '`Transactions`' sono presenti meno di dieci righe. Lo script di caricamento continua con il set di dati successivo.

Il secondo set di dati contiene le transazioni 8-14 e viene caricato nell'applicazione. La seconda condizione `When` viene valutata come TRUE perché ci sono più di dieci righe nella tabella '`Transactions`'. Pertanto, lo script termina.

Esempio 3 - Prefissi When multipli

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente una singola transazione viene creato come tabella chiamata '`Transactions`'.
- Un ciclo `For` che viene attivato contiene due condizioni nidificate `When` che valutano se:
    1. Vi sono meno di `100` record nella tabella '`Transactions`'.
    2. Il numero di record nella tabella '`Transactions`' non sia un multiplo di `6`

Script di caricamento

```
Transactions:
Load
       0 as id
```

```
Autogenerate 1;

For i = 1 to 100
       when NoOfRows('Transactions') < 100 when mod(NoOfRows('Transactions'),6) <> 0
               Concatenate
    Load
       if(isnull(Peek(id)),1,peek(id)+1) as id
      Autogenerate 7;
next i
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

- id

La tabella dei risultati mostra solo i primi cinque ID di transazione, ma lo script di caricamento crea 36 righe e termina una volta soddisfatta la condizione `When`.

Tabella dei risultati

id
0
1
2
3
4
5
+30 righe in più

Le condizioni `When` nidificate nel ciclo `For` valutano le seguenti domande:

- Sono presenti meno di 100 righe nella tabella '`Transactions`'?
- Il numero totale di record nella tabella '`Transactions`' non è un multiplo di sei?

Ogni volta che entrambe le condizioni `When` restituiscono un valore TRUE, vengono generati altri sette record e concatenati alla tabella '`Transactions`' esistente.

Le condizioni `When` restituiscono un valore TRUE per cinque volte. A questo punto, nella tabella '`Transactions`' vi è un totale di 36 righe di dati.

Quando vengono create 36 righe di dati nella tabella '`Transactions`', la seconda istruzione `When` restituisce il valore FALSE, pertanto l'istruzione LOAD successiva non viene più eseguita.

2.5 Istruzioni regolari dello script

Le istruzioni regolari vengono generalmente utilizzate per la manipolazione dei dati. Queste istruzioni possono essere scritte su un qualsiasi numero di righe nello script e devono sempre terminare con un punto e virgola, ";".

Tutte le parole chiave dello script possono essere immesse con qualsiasi combinazione di caratteri maiuscoli e minuscoli. I nomi dei campi e delle variabili utilizzati nelle istruzioni possono essere immessi indipendentemente dal formato del carattere.

Prospetto delle istruzioni regolari dello script

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Alias

L'istruzione **alias** viene utilizzata per impostare un alias in base al quale verrà rinominato un campo quando questo ricorrerà nello script che segue.

```
Alias fieldname as aliasname {,fieldname as aliasname}
```

Autonumber

Questa istruzione crea un valore intero univoco per ciascun valore calcolato distinto in un campo rilevato durante l'esecuzione dello script.

```
AutoNumber fields [Using namespace] ]
```

Binary

L'istruzione **binary** consente di caricare i dati da un altro documento QlikView, inclusi i dati in Section Access.

```
Binary  [path] filename
```

comment

Fornisce un modo per visualizzare i commenti dei campi (metadati) da database e fogli di calcolo. I nomi di campo non presenti nell'app verranno ignorati. In caso di più ricorrenze di uno stesso nome di campo, verrà applicato l'ultimo valore utilizzato.

```
Comment field *fieldlist using mapname
Comment field fieldname with comment
```

comment table

Fornisce un modo per visualizzare i commenti delle tabelle (metadati) da database o fogli di calcolo.

```
Comment table tablelist using mapname
Comment table tablename with comment
```

Connect

Questa funzionalità non è disponibile in Qlik Sense SaaS.

L'istruzione **CONNECT** consente di definire l'accesso di Qlik Sense a un database generico mediante l'interfaccia OLE DB/ODBC. Per ODBC, occorre innanzitutto specificare la sorgente dati utilizzando l'amministratore ODBC.

```
ODBC Connect TO connect-string [ ( access_info ) ]
OLEDB CONNECT TO connect-string [ ( access_info ) ]
CUSTOM CONNECT TO connect-string [ ( access_info ) ]
LIB CONNECT TO connection
```

Declare

L'istruzione **Declare** consente di creare definizioni di campo in cui è possibile definire relazioni tra i campi o le funzioni. È possibile utilizzare una serie di definizioni di campo per generare automaticamente campi derivati, che possono essere utilizzati come dimensioni. Ad esempio, è possibile creare una definizione di calendario e utilizzarla per generare dimensioni correlate, come ad esempio anno, mese, settimana e giorno da un campo della data.

```
definition_name:
Declare [Field[s]] Definition [Tagged tag_list ]
[Parameters parameter_list ]
Fields field_list
[Groups group_list ]

<definition name>:
Declare [Field][s] Definition
Using <existing_definition>
[With <parameter_assignment> ]
```

Derive

L'istruzione **Derive** consente di generare campi derivati in base a una definizione di campo creata mediante un'istruzione **Declare**. È possibile specificare i campi dati per cui derivare i campi oppure derivarli esplicitamente o implicitamente in base ai tag di campo.

```
Derive [Field[s]] From [Field[s]] field_list Using definition
Derive [Field[s]] From Explicit [Tag[s]] (tag_list) Using definition
Derive [Field[s]] From Implicit [Tag[s]] Using definition
```

Direct Query

L'istruzione **DIRECT QUERY** consente di accedere alle tabelle mediante una connessione ODBC o OLE DB utilizzando la funzione Direct Discovery.

```
Direct Query [path]
```

Directory

L'istruzione **Directory** definisce in quale directory ricercare i file dei dati nelle istruzioni **LOAD** successive finché non viene eseguita una nuova istruzione **Directory**.

```
Directory [path]
```

Disconnect

L'istruzione **Disconnect** termina l'attuale connessione ODBC/OLE DB/Personalizzata. Questa istruzione è opzionale.

```
Disconnect
```

drop field

Durante l'esecuzione dello script, in qualsiasi momento è possibile rilasciare dal modello dati e quindi dalla memoria uno o più campi di Qlik Sense utilizzando l'istruzione **drop field**.

*Sia **drop field** che **drop fields** sono formati consentiti, senza alcuna differenza effettiva. Se nessuna tabella viene specificata, il campo verrà rilasciato da tutte le tabelle in cui ricorre.*

```
Drop field  fieldname [ , fieldname2 ...] [from tablename1 [ , tablename2
...]]
drop fields fieldname [ , fieldname2 ...] [from tablename1 [ , tablename2
...]]
```

drop table

Durante l'esecuzione dello script, è possibile rilasciare dal modello dati, e quindi dalla memoria, una o più tabelle interne di Qlik Sense utilizzando l'istruzione **drop table**.

*I formati **drop table** e **drop tables** sono entrambi accettati.*

```
Drop table  tablename [, tablename2 ...]
drop tables[ tablename [, tablename2 ...]
```

Execute

L'istruzione **Execute** viene utilizzata per eseguire altri programmi, mentre Qlik Sense sta caricando i dati. Ad esempio, per effettuare le connessioni necessarie.

```
Execute commandline
```

FlushLog

L'istruzione **FlushLog** obbliga Qlik Sense a scrivere il contenuto del buffer dello script nel file di registro dello script.

```
FlushLog
```

Force

L'istruzione **force** impone a Qlik Sense di interpretare i nomi e i valori di campo delle istruzioni **LOAD** e **SELECT** successive in formato solo maiuscolo, solo minuscolo, sempre maiuscolo o conformemente alla visualizzazione attuale (formato misto). Questa istruzione permette di associare i valori di campo provenienti da tabelle create in base a convenzioni differenti.

```
Force ( capitalization | case upper | case lower | case mixed )
```

LOAD

L'istruzione **LOAD** carica i campi da un file, dai dati definiti nello script, da una tabella caricata in precedenza, da una pagina Web, dal risultato di un'istruzione **SELECT** seguente o dalla generazione automatica di dati. È anche possibile caricare dati da connessioni di analisi.

```
 Load [ distinct ] *fieldlist
[( from file [ format-spec ] |
from_field fieldassource [format-spec]
inline data [ format-spec ] |
resident table-label |
autogenerate size )]
[ where criterion | while criterion ]
[ group_by groupbyfieldlist ]
[order_by orderbyfieldlist ]
[extension pluginname.functionname(tabledescription)]
```

Let

L'istruzione **let** è un complemento all'istruzione **set**, utilizzata per definire le variabili degli script. L'istruzione **let**, a differenza dell'istruzione **set**, valuta l'espressione posta sul lato destro del simbolo '=' al tempo di esecuzione dello script prima dell'assegnazione alla variabile.

```
Let variablename=expression
```

Loosen Table

Una o più tabelle dati interne di Qlik Sense possono essere dichiarate logicamente disconnesse in modo esplicito durante l'esecuzione dello script, utilizzando un'istruzione **Loosen Table**. Quando una tabella è logicamente disconnessa, tutte le associazioni tra i valori di campo nella tabella vengono rimosse. È possibile ottenere un effetto simile caricando ogni campo della tabella logicamente disconnessa come tabelle indipendenti e scollegate. La disconnessione logica può rivelarsi utile durante il controllo per isolare temporaneamente parti differenti della struttura dei dati. Nel visualizzatore tabelle è possibile riconoscere una tabella logicamente disconnessa dalle linee punteggiate. L'utilizzo di una o più istruzioni **Loosen Table** nello script indica a Qlik Sense di ignorare ogni impostazione di tabelle logicamente disconnesse effettuata prima dell'esecuzione dello script.

```
tablename [ , tablename2 ...]
Loosen Tables tablename [ , tablename2 ...]
```

Map ... using

L'istruzione **map ... using** viene usata per eseguire il mapping di un certo valore di campo o una certa espressione sui valori di una tabella di mapping specifica. La tabella di mapping viene creata utilizzando l'istruzione **Mapping**.

```
Map  *fieldlist Using  mapname
```

NullAsNull

L'istruzione **NullAsNull** disattiva la conversione dei valori NULL in valori di stringa impostati in precedenza da un'istruzione **NullAsValue**.

```
NullAsNull *fieldlist
```

NullAsValue

L'istruzione **NullAsValue** specifica per quali campi NULL deve essere convertito in un valore.

```
NullAsValue *fieldlist
```

Qualify

L'istruzione **Qualify** consente di modificare la qualificazione dei nomi di campo, ad esempio il nome della tabella dei nomi di campo diventerà un prefisso.

```
Qualify *fieldlist
```

Rem

L'istruzione **rem** viene utilizzata per inserire osservazioni, o commenti, negli script o per disattivare temporaneamente istruzioni dello script senza rimuoverle.

```
Rem string
```

Rename Field

Questa funzione di script rinomina uno o più campi di Qlik Sense esistenti dopo che sono stati caricati.

```
Rename field (using mapname|oldname to newname{ , oldname to newname })
```

```
Rename Fields (using mapname | oldname to newname{ , oldname to newname })
```

Rename Table

Questa funzione di script rinomina una o più tabelle interne di Qlik Sense esistenti dopo che sono state caricate.

```
Rename table  (using mapname | oldname to newname{ , oldname to newname })
Rename Tables (using mapname | oldname to newname{ , oldname to newname })
```

Section

L'istruzione **section** consente di definire se le istruzioni successive **LOAD** e **SELECT** devono essere considerate come dati o come una definizione dei diritti di accesso.

```
Section (access | application)
```

Select

La selezione dei campi da una sorgente dati ODBC o da un provider OLE DB viene eseguita utilizzando le istruzioni SQL **SELECT** standard. Tuttavia, l'ambito nel quale le istruzioni **SELECT** vengono accettate dipende dal driver ODBC o dal provider OLE DB utilizzato.

```
Select [all | distinct | distinctrow | top n [percent] ] *fieldlist

From tablelist

[Where criterion ]

[Group by fieldlist [having criterion ] ]

[Order by fieldlist [asc | desc] ]

[ (Inner | Left | Right | Full)Join tablename on fieldref = fieldref ]
```

Set

L'istruzione **set** viene utilizzata per definire le variabili di script. Le variabili possono essere utilizzate per sostituire stringhe, percorsi, unità e così via.

```
Set variablename=string
```

Sleep

L'istruzione **sleep** interrompe l'esecuzione dello script per il periodo di tempo specificato.

```
Sleep n
```

SQL

L'istruzione **SQL** consente di inviare un comando arbitrario SQL tramite una connessione ODBC o OLE DB.

```
SQL sql_command
```

SQLColumns

L'istruzione **sqlcolumns** restituisce un gruppo di campi che descrivono le colonne di una sorgente dati ODBC o OLE DB sulla quale è stata eseguita un'istruzione **connect**.

```
SQLColumns
```

SQLTables

L'istruzione **sqltables** restituisce un gruppo di campi che descrivono le tabelle di una sorgente dati ODBC o OLE DB sulla quale è stata eseguita un'istruzione **connect**.

```
SQLTables
```

SQLTypes

L'istruzione **sqltypes** restituisce un gruppo di campi che descrivono i tipi di una sorgente dati ODBC o OLE DB sulla quale è stata eseguita un'istruzione **connect**.

```
SQLTypes
```

Star

La stringa utilizzata per rappresentare l'insieme di tutti i valori di un campo nel database può essere impostata tramite l'istruzione **star**. Interessa le istruzioni **LOAD** e **SELECT** successive.

```
Star is [ string ]
```

Store

L'istruzione **Store** crea un file QVD, o text.

```
Store [ *fieldlist from] table into filename [ format-spec ];
```

Tag

Questa istruzione dello script fornisce un modo per assegnare tag a uno o più campi o tabelle. Se viene effettuato un tentativo di contrassegnare un campo o una tabella non presente nell'app, tale contrassegno verrà ignorato. In caso di conflitto nelle ricorrenze di un nome di campo o di tag, si utilizza l'ultimo valore trovato.

```
Tag[field|fields] fieldlist with tagname
Tag [field|fields] fieldlist using mapname
Tag table tablelist with tagname
```

Trace

L'istruzione **trace** esegue la scrittura di una stringa nella finestra **Avanzamento dell'esecuzione dello script** e nel file di log dello script, quando viene utilizzato. Si rivela molto utile per le operazioni di debug. L'uso delle espansioni $ delle variabili calcolate prima dell'istruzione **trace** consente di personalizzare il messaggio.

```
Trace string
```

Unmap

L'istruzione **Unmap** disattiva il mapping del valore di campo specificato da un'istruzione **Map ... Using** precedente per i campi caricati successivamente.

```
Unmap *fieldlist
```

Unqualify

L'istruzione **Unqualify** viene utilizzata per disattivare la qualificazione dei nomi di campo che era stata precedentemente attivata dall'istruzione **Qualify**.

```
Unqualify *fieldlist
```

Untag

Questa istruzione dello script fornisce un modo per rimuovere tag da campi o tabelle. Se viene effettuato un tentativo di rimozione di un contrassegno da un campo o una tabella non presente nell'app, tale rimozione verrà ignorata.

```
Untag[field|fields] fieldlist with tagname
Tag [field|fields] fieldlist using mapname
Tag table tablelist with tagname
```

Alias

L'istruzione **alias** viene utilizzata per impostare un alias in base al quale verrà rinominato un campo quando questo ricorrerà nello script che segue.

Sintassi:

```
alias fieldname as aliasname {,fieldname as aliasname}
```

Argomenti:

Argomenti

Argomento	Descrizione
fieldname	Il nome del campo nei dati sorgente
aliasname	Un nome di alias che si desidera utilizzare

Esempi e risultati:

Esempio	Risultato
`Alias ID_N as NameID;`	
`Alias A as Name, B as Number, C as Date;`	Le modifiche dei nomi definite tramite questa istruzione vengono applicate a tutte le istruzioni **SELECT** e **LOAD** successive. È possibile definire un nuovo alias per un nome di campo mediante una nuova istruzione **alias** in qualsiasi posizione successiva nello script.

AutoNumber

Questa istruzione crea un valore intero univoco per ciascun valore calcolato distinto in un campo rilevato durante l'esecuzione dello script.

È possibile utilizzare la funzione *autonumber (page 571)* anche all'interno di un'istruzione **LOAD**, ma vi sono alcune limitazioni se si desidera utilizzare un caricamento ottimizzato. È possibile creare un caricamento ottimizzato caricando prima i dati da un file **QVD** e poi utilizzando l'istruzione **AutoNumber** per convertire i valori in chiavi di simboli.

Sintassi:

```
AutoNumber *fieldlist [Using namespace] ]
```

Argomenti:

Argomenti

Argomento	Descrizione
*fieldlist	Un elenco separato da virgole dei campi i cui valori devono essere sostituiti da un valore intero univoco. È possibile utilizzare i caratteri jolly ? e * nei nomi dei campi per includere tutti i campi con nomi corrispondenti. È anche possibile utilizzare * per includere tutti i campi. Quando si utilizzano caratteri jolly, racchiudere i nomi dei campi tra virgolette.
namespace	L'uso di `namespace` è opzionale. È possibile utilizzare questa opzione se si desidera creare uno spazio dei nomi, in cui valori identici in campi diversi condividono la stessa chiave. Se non si utilizza questa opzione, tutti i campi saranno indicizzati con una chiave separata.

Limiti:

Quando nello script sono presenti più istruzioni **LOAD**, è necessario inserire l'istruzione **AutoNumber** dopo l'istruzione **LOAD** finale.

Esempio - script con AutoNumber

Esempio di script

In questo esempio, i dati vengono prima caricati senza l'istruzione **AutoNumber**. L'istruzione **AutoNumber** viene quindi aggiunta per mostrare l'effetto.

Dati utilizzati nell'esempio

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare l'esempio di script in basso. Lasciare l'istruzione **AutoNumber** non commentata per il momento.

```
RegionSales: LOAD *,  Region &'|'& Year &'|'& Month as KeyToOtherTable INLINE [ Region, Year,
Month, Sales North,     2014,   May,    245 North,      2014,   May,    347 North,      2014,   June,   127 S
June,   645 South,      2013,   May,    367 South,      2013,   May,    221 ];
&'|'& Year &'|'& Month as KeyToOtherTable INLINE [Region, Year, Month, Budget North,  2014,
May,    200 North,  2014,   May,    350 North,  2014,   June,   150 South,  2014,   June,
500 South,  2013,   May,    300 South,  2013,   May,    200 ];  //AutoNumber KeyToOtherTable;
```

Creazione di visualizzazioni

Creare due visualizzazioni di tabelle in un foglio Qlik Sense. Aggiungere **KeyToOtherTable**, **Region**, **Year**, **Month** e **Sales** come dimensioni alla prima tabella. Aggiungere **KeyToOtherTable**, **Region**, **Year**, **Month** e **Budget** come dimensioni alla seconda tabella.

Risultato

Tabella RegionSales

KeyToOtherTable	Region	Year	Month	Sales
North|2014|June	North	2014	June	127
North|2014|May	North	2014	May	245
North|2014|May	North	2014	May	347
South|2013|May	South	2013	May	221
South|2013|May	South	2013	May	367
South|2014|June	South	2014	June	645

Tabella budget

KeyToOtherTable	Region	Year	Month	Budget
North|2014|June	North	2014	June	150
North|2014|May	North	2014	May	200
North|2014|May	North	2014	May	350
South|2013|May	South	2013	May	200
South|2013|May	South	2013	May	300
South|2014|June	South	2014	June	500

Spiegazione

L'esempio mostra un campo composito **KeyToOtherTable** che collega le due tabelle. **AutoNumber** non è utilizzato. Notare la lunghezza dei valori **KeyToOtherTable**.

Aggiungi istruzione AutoNumber

Eliminare il commento all'istruzione **AutoNumber** nello script di caricamento.

```
AutoNumber KeyToOtherTable;
```

Risultato

Tabella RegionSales

KeyToOtherTable	Region	Year	Month	Sales
1	North	2014	June	127
1	North	2014	May	245
2	North	2014	May	347
3	South	2013	May	221

KeyToOtherTable	Region	Year	Month	Sales
4	South	2013	May	367
4	South	2014	June	645

Tabella budget

KeyToOtherTable	Region	Year	Month	Budget
1	North	2014	June	150
1	North	2014	May	200
2	North	2014	May	350
3	South	2013	May	200
4	South	2013	May	300
4	South	2014	June	500

Spiegazione

I valori di campo **KeyToOtherTable** sono stati sostituiti con valori interi univoci e, di conseguenza, la lunghezza dei valori dei campi è stata ridotta, conservando così la memoria. I campi chiave di entrambe le tabelle sono interessati da **AutoNumber** e le tabelle rimangono collegate. L'esempio è breve a scopo dimostrativo, ma risulterebbe significativo con una tabella contenente un elevato numero di righe.

Binary

L'istruzione **binary** consente di caricare i dati da un'altra app Qlik Sense o documento QlikView, inclusi i dati in Section Access. Gli altri elementi dell'app non sono inclusi, ad esempio fogli, racconti, visualizzazioni, voci principali o variabili.

Nello script è consentita una sola istruzione **binary**. L'istruzione **binary** deve essere la prima istruzione dello script, ancora prima delle istruzioni SET, che sono generalmente poste all'inizio dello script.

Sintassi:

```
binary [path] filename
```

Argomenti:

Argomenti

Argomento	Descrizione
path	Il percorso del file che deve corrispondere a un riferimento a una connessione dati della cartella. Ciò si rivela necessario se il file non si trova nella directory di lavoro di Qlik Sense. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:**data*\ • relativo dell'app contenente questa riga di script **Esempio:** *data*\
filename	Il nome del file, inclusa l'estensione del file .qvw o .qvf.

Limiti:

Non è possibile utilizzare **binary** per caricare dati da un'app nella stessa distribuzione di Qlik Sense Enterprise facendo riferimento all'ID dell'app. Il caricamento può essere effettuato solo da un file *.qvf*.

Esempi

Stringa	Descrizione
`Binary lib://DataFolder/customer.qvw;`	In questo esempio, il file deve essere inserito nella connessione dati della **cartella**. Tale posizione può essere, ad esempio, una cartella creata dall'amministratore sul server Qlik Sense. Fare clic su **Crea nuova connessione** nell'editor caricamento dati e selezionare **Cartella** in **Percorsi file**.
`Binary customer.qvf;`	In questo esempio, il file deve essere inserito nella directory di lavoro Qlik Sense.
`Binary c:\qv\customer.qvw;`	Questo esempio, in cui viene utilizzato un percorso di file assoluto, può essere utilizzato solo nella modalità di creazione degli script legacy.

Comment field

Fornisce un modo per visualizzare i commenti dei campi (metadati) da database e fogli di calcolo. I nomi di campo non presenti nell'app verranno ignorati. In caso di più ricorrenze di uno stesso nome di campo, verrà applicato l'ultimo valore utilizzato.

Sintassi:

```
comment [fields] *fieldlist using mapname
comment [field] fieldname with comment
```

La tabella di mapping utilizzata deve essere composta da due colonne, la prima contenente i nomi dei campi e la seconda i commenti.

Argomenti:

Argomenti

Argomento	Descrizione
**fieldlist*	Un elenco separato da virgole dei campi da commentare. L'utilizzo di * per l'elenco dei campi indica tutti i campi. Nei nomi di campo sono consentiti i caratteri speciali * e ?. Se si utilizzano i caratteri speciali può essere necessario delimitare i nomi di campo tra virgolette.
mapname	Il nome di una tabella di mapping letta in precedenza in un'istruzione mapping **LOAD** o mapping **SELECT**.
fieldname	Il nome del campo al quale si desidera aggiungere il commento.
comment	Il commento da aggiungere al campo.

Example 1:

```
commentmap:
mapping LOAD * inline [
a,b
Alpha,This field contains text values
Num,This field contains numeric values
];
comment fields using commentmap;
```

Example 2:

```
comment field Alpha with AFieldContainingCharacters;
comment field Num with '*A field containing numbers';
comment Gamma with 'Mickey Mouse field';
```

Comment table

Fornisce un modo per visualizzare i commenti delle tabelle (metadati) da database o fogli di calcolo.

I nomi di tabella non presenti nell'app vengono ignorati. In caso di più ricorrenze di uno stesso nome di tabella, viene utilizzato l'ultimo valore trovato. La parola chiave può essere utilizzata per leggere i commenti da una sorgente dati.

Sintassi:

```
comment [tables] tablelist using mapname
comment [table] tablename with comment
```

Argomenti:

Argomenti

Argomento	Descrizione
tablelist	(table{,table})
mapname	Il nome di una tabella di mapping letta in precedenza in un'istruzione mapping **LOAD** o mapping **SELECT**.
tablename	Il nome della tabella alla quale si desidera aggiungere il commento.
comment	Il commento da aggiungere alla tabella.

Example 1:

```
Commentmap:
mapping LOAD * inline [
a,b
Main,This is the fact table
Currencies, Currency helper table
];
comment tables using Commentmap;
```

Example 2:

```
comment table Main with 'Main fact table';
```

Connect

L'istruzione **CONNECT** consente di definire l'accesso di Qlik Sense a un database generico mediante l'interfaccia OLE DB/ODBC. Per ODBC, occorre innanzitutto specificare la sorgente dati utilizzando l'amministratore ODBC.

Questa funzionalità non è disponibile in Qlik Sense SaaS.

Questa istruzione supporta solo le connessioni dati della cartella in modalità standard.

Sintassi:

```
ODBC CONNECT TO connect-string
OLEDB CONNECT TO connect-string
```

```
CUSTOM CONNECT TO connect-string
LIB CONNECT TO connection
```

Argomenti:

Argomenti

Argomento	Descrizione

connect-string	`connect-string ::= datasourcename { ; conn-spec-item }`
	La stringa di connessione è il nome della sorgente dati e un elenco opzionale di uno o più voci di specifica della connessione. Se il nome della sorgente dati contiene spazi vuoti o se vengono elencati voci di specifica della connessione, la stringa di connessione dovrà essere racchiusa tra virgolette.
	
	datasourcename deve essere una sorgente dati ODBC definita o una stringa che definisce un provider OLE DB.
	
	`conn-spec-item ::=`**`DBQ=`**`database_specifier |`**`DriverID=`**`driver_specifier |`**`UID=`**`userid |`**`PWD=`**`password`
	
	Le possibili voci di specifica della connessione possono differire a seconda del database. Per alcuni database sono disponibili anche voci diverse da quelle indicate. Per OLE DB, alcune delle voci di specifica della connessione sono obbligatori e non facoltativi.

connection	Il nome di una connessione dati memorizzata nell'editor caricamento dati.

Se **ODBC** è posizionato prima di **CONNECT**, verrà utilizzata l'interfaccia ODBC. In caso contrario, verrà utilizzata l'interfaccia OLE DB.

L'utilizzo di **LIB CONNECT TO** determina la connessione a un database mediante una connessione dati memorizzata che è stata creata nell'editor caricamento dati.

Example 1:

```
ODBC CONNECT TO 'Sales
DBQ=C:\Program Files\Access\Samples\Sales.mdb';
```

La sorgente dati definita da questa istruzione verrà utilizzata dalle successive istruzioni **Select (SQL)**, finché non verrà eseguita una nuova istruzione **CONNECT**.

Example 2:

```
LIB CONNECT TO 'DataConnection';
```

Connect32

Questa istruzione viene utilizzata come l'istruzione **CONNECT**, ma impone a un sistema a 64 bit di utilizzare un provider ODBC/OLE DB a 32 bit. Non applicabile in caso di istruzione connect personalizzata.

Connect64

Questa istruzione viene utilizzata come l'istruzione **CONNECT**, ma impone l'utilizzo di un provider a 64 bit. Non applicabile in caso di istruzione connect personalizzata.

Declare

L'istruzione **Declare** consente di creare definizioni di campo in cui è possibile definire relazioni tra i campi o le funzioni. È possibile utilizzare una serie di definizioni di campo per generare automaticamente campi derivati, che possono essere utilizzati come dimensioni. Ad esempio, è possibile creare una definizione di calendario e utilizzarla per generare dimensioni correlate, come ad esempio anno, mese, settimana e giorno da un campo della data.

È possibile utilizzare **Declare** per impostare una nuova definizione di campo o per creare una definizione di campo basata su una definizione già esistente.

Impostazione di una nuova definizione di campo

Sintassi:

```
definition_name:
Declare [Field[s]] Definition [Tagged tag_list ]
[Parameters parameter_list ]
Fields field_list
```

Argomenti:

Argomento	Descrizione
definition_name	Nome della definizione di campo, che termina con i due punti. *Non utilizzare autoCalendar come definizione dei campi del calendario, in quanto questo nome è riservato per i modelli del calendario generati automaticamente.* **Esempio:** `Calendar:`
tag_list	Un elenco separato da virgole di tag da applicare ai campi derivati dalla definizione di campo. L'applicazione dei tag è opzionale, tuttavia, se non si applicano i tag utilizzati per specificare l'ordinamento, ad esempio $date, $numeric o $text, per impostazione predefinita il campo derivato verrà ordinato in base all'ordine di caricamento. **Esempio:** `'$date'Thank you for bringing this to our attention, and apologies for the inconvenience.`

Argomento	Descrizione
parameter_list	Un elenco separato da virgole di parametri. A un parametro definito nella forma `name=value` viene assegnato un valore iniziale che può essere sostituito quando viene riutilizzata la definizione di campo. Opzionale. **Esempio:** `first_month_of_year = 1`
field_list	Un elenco separato da virgole di campi da generare quando viene utilizzata la definizione di campo. Un campo è definito nella forma `<expression> As field_name tagged tag`. Utilizzare `$1` per fare riferimento al campo dati dal quale devono essere generati i campi derivati. **Esempio:** `Year($1) As Year tagged ('$numeric')`

Esempio:

```
Calendar:
DECLARE FIELD DEFINITION TAGGED '$date'
   Parameters
      first_month_of_year = 1
   Fields
      Year($1) As Year Tagged ('$numeric'),
      Month($1) as Month Tagged ('$numeric'),
      Date($1) as Date Tagged ('$date'),
      Week($1) as Week Tagged ('$numeric'),
      Weekday($1) as Weekday Tagged ('$numeric'),
      DayNumberOfYear($1, first_month_of_year) as DayNumberOfYear Tagged ('$numeric')
;
```

Il calendario è ora definito ed è possibile applicarlo ai campi data caricati, in questo caso OrderDate e ShippingDate, utilizzando una clausola **Derive**.

Riutilizzo di una definizione di campo esistente

Sintassi:

```
<definition name>:
Declare [Field][s] Definition
Using <existing_definition>
[With <parameter_assignment> ]
```

Argomenti:

Argomento	Descrizione
definition_name	Nome della definizione di campo, che termina con i due punti. **Esempio:** `MyCalendar:`
existing_definition	La definizione di campo da riutilizzare durante la creazione di una nuova definizione di campo. La nuova definizione di campo funziona allo stesso modo della definizione sulla quale si basa, con l'eccezione dell'uso di parameter_assignment per modificare un valore utilizzato nelle espressioni di campo. **Esempio:** `Using Calendar`
parameter_assignment	Un elenco separato da virgole di assegnazioni di parametri. Un'assegnazione di parametro viene definita nella forma `name=value` e sostituisce il valore del parametro impostato nella definizione di campo di base. Opzionale. **Esempio:** `first_month_of_year = 4`

Esempio:

In questo esempio viene riutilizzata la definizione di calendario creata nell'esempio precedente. In questo caso si desidera utilizzare un anno fiscale che inizi ad aprile. Ciò si ottiene assegnando il valore 4 al parametro first_month_of_year, che influirà sul campo DayNumberOfYear definito.

Nell'esempio si suppone di utilizzare i dati campione e la definizione di campo dell'esempio precedente.

```
MyCalendar:
DECLARE FIELD DEFINITION USING Calendar WITH first_month_of_year=4;

DERIVE FIELDS FROM FIELDS OrderDate,ShippingDate USING MyCalendar;
```

Una volta ricaricato lo script di dati, i campi generati sono disponibili nell'editor dei fogli con i nomi OrderDate.MyCalendar.* e ShippingDate.MyCalendar.*.

Derive

L'istruzione **Derive** consente di generare campi derivati in base a una definizione di campo creata mediante un'istruzione **Declare**. È possibile specificare i campi dati per cui derivare i campi oppure derivarli esplicitamente o implicitamente in base ai tag di campo.

Sintassi:

```
Derive [Field[s]] From [Field[s]] field_list Using definition
```

```
Derive [Field[s]] From Explicit [Tag[s]] tag_list Using definition
Derive [Field[s]] From Implicit [Tag[s]] Using definition
```

Argomenti:

Argomenti

Argomento	Descrizione
definition	Nome della definizione di campo da utilizzare per la derivazione dei campi. **Esempio:** `Calendar`
field_list	Un elenco separato da virgole di campi dati da cui devono essere generati i campi derivati, in base alla definizione di campo. I campi dati devono essere campi già caricati nello script. **Esempio:** `OrderDate, ShippingDate`
tag_list	Un elenco separato da virgole di tag. I campi derivati verranno generati per tutti i campi dati con uno qualsiasi dei tag inclusi nell'elenco. L'elenco dei tag deve essere racchiuso tra parentesi tonde. **Esempio:** `('$date','$timestamp')`

Esempi:

- Derivare campi per campi dati specifici.
  In questo caso è necessario specificare i campi OrderDate e ShippingDate.
  `DERIVE FIELDS FROM FIELDS OrderDate,ShippingDate USING Calendar;`
- Derivare campi per tutti i campi con un tag specifico.
  In questo caso è necessario derivare i campi in base a Calendar per tutti i campi con un tag `$date`.
  `DERIVE FIELDS FROM EXPLICIT TAGS ('$date') USING Calendar;`
- Derivare campi per tutti i campi con il tag della definizione di campo.
  In questo caso è necessario derivare i campi per tutti i campi dati con lo stesso tag della definizione di campo Calendar, in questo caso `$date`.
  `DERIVE FIELDS FROM IMPLICIT TAG USING Calendar;`

Direct Query

L'istruzione **DIRECT QUERY** consente di accedere alle tabelle mediante una connessione ODBC o OLE DB utilizzando la funzione Direct Discovery.

Sintassi:

```
DIRECT QUERY DIMENSION fieldlist [MEASURE fieldlist] [DETAIL fieldlist] FROM
tablelist
[WHERE where_clause]
```

Le parole chiave **DIMENSION**, **MEASURE** e **DETAIL** possono essere utilizzate in qualsiasi ordine.

Le clausole con parole chiave **DIMENSION** e **FROM** sono richieste in tutte le istruzioni **DIRECT QUERY**. La parola chiave **FROM** deve essere collocata dopo la parola chiave **DIMENSION**.

I campi specificati direttamente dopo la parola chiave **DIMENSION** vengono caricati in memoria e possono essere utilizzati per creare associazioni tra dati in memoria e dati Direct Discovery.

*L'istruzione **DIRECT QUERY** non può contenere le clausole **DISTINCT** o **GROUP BY**.*

La parola chiave **MEASURE** consente di definire i campi che Qlik Sense riconosce su un "metalivello". I dati effettivi di un campo di misura risiedono nel database solo durante il processo di caricamento dei dati e vengono recuperati ad hoc dalle espressioni grafiche utilizzate in una visualizzazione.

Generalmente, i campi contenenti valori discreti che verranno utilizzati come dimensioni devono essere caricati con la parola chiave **DIMENSION**, mentre i numeri che verranno utilizzati solo nelle aggregazioni devono essere selezionati con la parola chiave **MEASURE**.

I campi **DETAIL** forniscono informazioni o dettagli, ad esempio campi dei commenti, che un utente può desiderare di visualizzare in una tabella di analisi dei dettagli. I campi **DETAIL** non possono essere utilizzati nelle espressioni grafiche.

Per impostazione predefinita, l'istruzione **DIRECT QUERY** è una sorgente dati neutra per le sorgenti dati che supportano SQL. Per tale motivo, è possibile utilizzare la stessa l'istruzione **DIRECT QUERY** per database SQL diversi senza che sia necessario apportare modifiche. Direct Discovery genera query adatte al database in base alle esigenze.

La sintassi della sorgente dati nativa può essere utilizzata quando l'utente conosce il database a cui inviare le query e desidera utilizzare estensioni specifiche del database in SQL. Viene supportata la sintassi della sorgente dati nativa:

- Come espressioni di campo nelle clausole **DIMENSION** e **MEASURE**
- Come contenuto della clausola **WHERE**

Esempi:

```
DIRECT QUERY

      DIMENSION Dim1, Dim2
      MEASURE

            NATIVE ('X % Y') AS X_MOD_Y

FROM TableName
DIRECT QUERY

      DIMENSION Dim1, Dim2
      MEASURE X, Y
      FROM TableName
      WHERE NATIVE ('EMAIL MATCHES "\*.EDU"')
```

I seguenti termini vengono utilizzati come parole chiave perciò non possono essere utilizzati come nomi di colonna o campo senza che vengano racchiusi tra virgolette: and, as, detach, detail, dimension, distinct, from, in, is, like, measure, native, not, or, where

Argomenti:

Argomento	Descrizione
fieldlist	Un elenco separato da virgole di specifiche dei campi, *fieldname {, fieldname}*. Una specifica di un campo può essere costituita da un nome di campo, condizione in cui lo stesso nome viene utilizzato per il nome di colonna del database e il nome di campo di Qlik Sense. Oppure una specifica di campo può essere costituita da un "alias di campo", nel cui caso a un'espressione di database o a un nome di colonna viene assegnato un nome di campo di Qlik Sense.
tablelist	Un elenco di nomi di tabelle o di viste del database da cui vengono caricati i dati. In genere, saranno viste contenenti un JOIN che viene eseguito sul database.

where_clause	In questo caso, non viene definita la sintassi completa delle clausole **WHERE**, tuttavia è consentita la maggior parte delle "espressioni relazioni" di SQL, compreso l'utilizzo delle chiamate di funzione, l'operatore **LIKE** per le stringhe, **IS NULL** e **IS NOT NULL** e **IN. BETWEEN** non è compreso.
	
	NOT è un operatore unario, contrariamente al modificatore di alcune parole chiave.
	
	Esempi:
	
	`WHERE x > 100 AND "Region Code" IN ('south', 'west')`
	`WHERE Code IS NOT NULL and Code LIKE '%prospect'`
	`WHERE NOT X in (1,2,3)`
	L'ultimo esempio non può essere scritto come:
	
	`WHERE X NOT in (1,2,3)`

Esempio:

In questo esempio viene utilizzata una tabella del database denominata TableName, contenente i campi Dim1, Dim2, Num1, Num2 e Num3.Dim1 e Dim2 verranno caricati nella serie di dati Qlik Sense.

```
DIRECT QUERY DIMENSTION Dim1, Dim2 MEASURE Num1, Num2, Num3 FROM TableName ;
```

Dim1 e Dim2 saranno disponibili per essere utilizzati come dimensioni. Num1, Num2 e Num3 saranno disponibili per le aggregazioni. Anche Dim1 e Dim2 saranno disponibili per le aggregazioni. Il tipo di aggregazioni per le quali è possibile utilizzare Dim1 e Dim2 dipende dai relativi tipi di dati. Ad esempio, in molti casi i campi **DIMENSION** contengono dati di stringa, ad esempio nomi o numeri di account. Questi dati non possono essere sommati, tuttavia possono essere conteggiati: `count(Dim1)`.

*Le istruzioni **DIRECT QUERY** vengono scritte direttamente nell'editor di script. Per semplificare la costruzione delle istruzioni **DIRECT QUERY**, è possibile generare un'istruzione **SELECT** da una connessione dati, quindi modificare lo script generato per trasformarlo in un'istruzione **DIRECT QUERY**.*
*Ad esempio, l'istruzione **SELECT**:*

```
SQL SELECT
 SalesOrderID,
 RevisionNumber,
 OrderDate,
 SubTotal,
 TaxAmt
FROM MyDB.Sales.SalesOrderHeader;
```

*può essere modificata nella seguente istruzione **DIRECT QUERY**:*

```
DIRECT QUERY
 DIMENSION
 SalesOrderID,
 RevisionNumber

MEASURE
 SubTotal,
 TaxAmt

DETAIL
OrderDate

FROM MyDB.Sales.SalesOrderHeader;
```

Elenchi dei campi Direct Discovery

Un elenco del campo è un elenco separato da virgola di specifiche di campo, *fieldname {, fieldname}*. Una specifica di un campo può essere costituita da un nome di campo, condizione in cui lo stesso nome viene utilizzato per il nome della colonna del database e per il nome di campo. Oppure una specifica di campo può essere costituita da un alias di campo, nel cui caso a un'espressione di database o a un nome di colonna viene assegnato un nome di campo di Qlik Sense.

I nomi di campo possono essere semplici oppure racchiusi tra virgolette. Un nome semplice inizia con un carattere alfabetico Unicode ed è seguito da una qualsiasi combinazione di caratteri alfanumerici o caratteri di sottolineatura. I nomi racchiusi tra virgolette iniziano con virgolette doppie e contengono qualsiasi sequenza di caratteri. Se un nome racchiuso tra virgolette contiene virgolette doppie, tali virgolette vengono rappresentate utilizzando due virgolette doppie adiacenti.

I nomi di campo di Qlik Sense rispettano la distinzione tra maiuscole e minuscole. La distinzione tra maiuscole e minuscole per i nomi di campo del database varia a seconda del database. Una query Direct Discovery mantiene la distinzione utilizzata in tutti gli identificatori e alias del campo. Nell'esempio seguente l'alias "MyState" viene utilizzato internamente per memorizzare i dati dalla colonna del database "STATEID".

```
DIRECT QUERY Dimension STATEID as MyState Measure AMOUNT from SALES_TABLE;
```

Il risultato sarà diverso rispetto a quello di un'istruzione **SQL Select** con un alias. Se l'alias non viene racchiuso tra virgolette in modo esplicito, il risultato presenterà la distinzione tra maiuscole e minuscole predefinita della colonna restituita dal database di destinazione. Nell'esempio seguente l'istruzione **SQL Select** in un database Oracle farà in modo che "MYSTATE," presenti solo lettere maiuscole, come l'alias interno di Qlik Sense, anche se per l'alias viene specificato il formato misto. L'istruzione **SQL Select** utilizza il nome della colonna restituito dal database, che nel caso di Oracle è tutto in formato maiuscolo.

```
SQL Select STATEID as MyState, STATENAME from STATE_TABLE;
```

Per evitare questo comportamento, utilizzare l'istruzione LOAD per specificare l'alias.

```
Load STATEID as MyState, STATENAME;
SQL Select STATEID, STATEMENT from STATE_TABLE;
```

In questo esempio la colonna "STATEID" viene memorizzata internamente da Qlik Sense come "MyState".

La maggior parte delle espressioni scalari del database è consentita come specifiche di campo. Nelle specifiche di campo è possibile utilizzare anche le chiamate di funzione. Le espressioni possono contenere costanti booleane, numeriche o stringhe contenute tra virgolette singole (le virgolette singole incorporate sono rappresentate da virgolette singole adiacenti).

Esempi:

```
DIRECT QUERY

   DIMENSION

      SalesOrderID, RevisionNumber

   MEASURE

      SubTotal AS "Sub Total"

FROM AdventureWorks.Sales.SalesOrderHeader;


DIRECT QUERY

   DIMENSION

      "SalesOrderID" AS "Sales Order ID"

   MEASURE

      SubTotal,TaxAmt,(SubTotal-TaxAmt) AS "Net Total"
```

```
      FROM AdventureWorks.Sales.SalesOrderHeader;


      DIRECT QUERY

         DIMENSION

            (2*Radius*3.14159) AS Circumference,

            Molecules/6.02e23 AS Moles

         MEASURE

            Num1 AS numA

      FROM TableName;
```

```
DIRECT QUERY
   DIMENSION
      concat(region, 'code') AS region_code
   MEASURE
      Num1 AS NumA
FROM TableName;
```

Direct Discovery non supporta l'utilizzo di aggregazioni nelle istruzioni **LOAD**. Se vengono utilizzate le aggregazioni, i risultati potrebbero essere imprevedibili. Si consiglia di non utilizzare un'istruzione **LOAD** come quella seguente:

```
DIRECT QUERY DIMENSION stateid, SUM(amount*7) AS MultiFirst MEASURE amount FROM sales_table;
```

La funzione **SUM** non deve essere contenuta nell'istruzione **LOAD**.

Anche Direct Discovery non supporta le funzioni di Qlik Sense nelle istruzioni **Direct Query**. Ad esempio, la specifica seguente per il campo **DIMENSION** restituisce un errore quando il campo "Mth" viene utilizzato come dimensione in una visualizzazione:

```
month(ModifiedDate) as Mth
```

Directory

L'istruzione **Directory** definisce in quale directory ricercare i file dei dati nelle istruzioni **LOAD** successive finché non viene eseguita una nuova istruzione **Directory**.

Sintassi:

```
Directory[path]
```

Se l'istruzione **Directory** viene emessa senza **path** o non viene inclusa, Qlik Sense eseguirà la ricerca nella directory di lavoro di Qlik Sense.

Argomenti:

Argomenti

Argomento	Descrizione
path	Un testo può essere interpretato come il percorso del file data. Il percorso corrisponde al percorso del file, che può essere: • assoluto **Esempio: *c:\data*** • relativo alla directory di lavoro dell'app Qlik Sense **Esempio: *data*** • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio: *http://www.qlik.com***

Esempi:

```
DIRECTORY C:\userfiles\data; // OR -> DIRECTORY data\

LOAD * FROM
[data1.csv] // ONLY THE FILE NAME CAN BE SPECIFIED HERE (WITHOUT THE FULL PATH)
(ansi, txt, delimiter is ',', embedded labels);

LOAD * FROM
[data2.txt] // ONLY THE FILE NAME CAN BE SPECIFIED HERE UNTIL A NEW DIRECTORY STATEMENT IS
MADE
(ansi, txt, delimiter is '\t', embedded labels);
```

Disconnect

L'istruzione **Disconnect** termina l'attuale connessione ODBC/OLE DB/Personalizzata. Questa istruzione è opzionale.

Sintassi:

```
Disconnect
```

La connessione viene terminata automaticamente quando viene eseguita una nuova istruzione **connect** o quando verrà completata l'esecuzione dello script.

Esempio:

```
Disconnect;
```

Drop

La parola chiave dello script **Drop** può essere utilizzata per eliminare tabelle o campi dal database.

Drop field

Durante l'esecuzione dello script, in qualsiasi momento è possibile rilasciare dal modello dati e quindi dalla memoria uno o più campi di Qlik Sense utilizzando l'istruzione **drop field**.

*Sia **drop field** che **drop fields** sono formati consentiti, senza alcuna differenza effettiva. Se nessuna tabella viene specificata, il campo verrà rilasciato da tutte le tabelle in cui ricorre.*

Sintassi:

```
Drop field fieldname { , fieldname2 ...} [from tablename1 { , tablename2
...}]
Drop fields fieldname { , fieldname2 ...} [from tablename1 { , tablename2
...}]
```

Esempi:

```
Drop field A;
Drop fields A,B;
Drop field A from X;
Drop fields A,B from X,Y;
```

Drop table

Durante l'esecuzione dello script, è possibile rilasciare dal modello dati, e quindi dalla memoria, una o più tabelle interne di Qlik Sense utilizzando l'istruzione **drop table**.

Sintassi:

```
drop table  tablename {, tablename2 ...}
drop tables tablename {, tablename2 ...}
```

*I formati **drop table** e **drop tables** sono entrambi accettati.*

Di conseguenza i seguenti elementi verranno persi:

- La/e tabella/e attuale/i.
- Tutti i campi che non fanno parte delle tabelle rimanenti.
- I valori dei rimanenti campi, provenienti esclusivamente dalla tabella(e) scaricata.

Esempi e risultati:

Esempio	Risultato
`drop table Orders, Salesmen, T456a;`	Questa riga causa la rimozione dalla memoria di tre tabelle.

`Tab1:`
`Load * Inline [`
`Customer, Items, UnitPrice`
`Bob, 5, 1.50`
`];`
 	
`Tab2:`
`LOAD Customer, Sum( Items * UnitPrice ) as Sales`
`resident Tab1`
`group by Customer;`
 	
`drop table Tab1;`
	Una volta creata la tabella *Tab2*, la tabella *Tab1* viene rimossa.

Drop table

Durante l'esecuzione dello script, è possibile rilasciare dal modello dati, e quindi dalla memoria, una o più tabelle interne di Qlik Sense utilizzando l'istruzione **drop table**.

Sintassi:

```
drop table  tablename {, tablename2 ...}
drop tables tablename {, tablename2 ...}
```

*I formati **drop table** e **drop tables** sono entrambi accettati.*

Di conseguenza i seguenti elementi verranno persi:

- La/e tabella/e attuale/i.
- Tutti i campi che non fanno parte delle tabelle rimanenti.
- I valori dei rimanenti campi, provenienti esclusivamente dalla tabella(e) scaricata.

Esempi e risultati:

Esempio	Risultato
`drop table Orders, Salesmen, T456a;`	Questa riga causa la rimozione dalla memoria di tre tabelle.

Esempio	Risultato
Tab1: Load * Inline [Customer, Items, UnitPrice Bob, 5, 1.50]; Tab2: LOAD Customer, Sum(Items * UnitPrice) as Sales resident Tab1 group by Customer; drop table Tab1;	Una volta creata la tabella *Tab2*, la tabella *Tab1* viene rimossa.

Execute

L'istruzione **Execute** viene utilizzata per eseguire altri programmi, mentre Qlik Sense sta caricando i dati. Ad esempio, per effettuare le connessioni necessarie.

Questa funzionalità non è disponibile in Qlik Sense SaaS.

Questa istruzione non è supportata in modalità standard.

Sintassi:

```
execute commandline
```

Argomenti:

Argomenti

Argomento	Descrizione
commandline	Un testo che può essere interpretato dal sistema operativo come riga di comando. È possibile fare riferimento a un percorso del file assoluto o a un percorso della cartella lib://.

Se si desidera utilizzare **Execute**, devono essere soddisfatte le condizioni seguenti:

- L'esecuzione deve essere in modalità legacy (applicabile per Qlik Sense e Qlik Sense Desktop).
- OverrideScriptSecurity deve essere impostato su 1 in *Settings.ini* (applicabile per Qlik Sense). *Settings.ini* si trova nel percorso *C:\ProgramData\Qlik\Sense\Engine* ed è generalmente un file vuoto.

*Se si imposta OverrideScriptSecurity per abilitare **Execute**, qualsiasi utente potrà eseguire i file sul server. Ad esempio, un utente può allegare un file eseguibile a un'app e quindi eseguire il file nello script di caricamento dei dati.*

Procedere come indicato di seguito:

1. Eseguire una copia di *Settings.ini* e aprirla in un editor di testo.
2. Verificare che il file includa *[Settings 7]* nella prima riga.
3. Inserire una nuova riga e digitare *OverrideScriptSecurity=1*.
4. Inserire una riga vuota alla fine del file.
5. Salvare il file.
6. Sostituire *Settings.ini* con il file modificato.
7. Riavviare Qlik Sense Engine Service (QES).

Se Qlik Sense viene eseguito come servizio, alcuni comandi potrebbero funzionare in modo imprevisto.

Esempio:

```
Execute C:\Program Files\Office12\Excel.exe;

Execute lib://win\notepad.exe // win is a folder connection referring to c:\windows
```

Field/Fields

Le parole chiave dello script **Field** e **Fields** vengono utilizzate nelle istruzioni **Declare**, **Derive**, **Drop**, **Comment**, **Rename** e **Tag/Untag**.

FlushLog

L'istruzione **FlushLog** obbliga Qlik Sense a scrivere il contenuto del buffer dello script nel file di registro dello script.

Sintassi:

```
FlushLog
```

Il contenuto del buffer viene inserito nel file di registro. Questo comando può risultare utile ai fini di debug, in quanto si riceveranno i dati che altrimenti andrebbero persi in un'esecuzione dello script non riuscita.

Esempio:

```
FlushLog;
```

Force

L'istruzione **force** impone a Qlik Sense di interpretare i nomi e i valori di campo delle istruzioni **LOAD** e **SELECT** successive in formato solo maiuscolo, solo minuscolo, sempre maiuscolo o conformemente alla visualizzazione attuale (formato misto). Questa istruzione permette di associare i valori di campo provenienti da tabelle create in base a convenzioni differenti.

Sintassi:

```
Force ( capitalization | case upper | case lower | case mixed )
```

Se non si specifica alcun valore, viene applicato il formato misto. L'istruzione force rimane valida finché non si specifica una nuova istruzione force.

L'istruzione **force** non ha alcun effetto sulla sezione relativa al controllo degli accessi: per tutti i valori di campo caricati la distinzione tra maiuscole e minuscole non viene rispettata.

Esempi e risultati

Esempio	Risultato

Questo esempio mostra come imporre l'utilizzo delle maiuscole.
 	
`FORCE Capitalization;`
`Capitalization:`
`LOAD * Inline [`
`ab`
`Cd`
`eF`
`GH`
`];`
	La tabella **Capitalization** contiene i seguenti valori:
	
	`Ab`
	`Cd`
	`Ef`
	`Gh`
	Tutti i valori sono scritti in lettere maiuscole.

Questo esempio mostra come imporre l'utilizzo dei caratteri maiuscoli.
 	
`FORCE Case Upper;`
`CaseUpper:`
`LOAD * Inline [`
`ab`
`Cd`
`eF`
`GH`
`];`
	La tabella **CaseUpper** contiene i seguenti valori:
	
	`AB`
	`CD`
	`EF`
	`GH`
	Tutti i valori sono scritti con caratteri maiuscoli.

Questo esempio mostra come imporre l'utilizzo dei caratteri minuscoli.
 	
`FORCE Case Lower;`
`CaseLower:`
`LOAD * Inline [`
`ab`
`Cd`
`eF`
`GH`
`];`
	La tabella **CaseLower** contiene i seguenti valori:
	
	`ab`
	`cd`
	`ef`
	`gh`
	Tutti i valori sono scritti con caratteri minuscoli.

Esempio	Risultato

Questo esempio mostra come imporre l'utilizzo del formato misto.
 	
`FORCE Case Mixed;`
`CaseMixed:`
`LOAD * Inline [`
`ab`
`Cd`
`eF`
`GH`
`];`
	La tabella **CaseMixed** contiene i seguenti valori:
	
	`ab`
	`Cd`
	`eF`
	`GH`
	Tutti i valori sono scritti in base alla visualizzazione attuale.

Vedere anche:

From

La parola chiave dello script **From** viene utilizzata nelle istruzioni **Load** per fare riferimento a un file e nelle istruzioni **Select** per fare riferimento a una tabella o a una vista del database.

Load

L'istruzione **LOAD** carica i campi da un file, dai dati definiti nello script, da una tabella caricata in precedenza, da una pagina Web, dal risultato di un'istruzione **SELECT** seguente o dalla generazione automatica di dati. È anche possibile caricare dati da connessioni di analisi.

Sintassi:

```
LOAD [ distinct ] fieldlist
[( from file [ format-spec ] |
from_field fieldassource [format-spec]|
inline data [ format-spec ] |
resident table-label |
autogenerate size ) |extension pluginname.functionname([script]
tabledescription)]
[ where criterion | while criterion ]
[ group by groupbyfieldlist ]
[order by orderbyfieldlist ]
```

Argomenti:

Argomenti

Argomento	Descrizione
distinct	È possibile utilizzare **distinct** come predicato se si desidera caricare solo record univoci. Se ci sono record duplicati, verrà caricata la prima istanza. Se si utilizza l'istruzione preceding LOAD, è necessario inserire **distinct** nella prima istruzione LOAD, dal momento che **distinct** ha effetto solo sulla tabella di destinazione.

Argomento	Descrizione
fieldlist	*fieldlist ::= (* | field {, * | field })* Un elenco di campi da caricare. L'utilizzo del carattere * come elenco dei campi indica tutti i campi della tabella. *field ::= (fieldref | expression)* [**as** *aliasname]* La definizione di campo deve sempre contenere un valore letterale, un riferimento a un campo esistente o un'espressione. *fieldref ::= (fieldname |@fieldnumber |@startpos:endpos [* ***I*** *|* ***U*** *|* ***R*** *|* ***B*** | **T**]) *fieldname* è un testo che è identico al nome di un campo nella tabella. Tenere presente che il nome di campo deve essere racchiuso da virgolette doppie o parentesi quadre se, ad esempio, contiene spazi. Talvolta i nomi dei campi non sono disponibili in modo esplicito. Verrà quindi utilizzata una notazione differente: *@fieldnumber* rappresenta il numero di campo di un file tabellare delimitato. Deve essere un intero positivo preceduto da "@". La numerazione viene sempre eseguita partendo da 1 fino al numero di campi presenti. *@startpos*:*endpos* rappresenta le posizioni di inizio e di fine di un campo in un file con record a lunghezza fissa. Le posizioni devono essere entrambe numeri interi positivi. I due valori numerici devono essere preceduti dal simbolo "@" e separati da due punti. La numerazione viene sempre eseguita partendo da 1 fino al numero di posizioni presenti. Nell'ultimo campo **n** viene utilizzato come posizione finale. • Se *@startpos*:*endpos* è seguito immediatamente dal carattere **I** o **U**, i byte letti verranno interpretati come un valore intero binario con segno (**I**) o senza segno (**U**), in base all'ordine dei byte Intel. Il numero di posizioni lette deve essere 1, 2 o 4. • Se *@startpos*:*endpos* è immediatamente seguito dal carattere **R**, la lettura dei byte verrà interpretata come un numero binario reale (a virgola mobile a 32 bit o a 64 bit conforme allo standard IEEE). Il numero di posizioni lette deve essere 4 o 8. • Se *@startpos*:*endpos* è immediatamente seguito dal carattere **B**, la lettura dei byte sarà interpretata come numeri BCD (Binary Coded Decimal), in base allo standard COMP-3. Può essere specificato un numero di byte qualsiasi. *expression* può essere una funzione numerica o una funzione stringa basata su uno o molti altri campi della stessa tabella. Per ulteriori informazioni, vedere la sintassi delle espressioni. **as** viene utilizzato per assegnare un nuovo nome al campo.

Argomento	Descrizione
from	**from** viene utilizzato se i dati devono essere caricati da un file mediante una connessione dati di una cartella o di un file Web. *file ::= [path] filename* **Esempio:** ***'lib://Table Files/'*** Se tale percorso viene omesso, Qlik Sense ricerca il file nella directory specificata dall'istruzione **Directory**. Se non è presente alcuna istruzione **Directory**, Qlik Sense eseguirà la ricerca nella directory di lavoro, *C:\Users\{user}\Documents\Qlik\Sense\Apps*. *In un'installazione server di Qlik Sense, la directory di lavoro è specificata in Qlik Sense Repository Service e, per impostazione predefinita, è C:\ProgramData\Qlik\Sense\Apps.* *filename* può contenere caratteri speciali nello standard DOS (* e ?). Tutti i file corrispondenti verranno caricati nella directory specificata. *format-spec ::= (fspec-item { , fspec-item })* La specifica del formato è costituita da un elenco di più voci di specifica del formato, racchiuse tra parentesi. **Modalità di script legacy** Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** ***c:\data*** • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** ***data*** • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio:** ***http://www.qlik.com***

Argomento	Descrizione
from_field	**from_field** viene utilizzato nel caso in cui i dati devono essere caricati da un campo caricato in precedenza. *fieldassource::=*(*tablename, fieldname*) Il campo è il nome del valore di *tablename* e *fieldname* caricato in precedenza. *format-spec ::= (fspec-item {, fspec-item })* La specifica del formato è costituita da un elenco di più voci di specifica del formato, racchiuse tra parentesi.
inline	**inline** viene utilizzato quando i dati devono essere immessi direttamente nello script e non caricati da un file. *data ::= [text]* I dati immessi mediante una clausola **inline** devono essere racchiusi tra virgolette doppie o tra parentesi quadre. Il testo tra parentesi o virgolette viene interpretato allo stesso modo del contenuto di un file. Pertanto, quando si desidera modificare o immettere una nuova riga in un file di testo, è necessario eseguire questa operazione anche nel testo di una clausola **inline**, ad esempio premendo il tasto INVIO mentre si digita lo script. Il numero di colonne è definito nella prima riga. *format-spec ::= (fspec-item {, fspec-item })* La specifica del formato è costituita da un elenco di più voci di specifica del formato, racchiuse tra parentesi.
resident	**resident** viene usato se i dati devono essere caricati da una tabella caricata in precedenza. *table label* è un'etichetta che precede le istruzioni **LOAD** o **SELECT** che hanno creato la tabella originale. L'etichetta dovrà essere indicata con i due punti finali.
autogenerate	**autogenerate** viene utilizzato se i dati devono essere automaticamente generati da Qlik Sense. *size ::= number* *Number* è un numero intero e indica il numero di record che da generare. L'elenco di campi non deve contenere espressioni che richiedono dati provenienti da una sorgente dati esterna o da una tabella caricata in precedenza, a meno che non si faccia riferimento a un singolo valore di campo in una tabella caricata in precedenza con la funzione **Peek**.

Argomento	Descrizione
extension	È possibile caricare dati da connessioni di analisi. È necessario utilizzare la clausola **extension** per chiamare una funzione definita nel plug-in SSE (Server-Side Extension) o per valutare uno script. È possibile inviare una singola tabella al plug-in SSE, che restituisce una singola tabella di dati. Se il plug-in non specifica i nomi dei campi restituiti, i campi saranno denominati Field1, Field2 e così via. `Extension pluginname.functionname( tabledescription );` • Caricamento di dati mediante una funzione in un plug-in SSE *tabledescription ::= (table { ,tablefield})* Se i campi della tabella non vengono specificati, verranno utilizzati in ordine di caricamento. • Caricamento di dati mediante valutazione di uno script in un plug-in SSE *tabledescription ::= (script, table { ,tablefield})* **Gestione dei tipi di dati nella definizione dei campi della tabella** I tipi di dati sono riconosciuti automaticamente nelle connessioni di analisi. Se i dati non hanno valori numerici e hanno almeno una stringa di testo non NULL, il campo è considerato di tipo testo. In tutti gli altri casi è considerato di tipo numerico. È possibile forzare il tipo di dati racchiudendo il nome del campo in **String()** o **Mixed()**. • **String()** forza il campo in testo. Se il campo è numerico, viene estratta la parte di testo del valore duale, senza eseguire alcuna conversione. • **Mixed()** forza il tipo del campo su duale. **String()** o **Mixed()** non possono essere utilizzati esternamente alle definizioni dei campi della tabella **extension** e non è possibile utilizzare altre funzioni di Qlik Sense nella definizione di un campo della tabella. **Ulteriori informazioni sulle connessioni di analisi** Prima di poter utilizzare le connessioni di analisi, è necessario configurarle.
where	**where** è una clausola utilizzata per dichiarare se un record deve essere incluso o meno nella selezione. La selezione viene inclusa se *criterion* è True. *criterion* è un'espressione logica.
while	**while** è una clausola che specifica se un record deve essere letto ripetutamente. Viene letto lo stesso record finché *criterion* è True. Per risultare utile, una clausola **while** deve generalmente includere la funzione **IterNo( )**. *criterion* è un'espressione logica.

Argomento	Descrizione
group by	**group by** è una clausola usata per definire su quali campi devono essere aggregati (raggruppati) i dati. I campi di aggregazione devono essere inclusi in qualche modo nelle espressioni caricate. Nessun altro campo tranne i campi di aggregazione può essere usato al di fuori delle funzioni di aggregazione nelle espressioni caricate. *groupbyfieldlist ::= (fieldname { ,fieldname })*
order by	**order by** è una clausola utilizzata per ordinare i record di una tabella residente prima che vengano elaborati da un'istruzione **load**. La tabella residente può essere ordinata su uno o più campi, in modo crescente o decrescente. L'ordinamento viene eseguito innanzitutto in base ai valori numerici e secondariamente in base all'ordine di confronto nazionale. Questa clausola può essere utilizzata solamente quando la sorgente dati è una tabella residente. I campi di ordinamento specificano il campo in base al quale viene ordinata la tabella residente. Il campo può essere specificato da un nome o dal suo numero nella tabella residente (il primo campo è il numero 1). orderbyfieldlist ::= fieldname [sortorder] { , fieldname [sortorder] } *sortorder* è *asc* per crescente o *desc* per decrescente. Se *sortorder* non è specificato, viene utilizzato *asc*. *fieldname*, *path*, *filename* e *aliasname* sono stringhe di testo che rappresentano ciò che implica il loro rispettivo nome. Qualsiasi campo presente nella tabella sorgente può essere utilizzato come *fieldname*. Tuttavia, i campi creati mediante la clausola as (*aliasname*) non appartengono all'ambito e non possono essere utilizzati all'interno della stessa istruzione **load**.

Se non viene specificata alcuna sorgente dati mediante una clausola **from**, **inline**, **resident**, **from_field, extension** o **autogenerate** , i dati verranno caricati dal risultato dell'istruzione **SELECT** o **LOAD** immediatamente successiva. L'istruzione successiva non dovrà avere un prefisso.

Esempi:

Caricamento di formati di file differenti
Caricare un file di dati delimitato con le opzioni predefinite:

```
LOAD * from data1.csv;
```

Caricare un file di dati delimitato da una connessione della libreria (DataFiles):

```
LOAD * from 'lib://DataFiles/data1.csv';
```

Caricare tutti i file di dati delimitati da una connessione della libreria (DataFiles):

```
LOAD * from 'lib://DataFiles/*.csv';
```

Caricare un file delimitato specificando la virgola come delimitatore e con etichette incorporate:

```
LOAD * from 'c:\userfiles\data1.csv' (ansi, txt, delimiter is ',', embedded labels);
```

Caricare un file delimitato specificando la tabulazione come delimitatore e con etichette incorporate:

```
LOAD * from 'c:\userfiles\data2.txt' (ansi, txt, delimiter is '\t', embedded labels);
```

Caricare un file dif con intestazioni incorporate:

```
LOAD * from file2.dif (ansi, dif, embedded labels);
```

Caricare tre campi da un file Record Fixed senza intestazioni:

```
LOAD @1:2 as ID, @3:25 as Name, @57:80 as City from data4.fix (ansi, fix, no labels, header is
0, record is 80);
```

Caricare un file QVX specificando un percorso assoluto:

```
LOAD * from C:\qdssamples\xyz.qvx (qvx);
```

Caricamento di file Web

Caricare dall'URL predefinito impostato nella connessione dati del file Web:

```
LOAD * from [lib://MyWebFile];
```

Caricare da un URL specifico e sostituire l'URL impostato nella connessione dati del file Web:

```
LOAD * from [lib://MyWebFile] (URL is 'http://localhost:8000/foo.bar');
```

Caricare da un URL specifico impostato in una variabile mediante espansione con simbolo del dollaro:

```
SET dynamicURL = 'http://localhost/foo.bar';
LOAD * from [lib://MyWebFile] (URL is '$(dynamicURL)');
```

Selezione di alcuni campi, ridenominazione e calcolo dei campi

Caricare solo tre campi specifici da un file delimitato:

```
LOAD FirstName, LastName, Number from data1.csv;
```

Assegnare il nome A al primo campo e il nome B al secondo campo durante il caricamento di un file senza etichette:

```
LOAD @1 as A, @2 as B from data3.txt (ansi, txt, delimiter is '\t', no labels);
```

Caricare Name come concatenazione di FirstName, uno spazio e LastName:

```
LOAD FirstName&' '&LastName as Name from data1.csv;
```

Caricare Quantity, Price e Value (il prodotto di Quantity e Price):

```
LOAD Quantity, Price, Quantity*Price as Value from data1.csv;
```

Selezione di alcuni record

Caricare solo i record univoci mentre i record duplicati verranno eliminati:

```
LOAD distinct FirstName, LastName, Number from data1.csv;
```

Caricare solo i record in cui il campo Litres presenta un valore superiore a zero:

```
LOAD * from Consumption.csv where Litres>0;
```

Caricamento di dati non presenti nel file e di dati generati automaticamente
Caricare una tabella con dati inline, due campi denominati CatID e Category:

```
LOAD * Inline
[CatID, Category
0,Regular
1,Occasional
2,Permanent];
```

Caricare una tabella con dati inline, tre campi denominati UserID, Password e Access:

```
LOAD * Inline [UserID, Password, Access
A, ABC456, User
B, VIP789, Admin];
```

Caricare una tabella con 10.000 righe. Il campo A conterrà il numero del record di lettura (1,2,3,4,5...), mentre il campo B conterrà un numero casuale compreso tra 0 e 1:

```
LOAD RecNo( ) as A, rand( ) as B autogenerate(10000);
```

La parentesi dopo autogenerate è consentita, ma non obbligatoria.

Caricamento di dati da una tabella caricata in precedenza
Come prima cosa carichiamo un file tabella delimitato assegnandogli il nome tab1:

```
tab1:
SELECT A,B,C,D from 'lib://DataFiles/data1.csv';
```

Caricare i campi dalla tabella tab1 già caricata come tab2:

```
tab2:
LOAD A,B,month(C),A*B+D as E resident tab1;
```

Caricare i campi dalla tabella già caricata tab1, ma solo record in cui A è maggiore di B:

```
tab3:
LOAD A,A+B+C resident tab1 where A>B;
```

Caricare i campi dalla tabella già caricata tab1 ordinata in base a A:

```
LOAD A,B*C as E resident tab1 order by A;
```

Caricare campi dalla tabella tab1 già caricata, ordinata in base al primo campo, quindi in base al secondo campo:

```
LOAD A,B*C as E resident tab1 order by 1,2;
```

Caricare i campi dalla tabella tab1 già caricata, ordinata in ordine decrescente in base a C, quindi in ordine crescente in base a B e quindi in base al primo campo in ordine decrescente:

```
LOAD A,B*C as E resident tab1 order by C desc, B asc, 1 desc;
```

Caricamento dei dati da campi caricati in precedenza
Caricare il campo Types da tabelle caricate in precedenza Characters come A:

```
LOAD A from_field (Characters, Types);
```

Caricamento dei dati da una tabella successiva (precedente il caricamento)
Caricare A, B e calcolare i campi X e Y da Table1 caricata nell'istruzione **SELECT** successiva:

```
LOAD A, B, if(C>0,'positive','negative') as X, weekday(D) as Y;
SELECT A,B,C,D from Table1;
```

Raggruppamento di dati
Caricare i campi raggruppati (aggregati) in base a ArtNo:

```
LOAD ArtNo, round(Sum(TransAmount),0.05) as ArtNoTotal from table.csv group by ArtNo;
```

Caricare i campi raggruppati (aggregati) in base a Week e ArtNo:

```
LOAD Week, ArtNo, round(Avg(TransAmount),0.05) as WeekArtNoAverages from table.csv group by
Week, ArtNo;
```

Lettura ripetuta di un record
In questo esempio è presente un file di input Grades.csv contenente i voti per ciascuno studente raccolti in un singolo campo:

```
Student,Grades
Mike,5234
John,3345
Pete,1234
Paul,3352
```

I voti, in una scala da 1 a 5 , rappresentano le materie Math, English, Science e History. È possibile suddividere i voti in valori separati leggendo i record più volte con una clausola **while**, utilizzando la funzione **IterNo( )** come contatore. Durante ciascuna lettura, il voto viene estratto con la funzione **Mid** e memorizzato in Grade, mentre la materia viene selezionata utilizzando la funzione **pick** e memorizzata in Subject. La clausola finale **while** contiene il test per verificare che siano stati letti tutti i voti (in questo caso quattro per studente), il che significa che deve essere letto il record dello studente successivo.

```
MyTab:
LOAD Student,
mid(Grades,IterNo( ),1) as Grade,
pick(IterNo( ), 'Math', 'English', 'Science', 'History') as Subject from Grades.csv
while IsNum(mid(Grades,IterNo(),1));
```

Il risultato è una tabella contenente i seguenti dati:

Student	Subject	Grade
John	English	3
John	History	5
John	Math	3
John	Science	4
Mike	English	2
Mike	History	4
Mike	Math	5
Mike	Science	3
Paul	English	3
Paul	History	2
Paul	Math	3
Paul	Science	5
Pete	English	2
Pete	History	4
Pete	Math	1
Pete	Science	3

Caricamento da connessioni di analisi
Sono utilizzati i dati campione seguenti.

```
Values:
Load
 Rand() as A,
 Rand() as B,
 Rand() as C
AutoGenerate(50);
```

Caricamento di dati mediante una funzione

In questi esempi si suppone di avere un plug-in di connessione di analisi denominato *P* contenente una funzione personalizzata *Calculate(Parameter1, Parameter2)*. La funzione restituisce la tabella *Results* contenente i campi *Field1* e *Field2*.

```
Load * Extension P.Calculate( Values{A, C} );
```
Carica tutti i campi restituiti quando si inviano i campi A e C alla funzione.

```
Load Field1 Extension P.Calculate( Values{A, C} );
```
Carica solo il campo Field1 quando si inviano i campi A e C alla funzione.

```
Load * Extension P.Calculate( Values );
```
Carica tutti i campi restituiti quando si inviano i campi A e B alla funzione. Dal momento che i campi non sono specificati, vengono utilizzati A e B, ossia i primi nell'ordine nella tabella.

```
Load * Extension P.Calculate( Values {C, C});
```
Carica tutti i campi restituiti quando si invia il campo C a entrambi i parametri della funzione.

```
Load * Extension P.Calculate( Values {String(A), Mixed(B)});
```
Carica tutti i campi restituiti quando si invia il campo A forzato come stringa e il campo B forzato come numerico alla funzione.

Caricamento di dati mediante valutazione di uno script

`Load A as A_echo, B as B_echo Extension R.ScriptEval( 'q;', Values{A, B} );`
Carica la tabella restituita dallo script q quando si inviano i valori di A e B.

`Load * Extension R.ScriptEval( '$(My_R_Script)', Values{A, B} );`
Carica la tabella restituita dallo script memorizzata nella variabile My_R_Script quando si inviano i valori di A e B.

`Load * Extension R.ScriptEval( '$(My_R_Script)', Values{B as D, *} );`
Carica la tabella restituita dallo script memorizzato nella variabile My_R_Script quando si inviano i valori di B rinominato in D, A e C. L'uso dell'asterisco (*) consente di inviare i rimanenti campi senza riferimento.

All'estensione file delle connessioni DataFiles si applica la distinzione tra maiuscole e minuscole. Ad esempio: .qvd.

Voci per la specifica del formato

Ogni voce di specifica del formato definisce una determinata proprietà del file tabellare:

`fspec-item ::=` [**ansi | oem | mac | UTF-8 | Unicode | txt | fix | dif | biff | ooxml | html | xml | kml | qvd | qvx delimiter is** `char` | **no eof | embedded labels | explicit labels | no labels | table is [**`tablename`**]** | **header is** `n` | **header is** `line` | **header is** `n` **lines** | **comment is** `string` | **record is** `n` | **record is** `line` | **record is** `n` **lines** | **no quotes |msq** | **URL is** `string` | **userAgent is** `string`]

Set di caratteri

Il set di caratteri e un identificatore di file per l'istruzione **LOAD** che definisce il set di caratteri utilizzato nel file.

Gli identificatori **ansi**, **oem** e **mac** venivano utilizzati in QlikView e sono ancora validi. Tuttavia, non verranno generati quando si crea l'istruzione **LOAD** con Qlik Sense.

Sintassi:

```
utf8 | unicode | ansi | oem | mac | codepage is
```

Argomenti:

Argomenti

Argomento	Descrizione
utf8	Set di caratteri UTF-8
unicode	Set di caratteri Unicode
ansi	Windows, codice pagina 1252
oem	DOS, OS/2, AS400 e altri
mac	Codice pagina 10000

Argomento	Descrizione
codepage is	Con l'identificatore **codepage** è possibile utilizzare qualsiasi codice pagina Windows come *N*.

Limiti:

La conversione dal set di caratteri **oem** non viene implementata per MacOS. Se non si effettua alcuna selezione, in Windows verrà utilizzato il codice pagina 1252.

Esempio:

```
LOAD * from a.txt (utf8, txt, delimiter is ',' , embedded labels)
LOAD * from a.txt (unicode, txt, delimiter is ',' , embedded labels)
LOAD * from a.txt (codepage is 10000, txt, delimiter is ',' , no labels)
```

Vedere anche:

p *Load (page 155)*

Formato delle tabelle

Il formato delle tabelle è un identificatore di file per l'istruzione **LOAD** che definisce il tipo di file. Se non si specifica alcun valore, viene utilizzato un file *.txt*.

Tipi di formato tabella

Tipo	Descrizione
txt	In un file di testo delimitato, le colonne nella tabella sono separate da un carattere di delimitazione.
fix	In un file Record Fixed, ogni campo è costituito da un numero fisso di caratteri. Generalmente, molti file di lunghezza record fissa contengono record divisi da un carattere di separazione, tuttavia sono disponibili opzioni più avanzate per specificare le dimensioni dei record in byte o per includere più di una riga con **Record is**. *Se i dati contengono caratteri a più byte, le interruzioni di campo possono disallinearsi dato che il formato si basa su una lunghezza di byte fissa.*
dif	In un file *.dif* (Data Interchange Format), viene utilizzato un formato speciale per la definizione della tabella.
biff	Qlik Sense è inoltre in grado di interpretare i dati nei file Excel standard mediante il formato *biff* (Binary Interchange File Format).

Tipo	Descrizione
ooxml	Excel 2007 e versioni successive utilizzano il formato ooxml *.xslx*.
html	Se la tabella fa parte di una pagina o un file html, occorre utilizzare html.
xml	xml (Extensible Markup Language) è un linguaggio di markup comune utilizzato per rappresentare le strutture dei dati in un formato testuale.
qvd	Il formato *qvd* è il formato di file QVD proprietario, esportato da un'app Qlik Sense.
qvx	*qvx* è un formato file/flusso di dati per output a prestazioni elevate in Qlik Sense.

Delimiter is

Per i file tabellari delimitati, è possibile specificare un delimitatore arbitrario utilizzando l'identificatore **delimiter is**. Questo identificatore è pertinente solo per i file .txt delimitati.

Sintassi:

```
delimiter is char
```

Argomenti:

Argomenti

Argomento	Descrizione
char	Specifica un singolo carattere dai caratteri 127 ASCII.

Inoltre, possono essere utilizzati i seguenti valori:

Valori opzionali

Valore	Descrizione
'\t'	che rappresenta un segno di tabulazione, con o senza virgolette.
'\\'	che rappresenta un carattere di barra rovesciata (\ ).
'spaces'	che rappresenta tutte le combinazioni di uno o più spazi. I caratteri non stampabili con codice ASCII inferiore a 32, ad eccezione di CR e LF, verranno interpretati come spazi.

Se non viene specificato alcun valore, viene utilizzato **delimiter is ','**.

Esempio:

```
LOAD * from a.txt (utf8, txt, delimiter is ',' , embedded labels);
```

Vedere anche:

p *Load (page 155)*

No eof

L'identificatore **no eof** serve ad ignorare il carattere di fine file quando si caricano i file **.txt** delimitati.

Sintassi:

```
no eof
```

Se si utilizza l'identificatore **no eof**, i caratteri con punto di codice 26, che altrimenti denota la fine del file, vengono ignorati e possono fare parte di un valore di campo.

Questo identificatore risulta pertinente solo per i file di testo delimitati.

Esempio:

```
LOAD * from a.txt (txt, utf8, embedded labels, delimiter is ' ', no eof);
```

Vedere anche:

p *Load (page 155)*

Labels

Labels è un identificatore di file per l'istruzione **LOAD** che consente di individuare i nomi dei campi all'interno di un file.

Sintassi:

```
embedded labels|explicit labels|no labels
```

I nomi di campo possono essere collocati in posizioni differenti del file. Se il primo record contiene i nomi di campo, è consigliabile utilizzare **embedded labels**. Se non vi sono nomi di campo, è necessario utilizzare **no labels**. Talvolta, nei file *dif* viene utilizzata una sezione di intestazione separata con nomi di campo espliciti. In questi casi si consiglia di utilizzare **explicit labels**. Se non si è specificato alcun elemento, viene utilizzato **embedded labels** anche per i file *dif*.

Example 1:

```
LOAD * from a.txt (unicode, txt, delimiter is ',' , embedded labels
```

Example 2:

```
LOAD * from a.txt (codePage is 1252, txt, delimiter is ',' , no labels)
```

Vedere anche:

p *Load (page 155)*

Header is

Specifica la dimensione dell'intestazione nei file tabellari. È possibile specificare una lunghezza arbitraria per l'intestazione mediante l'identificatore **header is**. Un'intestazione è una sezione di testo non utilizzata da Qlik Sense.

Sintassi:

```
header is n
header is line
header is n lines
```

La lunghezza dell'intestazione può essere espressa in byte (**header is n**) o in linee (**header is line** o **header is n lines**). **n** deve essere un numero intero, che rappresenti la lunghezza dell'intestazione. Se non specificato, viene utilizzato **header is 0**. L'identificatore **header is** è pertinente solo per i file tabellari.

Esempio:

Questo è un esempio di tabella della sorgente dati contenente una riga di testo di intestazione che Qlik Sense non deve interpretare come dati.

```
*Header line
Col1,Col2
a,B
c,D
```

Utilizzando l'identificatore **header is 1 lines**, la prima riga non verrà caricata come dati. Nell'esempio, l'identificatore **embedded labels** indica a Qlik Sense di interpretare la prima riga non esclusa come contenente etichette di campo.

```
LOAD Col1, Col2
FROM 'lib://files/header.txt'
(txt, embedded labels, delimiter is ',', msq, header is 1 lines);
```

Il risultato è una tabella con due campi Col1 e Col2.

Vedere anche:

p *Load (page 155)*

Record is

Per i file Record Fixed, la lunghezza del record deve essere specificata tramite l'identificatore **record is**.

Sintassi:

```
Record is n
Record is line
Record is n lines
```

Argomenti:

Argomenti

Argomento	Descrizione
n	Specifica la lunghezza del record in byte.
line	Specifica la lunghezza del record come riga singola.
n lines	Specifica la lunghezza del record in righe, dove n è un numero intero positivo che rappresenta la lunghezza del record.

Limiti:

L'identificatore **record is** è pertinente solo per i file **fix**.

Vedere anche:

p *Load (page 155)*

Quotes

Quotes è un identificatore di file per l'istruzione **LOAD** che stabilisce se è possibile utilizzare le virgolette e la precedenza tra virgolette e separatori. Solo per file di testo.

Sintassi:

```
no quotes
msq
```

Se l'identificatore viene omesso, è possibile utilizzare le virgolette standard, ad esempio le virgolette " " o ' ' ma solo se sono il primo e l'ultimo carattere non vuoto di un valore di campo.

Argomenti:

Argomenti

Argomento	Descrizione
no quotes	Viene utilizzato solo se le virgolette non sono accettate in un file di testo.
msq	Viene utilizzato per specificare virgolette in stile moderno che consentono contenuti multiriga nei campi. I campi che contengono caratteri di fine riga devono essere racchiusi tra virgolette doppie. Un limite dell'opzione msq è che i singoli caratteri di virgoletta doppia (") che compaiono come primo o ultimo carattere nel contenuto di un campo vengono interpretati come segno iniziale o finale del contenuto multiriga. Ciò potrebbe causare risultati imprevisti nella serie di dati caricata. In questo caso, utilizzare le virgolette standard omettendo l'identificatore.

XML

Questo identificatore di script viene utilizzato per il caricamento dei file xml. Le opzioni valide per l'identificatore **XML** sono elencate nella sintassi.

Non è possibile caricare file DTD in Qlik Sense.

Sintassi:

```
xmlsimple
```

Vedere anche:

p *Load (page 155)*

KML

Questo identificatore di script viene utilizzato durante il caricamento dei file KML da utilizzare in una visualizzazione della mappa.

Sintassi:

```
kml
```

Il file KML può rappresentare i dati di un'area (ad esempio paesi o regioni) rappresentati da poligoni, dati di linee (ad esempio binari o strade) o dati di punti (ad esempio città o luoghi) rappresentati da punti nella forma [long, lat].

URL is

Questo identificatore di script è utilizzato per impostare l'URL di una connessione dati a un file Web quando si carica un file Web.

Sintassi:

```
URL is string
```

Argomenti:

Argomenti

Argomento	Descrizione
string	Specifica l'URL del file da caricare. Questo valore sostituirà l'URL impostato nella connessione al file Web utilizzata.

Limiti:

L'identificatore **URL is** è rilevante solo per i file Web. È necessario utilizzare una connessione dati al file Web esistente.

Vedere anche:

p *Load (page 155)*

userAgent is

Questo identificatore di script è utilizzato per impostare l'agente utente del browser quando si carica un file Web.

Sintassi:

```
userAgent is string
```

Argomenti:

Argomenti

Argomento	Descrizione
string	Specifica la stringa dell'agente utente del browser. Questo valore sostituirà l'agente utente del browser predefinito, "Mozilla/5.0".

Limiti:

L'identificatore **userAgent is** è rilevante solo per i file Web.

Vedere anche:

p *Load (page 155)*

Let

L'istruzione **let** è un complemento all'istruzione **set**, utilizzata per definire le variabili degli script. L'istruzione **let**, a differenza dell'istruzione **set**, valuta l'espressione posta sul lato destro del simbolo '=' al tempo di esecuzione dello script prima dell'assegnazione alla variabile.

Sintassi:

```
Let variablename=expression
```

Esempi e risultati:

Esempio	Risultato

`Set x=3+4;`
 	
`Let y=3+4;`
 	
`z=$(y)+1;`
	`$(x)` verrà valutato come ' 3+4 '
	
	`$(y)` verrà valutato come ' 7 '
	
	`$(z)` verrà valutato come ' 8 '
	
	Notare la differenza tra le istruzioni **Set** e **Let**. L'istruzione **Set** assegna la stringa '3+4' alla variabile, mentre l'istruzione **Let** valuta la stringa e assegna 7 alla variabile.

Let T=now( );	`$(T)` verrà restituito il valore dell'ora attuale.

Loosen Table

Una o più tabelle dati interne di Qlik Sense possono essere dichiarate logicamente disconnesse in modo esplicito durante l'esecuzione dello script, utilizzando un'istruzione **Loosen Table**. Quando una tabella è logicamente disconnessa, tutte le associazioni tra i valori di campo nella tabella vengono rimosse. È possibile ottenere un effetto simile caricando ogni campo della tabella logicamente disconnessa come tabelle indipendenti e scollegate. La disconnessione logica può rivelarsi utile durante il controllo per isolare temporaneamente parti differenti della struttura dei dati. Nel visualizzatore tabelle è possibile riconoscere una tabella logicamente disconnessa dalle linee punteggiate. L'utilizzo di una o più istruzioni **Loosen Table** nello script indica a Qlik Sense di ignorare ogni impostazione di tabelle logicamente disconnesse effettuata prima dell'esecuzione dello script.

Sintassi:

```
Loosen Tabletablename [ , tablename2 ...]
Loosen Tablestablename [ , tablename2 ...]
```

È possibile utilizzare la sintassi: **Loosen Table** o **Loosen Tables**.

*Se Qlik Sense dovesse individuare riferimenti circolari nella struttura dei dati che non possono essere interrotti da tabelle logicamente disconnesse in modo interattivo o esplicito nello script, verrà forzata l'impostazione logicamente disconnessa per una o più tabelle aggiuntive finché non verranno eliminati tutti i riferimenti circolari. In questo caso, nella finestra di dialogo relativa all'**avviso di ciclo**, verrà visualizzato un avviso.*

Esempio:

```
Tab1:
SELECT * from Trans;
Loosen Table Tab1;
```

Map

L'istruzione **map ... using** viene usata per eseguire il mapping di un certo valore di campo o una certa espressione sui valori di una tabella di mapping specifica. La tabella di mapping viene creata utilizzando l'istruzione **Mapping**.

Sintassi:

```
Map fieldlist Using  mapname
```

Il mapping automatico viene eseguito per i campi caricati dopo l'istruzione **Map … Using** fino alla fine dello script o finché non viene rilevata un'istruzione **Unmap**.

Il mapping viene eseguito al termine della catena di eventi che conducono alla memorizzazione del campo nella tabella interna in Qlik Sense. Questo significa che le operazioni di mapping non vengono eseguite ogni volta che si incontra un nome di campo come parte di un'espressione, ma più propriamente quando il valore viene salvato nel nome di campo nella tabella interna. Se è richiesto il mapping a livello di espressione, occorre utilizzare la funzione **Applymap()**.

Argomenti:

Argomenti

Argomento	Descrizione
fieldlist	Un elenco separato da virgole dei campi di cui occorre eseguire il mapping da questo punto nello script. L'utilizzo di * per l'elenco dei campi indica tutti i campi. Nei nomi di campo sono consentiti i caratteri speciali * e ?. Se si utilizzano i caratteri speciali può essere necessario delimitare i nomi di campo tra virgolette.
mapname	Il nome di una tabella di mapping letta in precedenza in un'istruzione **mapping load** o **mapping select**.

Esempi e risultati:

Esempio	Risultato
Map Country Using Cmap;	Esegue il mapping del campo Country utilizzando la mappa Cmap.
Map A, B, C Using X;	Esegue il mapping dei campi A, B e C utilizzando la mappa X.
Map * Using GenMap;	Esegue il mapping di tutti i campi utilizzando GenMap.

NullAsNull

L'istruzione **NullAsNull** disattiva la conversione dei valori NULL in valori di stringa impostati in precedenza da un'istruzione **NullAsValue**.

Sintassi:

```
NullAsNull *fieldlist
```

L'istruzione **NullAsValue** funge da interruttore e può essere attivata o disattivata diverse volte all'interno di uno script utilizzando un'istruzione **NullAsValue** o un'istruzione **NullAsNull**.

Argomenti:

Argomenti

Argomento	Descrizione
*fieldlist	Un elenco separato da virgole dei campi per il quale è necessario attivare l'istruzione **NullAsNull**. L'utilizzo di * per l'elenco dei campi indica tutti i campi. Nei nomi di campo sono consentiti i caratteri speciali * e ?. Se si utilizzano i caratteri speciali può essere necessario delimitare i nomi di campo tra virgolette.

Esempio:

```
NullAsNull A,B;
LOAD A,B from x.csv;
```

NullAsValue

L'istruzione **NullAsValue** specifica per quali campi NULL deve essere convertito in un valore.

Sintassi:

```
NullAsValue *fieldlist
```

Per impostazione predefinita, Qlik Sense considera i valori NULL come entità mancati o non definite. Tuttavia, alcuni database considerano i valori NULL valori speciali piuttosto che semplici valori mancanti. È possibile sospendere il divieto di collegamento reciproco dei valori NULL con altri valori NULL mediante l'istruzione **NullAsValue**.

L'istruzione **NullAsValue** funge da interruttore e viene applicata alle istruzioni di caricamento successive. Questa istruzione può essere disattivata di nuovo utilizzando l'istruzione **NullAsNull**.

Argomenti:

Argomenti

Argomento	Descrizione
*fieldlist	Un elenco separato da virgole dei campi per il quale è necessario attivare l'istruzione **NullAsValue**. L'utilizzo di * per l'elenco dei campi indica tutti i campi. Nei nomi di campo sono consentiti i caratteri speciali * e ?. Se si utilizzano i caratteri speciali può essere necessario delimitare i nomi di campo tra virgolette.

Esempio:

```
NullAsValue A,B;
Set NullValue = 'NULL';
LOAD A,B from x.csv;
```

Qualify

L'istruzione **Qualify** consente di modificare la qualificazione dei nomi di campo, ad esempio il nome della tabella dei nomi di campo diventerà un prefisso.

Sintassi:

```
Qualify *fieldlist
```

L'unione automatica dei campi con lo stesso nome in tabelle differenti può essere sospesa con l'utilizzo dell'istruzione **qualify**, che qualifica il nome di campo con il proprio nome di tabella. Se qualificato, il nome campo verrà ridenominato quando verrà trovato in una tabella. Il nuovo nome sarà nel formato *tablename.fieldname*. *Tablename* è equivalente all'etichetta della tabella attuale oppure, se non esiste alcuna etichetta, al nome visualizzato dopo **from** nelle istruzioni **LOAD** e **SELECT**.

La qualificazione viene effettuata per tutti i campi caricati dopo l'istruzione **qualify**

e, per impostazione predefinita, è sempre disattivata all'inizio dell'esecuzione dello script. La qualificazione di un nome di campo può essere attivata in qualsiasi momento utilizzando l'istruzione **qualify**. La qualificazione può essere disattivata in qualsiasi momento utilizzando l'istruzione **Unqualify**.

*L'istruzione **qualify** non deve essere utilizzata insieme al ricaricamento parziale.*

Argomenti:

Argomenti

Argomento	Descrizione
*fieldlist	Un elenco separato da virgola dei campi per i quali è necessario attivare la qualificazione. L'utilizzo di * per l'elenco dei campi indica tutti i campi. Nei nomi di campo sono consentiti i caratteri speciali * e ?. Se si utilizzano i caratteri speciali può essere necessario delimitare i nomi di campo tra virgolette.

Example 1:

```
Qualify B;
LOAD A,B from x.csv;
LOAD A,B from y.csv;
```

Le due tabelle **x.csv** e **y.csv** vengono associate mediante **A**. Tre campi daranno come risultato: A, x.B, y.B.

Example 2:

In un database poco familiare si rivela spesso utile iniziare associando le varie tabelle secondo un unico campo o un numero minore di campi, come illustrato nel seguente esempio:

```
qualify *;
unqualify TransID;
SQL SELECT * from tab1;
SQL SELECT * from tab2;
SQL SELECT * from tab3;
```

Solo il campo **TransID** viene utilizzato per le associazioni tra le tre tabelle *tab1*, *tab2* e *tab3*.

Rem

L'istruzione **rem** viene utilizzata per inserire osservazioni, o commenti, negli script o per disattivare temporaneamente istruzioni dello script senza rimuoverle.

Sintassi:

```
Rem string
```

Tutti i contenuti compresi tra **rem** e il punto e virgola **;** successivo vengono interpretati come un commento.

Per inserire commenti negli script, sono disponibili altri due metodi:

1. È possibile creare un commento in un punto qualsiasi dello script, eccetto tra due virgolette, inserendo la sezione interessata tra **/*** e ***/**.
2. Digitando **//** all'interno dello script, tutto il testo che segue a destra nella stessa riga diventa un commento. (Tenere presente l'eccezione di //: che può far parte di un indirizzo Internet).

Argomenti:

Argomenti

Argomento	Descrizione
string	Un testo arbitrario.

Esempio:

```
Rem ** This is a comment **;
/* This is also a comment */
// This is a comment as well
```

Rename

La parola chiave dello script **Rename** consente di rinominare tabelle o campi già caricati.

Rename field

Questa funzione di script rinomina uno o più campi di Qlik Sense esistenti dopo che sono stati caricati.

Si sconsiglia di denominare una variabile con lo stesso nome utilizzato per un campo o una funzione in Qlik Sense.

È possibile utilizzare la sintassi: **rename field** o **rename fields**.

Sintassi:

```
Rename Field (using mapname | oldname to newname{ , oldname to newname })
Rename Fields (using mapname | oldname to newname{ , oldname to newname })
```

Argomenti:

Argomento	Descrizione
mapname	Il nome di una tabella di mapping caricata in precedenza, contenente una o più coppie di nomi di campo obsoleti e nuovi.
oldname	Il nome di campo obsoleto.
newname	Il nome di campo nuovo.

Limiti:

Non è possibile rinominare due campi in modo che abbiano lo stesso nome.

Example 1:

```
Rename Field XAZ0007 to Sales;
```

Example 2:

```
FieldMap:
Mapping SQL SELECT oldnames, newnames from datadictionary;
Rename Fields using FieldMap;
```

Rename table

Questa funzione di script rinomina una o più tabelle interne di Qlik Sense esistenti dopo che sono state caricate.

È possibile utilizzare la sintassi: **rename table** o **rename tables**.

Sintassi:

```
Rename Table (using mapname | oldname to newname{ , oldname to newname })
Rename Tables (using mapname | oldname to newname{ , oldname to newname })
```

Argomenti:

Argomenti

Argomento	Descrizione
mapname	Il nome di una tabella di mapping caricata in precedenza, contenente una o più coppie di nomi di tabella obsoleti e nuovi.
oldname	Il nome di tabella obsoleto.
newname	Il nome di tabella nuovo.

Limiti:

Non è possibile ridenominare nello stesso modo due tabelle con nomi diversi. Lo script genererà un errore se si tenta di rinominare una tabella con lo stesso nome di una tabella esistente.

Example 1:

```
Tab1:
SELECT * from Trans;
Rename Table Tab1 to Xyz;
```

Example 2:

```
TabMap:
Mapping LOAD oldnames, newnames from tabnames.csv;
Rename Tables using TabMap;
```

Search

L'istruzione **Search** viene utilizzata per includere o escludere campi nella ricerca intelligente.

Sintassi:

```
Search Include *fieldlist
Search Exclude *fieldlist
```

È possibile utilizzare diverse istruzioni Search per affinare la selezione dei campi da includere. Le istruzioni vengono valutate dall'alto verso il basso.

Argomenti:

Argomenti

Argomento	Descrizione
*fieldlist	Un elenco separato da virgole dei campi da includere o escludere dalle ricerche nella ricerca intelligente. L'utilizzo di * per l'elenco dei campi indica tutti i campi. Nei nomi di campo sono consentiti i caratteri speciali * e ?. Se si utilizzano i caratteri speciali può essere necessario delimitare i nomi di campo tra virgolette.

Esempio:

Esempi di ricerche

Istruzione	Descrizione
`Search Include *;`	Consente di includere tutti i campi nelle ricerche della ricerca intelligente.
`Search Exclude [*ID];`	Consente di escludere tutti i campi che terminano con ID dalle ricerche della ricerca intelligente.
`Search Exclude '*ID';`	Consente di escludere tutti i campi che terminano con ID dalle ricerche della ricerca intelligente.
`Search Include ProductID;`	Consente di includere il campo ProductID nelle ricerche della ricerca intelligente.

Il risultato combinato di queste tre istruzioni, in questa sequenza, è che dalle ricerche della ricerca intelligente vengono esclusi tutti i campi che terminano con ID tranne ProductID.

Section

L'istruzione **section** consente di definire se le istruzioni successive **LOAD** e **SELECT** devono essere considerate come dati o come una definizione dei diritti di accesso.

Sintassi:

```
Section (access | application)
```

Se non viene specificato alcun valore, viene utilizzato **section application**. La definizione **section** è valida finché non si specifica una nuova istruzione **section**.

Esempio:

```
Section access;
Section application;
```

Select

La selezione dei campi da una sorgente dati ODBC o da un provider OLE DB viene eseguita utilizzando le istruzioni SQL **SELECT** standard. Tuttavia, l'ambito nel quale le istruzioni **SELECT** vengono accettate dipende dal driver ODBC o dal provider OLE DB utilizzato. L'utilizzo dell'istruzione **SELECT** richiede una connessione dati aperta all'origine.

Sintassi:

```
Select [all | distinct | distinctrow | top n [percent] ] fieldlist

From tablelist

[where criterion ]
```

```
[group by fieldlist [having criterion ] ]

[order by fieldlist [asc | desc] ]

[ (Inner | Left | Right | Full) join tablename on fieldref = fieldref ]
```

Inoltre, talvolta è possibile concatenare diverse istruzioni **SELECT** in un'unica istruzione tramite l'utilizzo dell'operatore **union**:

```
selectstatement Union selectstatement
```

L'istruzione **SELECT** viene interpretata dal driver ODBC o dal provider OLE DB e, pertanto, possono verificarsi deviazioni dalla sintassi generale SQL a seconda delle caratteristiche dei driver ODBC o del provider OLE DB, ad esempio:

- Talvolta **as** non è consentito, vale a dire *aliasname* deve seguire immediatamente *fieldname*.
- Talvolta **as** è obbligatorio se si utilizza un *aliasname*.
- **distinct**, **as**, **where**, **group by**, **order by** o **union** a volte non sono supportati.
- Il driver ODBC talvolta non accetta tutte le diverse virgolette elencate in precedenza.

*La descrizione qui fornita dell'istruzione SQL **SELECT** non è completa. Ad esempio, le istruzioni **SELECT** possono essere nidificate, più unioni possono essere inserite in un'unica istruzione **SELECT**, a volte il numero di funzioni consentito nelle espressioni può essere molto alto e così via.*

Argomenti:

Argomenti

Argomento	Descrizione
distinct	**distinct** è un predicato che viene utilizzato se le combinazioni duplicate dei valori nei campi selezionati devono essere caricate una sola volta.
distinctrow	**distinctrow** è un predicato che viene utilizzato se i record duplicati presenti nella tabella sorgente devono essere caricati una sola volta.

Argomento	Descrizione
fieldlist	**fieldlist ::= (*| field) {, field }** Un elenco dei campi da selezionare. L'utilizzo del simbolo * come elenco dei campi indica tutti i campi della tabella. **fieldlist ::= field {, field }** Un elenco di uno o più campi separati da virgole. **field ::= (fieldref| expression) [as aliasname]** L'espressione può essere, ad esempio, una funzione numerica oppure una funzione di stringa basata su uno o più campi diversi. Tra gli operatori e le funzioni generalmente accettati si annoverano: +, -, *, /, & (concatenazione di stringhe), sum(fieldname), count (fieldname), avg(fieldname)(average), month(fieldname) ecc. Per ulteriori informazioni, consultare la documentazione del driver ODBC. **fieldref ::= [tablename.] fieldname** **tablename** e **fieldname** sono stringhe di testo identiche a ciò che implicano. Se contengono spazi, ad esempio, devono essere incluse fra doppie virgolette diritte. La clausola **as** viene utilizzata per assegnare un nuovo nome al campo.
from	**tablelist ::= table {, table }** Elenco di tabelle da cui vengono selezionati i campi. **table ::= tablename [[as] aliasname]** **tablename** può essere inserito o meno tra virgolette.
where	**where** è una clausola utilizzata per dichiarare se un record deve essere incluso o meno nella selezione. **criterion** è un'espressione logica che a volte può risultare molto complessa. Alcuni degli operatori accettati sono: funzioni e operatori numerici, **=**, **<>** o **#(**diverso da), **>**, **>=**, **<**, **<=**, **and**, **or**, **not**, **exists**, **some**, **all**, **in** e anche le nuove istruzioni **SELECT**. Consultare la documentazione del driver ODBC o del provider OLE DB per ulteriori informazioni.
group by	**group by** è una clausola utilizzata per aggregare (raggruppare) più record in uno solo. All'interno di un gruppo, per un determinato campo, tutti i record devono avere lo stesso valore o il campo può essere utilizzato solo all'interno di un'espressione, ad esempio una somma o una media. L'espressione basata su uno o più campi viene definita nell'espressione del simbolo del campo.
having	**having** è una clausola utilizzata per qualificare i gruppi analogamente al modo in cui la clausola **where** viene utilizzata per qualificare i record.
order by	**order by** è una clausola utilizzata per dichiarare la sequenza di ordinamento della tabella risultante dall'istruzione **SELECT**.
join	**join** è un qualificatore che dichiara se diverse tabelle devono essere unite in una sola. I nomi dei campi e delle tabelle devono essere delimitati da virgolette se contengono spazi vuoti o lettere dell'alfabeto nazionale. Quando lo script viene generato automaticamente da Qlik Sense, le virgolette utilizzate sono quelle preferite dal driver ODBC o dal provider OLE DB specificato nella definizione della sorgente dati nell'istruzione **Connect**.

Example 1:

```
SELECT * FROM `Categories`;
```

Example 2:

```
SELECT `Category ID`, `Category Name` FROM `Categories`;
```

Example 3:

```
SELECT `Order ID`, `Product ID`,

`Unit Price` * Quantity * (1-Discount) as NetSales

FROM `Order Details`;
```

Example 4:

```
SELECT `Order Details`.`Order ID`,

Sum(`Order Details`.`Unit Price` * `Order Details`.Quantity) as `Result`

FROM `Order Details`, Orders

where Orders.`Order ID` = `Order Details`.`Order ID`

group by `Order Details`.`Order ID`;
```

Set

L'istruzione **set** viene utilizzata per definire le variabili di script. Le variabili possono essere utilizzate per sostituire stringhe, percorsi, unità e così via.

Sintassi:

```
Set variablename=string
```

Example 1:

```
Set FileToUse=Data1.csv;
```

Example 2:

```
Set Constant="My string";
```

Example 3:

```
Set BudgetYear=2012;
```

Sleep

L'istruzione **sleep** interrompe l'esecuzione dello script per il periodo di tempo specificato.

Sintassi:

```
Sleep n
```

Argomenti:

Argomento	Descrizione
n	Indicato in millisecondi, in cui *n* rappresenta un numero intero positivo che non supera 3600000 (ad esempio, 1 ora). Il valore può essere un'espressione.

Example 1:

```
Sleep 10000;
```

Example 2:

```
Sleep t*1000;
```

SQL

L'istruzione **SQL** consente di inviare un comando arbitrario SQL tramite una connessione ODBC o OLE DB.

Sintassi:

```
SQL sql_command
```

L'invio di istruzioni SQL che aggiornano il database restituisce un errore se Qlik Sense ha aperto la connessione ODBC in modalità di sola lettura.

La sintassi:

```
SQL SELECT * from tab1;
```

è consentita e, per ragioni di uniformità, rappresenta la sintassi preferita per l'istruzione **SELECT**. Il prefisso SQL rimarrà comunque opzionale per le istruzioni **SELECT**.

Argomenti:

Argomento	Descrizione
sql_command	Un comando SQL valido.

Example 1:

```
SQL leave;
```

Example 2:

```
SQL Execute <storedProc>;
```

SQLColumns

L'istruzione **sqlcolumns** restituisce un gruppo di campi che descrivono le colonne di una sorgente dati ODBC o OLE DB sulla quale è stata eseguita un'istruzione **connect**.

Sintassi:

```
SQLcolumns
```

Questi campi possono essere combinati con i campi generati dai comandi **sqltables** e **sqltypes** per ottenere una visione d'insieme di un determinato database. I dodici campi standard sono:

TABLE_QUALIFIER

TABLE_OWNER

TABLE_NAME

COLUMN_NAME

DATA_TYPE

TYPE_NAME

PRECISION

LENGTH

SCALE

RADIX

NULLABLE

REMARKS

Per una descrizione dettagliata di questi campi, consultare un manuale di riferimento di ODBC.

Esempio:

```
Connect to 'MS Access 7.0 Database; DBQ=C:\Course3\DataSrc\QWT.mbd';
SQLcolumns;
```

Alcuni driver ODBC potrebbero non supportare questo comando. Alcuni driver ODBC potrebbero produrre campi aggiuntivi.

SQLTables

L'istruzione **sqltables** restituisce un gruppo di campi che descrivono le tabelle di una sorgente dati ODBC o OLE DB sulla quale è stata eseguita un'istruzione **connect**.

Sintassi:

```
SQLTables
```

Questi campi possono essere combinati con i campi generati dai comandi **sqlcolumns** e **sqltypes** per ottenere una visione d'insieme di un determinato database. I cinque campi standard sono:

TABLE_QUALIFIER

TABLE_OWNER

TABLE_NAME

TABLE_TYPE

REMARKS

Per una descrizione dettagliata di questi campi, consultare un manuale di riferimento di ODBC.

Esempio:

```
Connect to 'MS Access 7.0 Database; DBQ=C:\Course3\DataSrc\QWT.mbd';
SQLTables;
```

Alcuni driver ODBC potrebbero non supportare questo comando. Alcuni driver ODBC potrebbero produrre campi aggiuntivi.

SQLTypes

L'istruzione **sqltypes** restituisce un gruppo di campi che descrivono i tipi di una sorgente dati ODBC o OLE DB sulla quale è stata eseguita un'istruzione **connect**.

Sintassi:

```
SQLTypes
```

Questi campi possono essere combinati con i campi generati dai comandi **sqlcolumns** e **sqltables** per ottenere una visione d'insieme di un determinato database. I quindici campi standard sono:

TYPE_NAME

DATA_TYPE

PRECISION

LITERAL_PREFIX

LITERAL_SUFFIX

CREATE_PARAMS

NULLABLE

CASE_SENSITIVE

SEARCHABLE

UNSIGNED_ATTRIBUTE

MONEY

AUTO_INCREMENT

LOCAL_TYPE_NAME

MINIMUM_SCALE

MAXIMUM_SCALE

Per una descrizione dettagliata di questi campi, consultare un manuale di riferimento di ODBC.

Esempio:

```
Connect to 'MS Access 7.0 Database; DBQ=C:\Course3\DataSrc\QWT.mbd';
SQLTypes;
```

Alcuni driver ODBC potrebbero non supportare questo comando. Alcuni driver ODBC potrebbero produrre campi aggiuntivi.

Star

La stringa utilizzata per rappresentare l'insieme di tutti i valori di un campo nel database può essere impostata tramite l'istruzione **star**. Interessa le istruzioni **LOAD** e **SELECT** successive.

Sintassi:

```
Star is[ string ]
```

Argomenti:

Argomenti

Argomento	Descrizione
string	Un testo arbitrario. Tenere presente che se la stringa contiene spazi deve essere racchiusa tra virgolette. Se non si specifica alcun valore, viene utilizzato **star is;**, ossia non è disponibile alcun simbolo star finché non viene specificato in maniera esplicita. Questa definizione è valida finché non si specifica una nuova istruzione **star**.

Si sconsiglia l'uso dell'istruzione **Star is** nella parte dei dati dello script (sotto **Section Application**) se si utilizza Section Access. Il carattere asterisco è tuttavia pienamente supportato per i campi protetti nella parte **Section Access** dello script. In questo caso non è necessario utilizzare l'istruzione **Star is** esplicita in quanto è sempre implicita in Section Access.

Limiti

- Non è possibile utilizzare il carattere asterisco con i campi chiave, ovvero i campi che collegano le tabelle.
- Non è possibile utilizzare il carattere asterisco con qualsiasi campo interessato dall'istruzione **Unqualify**, in quanto ciò può avere ripercussioni sui campi che collegano le tabelle.
- Non è possibile utilizzare il carattere asterisco con tabelle non logiche, ad esempio tabelle info-load o tabelle mapping-load.
- Quando viene utilizzato in un campo di riduzione (un campo collegato ai dati) in Section Access, il carattere asterisco rappresenta i valori elencati in questo campo in Section Access. Non rappresenta altri valori che possono esistere nei dati, ma non sono elencati in Section Access.
- Non è possibile utilizzare il carattere asterisco con campi interessati da qualsiasi forma di riduzione dei dati al di fuori dell'area **Section Access**.

Esempio

L'esempio seguente è un estratto di uno script di caricamento dei dati contenente la sezione relativa al controllo degli accessi.

```
Star is *;



Section Access;

LOAD * INLINE [

ACCESS, USERID, OMIT

ADMIN, ADMIN,

USER, USER1, SALES

USER, USER2, WAREHOUSE

USER, USER3, EMPLOYEES

USER, USER4, SALES

USER, USER4, WAREHOUSE

USER, USER5, *

];
```

```
Section Application;

LOAD * INLINE [

SALES, WAREHOUSE, EMPLOYEES, ORDERS

1, 2, 3, 4

];
```

Viene applicato quanto segue:

- Il segno *Star* corrisponde a *.
- L'utente *ADMIN* visualizza tutti i campi. Non viene omesso nulla.
- L'utente *USER1* non può visualizzare il campo *SALES*.
- L'utente *USER2* non può visualizzare il campo *WAREHOUSE* .
- L'utente *USER3* non può visualizzare il campo *EMPLOYEES*.
- L'utente *USER4* è stato aggiunto due volte alla soluzione per omettere (OMIT) due campi relativi a questo utente, *SALES* e *WAREHOUSE*.
- *USER5* presenta un "*" aggiunto che significa che tutti i campi elencati in OMIT sono indisponibili, ovvero, l'utente *USER5* non può vedere i campi *SALES*, *WAREHOUSE* e *EMPLOYEES* ma questo utente può visualizzare il campo *ORDERS*.

Store

L'istruzione **Store** crea un file QVD, o text.

Sintassi:

```
Store [ fieldlist from] table into filename [ format-spec ];
```

L'istruzione creerà un file con nome esplicito QVD o un file di testo.

L'istruzione può esportare campi solo da una tabella dati. Se occorre esportare i campi da più tabelle, un'operazione join esplicita deve essere eseguita precedentemente nello script per la creazione della tabella dati da esportare.

I valori di testo sono esportati nel file CSV nel formato UTF-8. È possibile specificare un delimitatore, vedere **LOAD**. L'istruzione **store** associata a un file CSV non supporta l'esportazione BIFF.

Argomenti:

Argomenti del comando Store

Argomento	Descrizione
*fieldlist::= (* | field) { , field })*	Un elenco dei campi da selezionare. L'utilizzo del carattere * per l'elenco dei campi indica tutti i campi. *field::= fieldname [***as** *aliasname]* *fieldname* è un testo che è identico al nome di campo in *table*. (Tenere presente che il nome di campo deve essere racchiuso da virgolette doppie diritte o parentesi quadre se, ad esempio, contiene spazi o altri caratteri non standard). *aliasname* è un nome alternativo per il campo da utilizzare nel file QVD o CSV risultante.
table	Un'etichetta dello script che rappresenta una tabella già caricata da utilizzare come sorgente dei dati.
filename	Il nome del file di destinazione che include un percorso valido a una connessione dati della cartella esistente. **Esempio: *'lib://Table Files/target.qvd'*** Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio: *c:\data\sales.qvd*** • relativo alla directory di lavoro dell'app Qlik Sense **Esempio: *data\sales.qvd*** Se il percorso viene omesso, Qlik Sense memorizza il file nella directory specificata dall'istruzione **Directory**. Se non è presente alcuna istruzione **Directory**, Qlik Sense memorizza il file nella directory di lavoro, *C:\Users\{user}\Documents\Qlik\Sense\Apps*.
*format-spec ::=***(** (**txt** | **qvd**)**)**	È possibile imostare la specifica del formato per entrambi i formati file. Se si omette la specifica del formato, viene utilizzato **qvd**. • **txt** per i file di testo. • **qvd** per i file qvd.

Esempi:

```
Store mytable into xyz.qvd (qvd);
Store * from mytable into 'lib://FolderConnection/myfile.qvd';
Store Name, RegNo from mytable into xyz.qvd;
Store Name as a, RegNo as b from mytable into 'lib://FolderConnection/myfile.qvd';
Store mytable into myfile.txt (txt);
Store * from mytable into 'lib://FolderConnection/myfile.qvd';
```

All'estensione file delle connessioni DataFiles si applica la distinzione tra maiuscole e minuscole. Ad esempio: .qvd.

Table/Tables

Le parole chiave dello script **Table** e **Tables** vengono utilizzate sia nelle istruzioni **Drop**, **Comment** e **Rename** che come identificatori di formato nelle istruzioni **Load**.

Tag

Questa istruzione dello script fornisce un modo per assegnare tag a uno o più campi o tabelle. Se viene effettuato un tentativo di contrassegnare un campo o una tabella non presente nell'app, tale contrassegno verrà ignorato. In caso di conflitto nelle ricorrenze di un nome di campo o di tag, si utilizza l'ultimo valore trovato.

Sintassi:

```
Tag [field|fields] fieldlist with tagname
```

```
Tag [field|fields] fieldlist using mapname
```

```
Tag table tablelist with tagname
```

Argomenti

Argomento	Descrizione
fieldlist	Uno o più campi che dovrebbero essere contrassegnati in un elenco separato da virgole.
mapname	Il nome di una tabella di mapping caricata in precedenza in un'istruzione **mapping Load** o **mapping Select**.
tablelist	Un elenco separato da virgole di tabelle che dovrebbero essere contrassegnate.
tagname	Il nome del tag da applicare al campo.

Example 1:

```
tagmap:
mapping LOAD * inline [
a,b
```

```
Alpha,MyTag
Num,MyTag
];
tag fields using tagmap;
```

Example 2:

```
tag field Alpha with 'MyTag2';
```

Trace

L'istruzione **trace** esegue la scrittura di una stringa nella finestra **Avanzamento dell'esecuzione dello script** e nel file di log dello script, quando viene utilizzato. Si rivela molto utile per le operazioni di debug. L'uso delle espansioni $ delle variabili calcolate prima dell'istruzione **trace** consente di personalizzare il messaggio.

Sintassi:

```
Trace string
```

Example 1:

L'istruzione seguente può essere utilizzata subito dopo l'istruzione LOAD che carica la tabella 'Principale'.

```
Trace Main table loaded;
```

Ciò visualizzerà il testo 'Tabella principale caricata' nella finestra di dialogo esecuzione script e nel file di log.

Example 2:

Le istruzioni seguenti possono essere utilizzate subito dopo l'istruzione LOAD che carica la tabella 'Principale'.

```
Let MyMessage = NoOfRows('Main') & ' rows in Main table';

Trace $(MyMessage);
```

Ciò visualizzerà un testo che mostra il numero di righe nella finestra di dialogo di esecuzione script e nel file di log, ad esempio '265.391 righe nella tabella principale'.

Unmap

L'istruzione **Unmap** disattiva il mapping del valore di campo specificato da un'istruzione **Map ... Using** precedente per i campi caricati successivamente.

Sintassi:

```
Unmap *fieldlist
```

Argomenti:

Argomenti

Argomento	Descrizione
*fieldlist	Un elenco separato da virgole dei campi per i quali non deve più essere eseguito il mapping da questo punto nello script. L'utilizzo di * per l'elenco dei campi indica tutti i campi. Nei nomi di campo sono consentiti i caratteri speciali * e ?. Se si utilizzano i caratteri speciali può essere necessario delimitare i nomi di campo tra virgolette.

Esempi e risultati:

Esempio	Risultato
Unmap Country;	Disabilita il mapping del campo Country.
Unmap A, B, C;	Disabilita il mapping dei campi A, B e C.
Unmap * ;	Disabilita il mapping di tutti i campi.

Unqualify

L'istruzione **Unqualify** viene utilizzata per disattivare la qualificazione dei nomi di campo che era stata precedentemente attivata dall'istruzione **Qualify**.

Sintassi:

```
Unqualify *fieldlist
```

Argomenti:

Argomenti

Argomento	Descrizione
*fieldlist	Un elenco separato da virgola dei campi per i quali è necessario attivare la qualificazione. L'utilizzo di * per l'elenco dei campi indica tutti i campi. Nei nomi di campo sono consentiti i caratteri speciali * e ?. Se si utilizzano i caratteri speciali può essere necessario delimitare i nomi di campo tra virgolette. Consultare la documentazione relativa all'istruzione **Qualify** per ulteriori informazioni.

Example 1:

In un database poco familiare si rivela spesso utile iniziare associando le varie tabelle secondo un unico campo o un numero minore di campi, come illustrato nel seguente esempio:

```
qualify *;
unqualify TransID;
SQL SELECT * from tab1;
SQL SELECT * from tab2;
SQL SELECT * from tab3;
```

Inizialmente, la qualificazione è attivata per tutti i campi.

Dopodiché, la qualificazione è disattivata per **TransID**.
Solo il campo **TransID** viene utilizzato per le associazioni tra le tre tabelle *tab1*, *tab2* e *tab3*. Tutti gli altri campi verranno qualificati con lo stesso nome tabella.

Untag

Questa istruzione dello script fornisce un modo per rimuovere tag da campi o tabelle. Se viene effettuato un tentativo di rimozione di un contrassegno da un campo o una tabella non presente nell'app, tale rimozione verrà ignorata.

Sintassi:

```
Untag [field|fields] fieldlist with tagname
```

```
Untag [field|fields] fieldlist using mapname
```

```
Untag table tablelist with tagname
```

Argomenti:

Argomenti

Argomento	Descrizione
fieldlist	Uno o più campi i cui tag dovrebbero essere rimossi, in un elenco separato da virgole.
mapname	Il nome di una tabella di mapping caricata in precedenza in un'istruzione mapping **LOAD** o mapping **SELECT**.
tablelist	Un elenco separato da virgole delle tabelle da cui rimuovere i contrassegni.
tagname	Il nome del tag che deve essere rimosso dal campo.

Example 1:

```
tagmap:
mapping LOAD * inline [
a,b
Alpha,MyTag
Num,MyTag
];
Untag fields using tagmap;
```

Example 2:

```
Untag field Alpha with MyTag2;
```

2.6 Directory di lavoro

Se in un'istruzione dello script si fa riferimento a un file di cui viene omesso il percorso, Qlik Sense eseguirà la ricerca del file nell'ordine seguente:

1. La directory specificata da un'istruzione **Directory** (supportata solo nella modalità di creazione degli script legacy).
2. Se non è presente alcuna istruzione **Directory**, Qlik Sense eseguirà la ricerca nella directory di lavoro.

Directory di lavoro Qlik Sense Desktop

In Qlik Sense Desktop, la directory di lavoro è *C:\Users\{user}\Documents\Qlik\Sense\Apps*.

Directory di lavoro Qlik Sense

In un'installazione server di Qlik Sense, la directory di lavoro è specificata in Qlik Sense Repository Service e, per impostazione predefinita, è *C:\ProgramData\Qlik\Sense\Apps*. Per ulteriori informazioni, vedere la Guida di Qlik Management Console.

2 Utilizzo delle variabili nell'editor caricamento dati

Una variabile in Qlik Sense può essere definita come un raccoglitore che memorizza un valore statico o un calcolo, ad esempio un valore numerico o alfanumerico. Quando si utilizza la variabile nell'app, qualsiasi modifica apportata alla variabile viene applicata ovunque venga utilizzata. È possibile definire le variabili utilizzando la relativa panoramica o nello script tramite l'editor caricamento dati. Per impostare il valore di una variabile si utilizzano le istruzioni **Let** o **Set** nello script di caricamento dei dati.

È inoltre possibile lavorare con le variabili di Qlik Sense dalla relativa panoramica mentre si modifica un foglio.

2.7 Panoramica

Se il primo carattere del valore di una variabile è un segno di uguale ' = ', Qlik Sense tenta di valutare il valore come se si trattasse di una formula (espressione di Qlik Sense), quindi visualizza o restituisce il risultato anziché il testo effettivo della formula.

Quando vengono utilizzate, la variabile viene sostituita dal suo valore. Le variabili possono essere utilizzate nello script per l'espansione con simbolo del dollaro e in diverse istruzioni di controllo. Ciò si rivela molto utile se la stessa stringa viene ripetuta molte volte nello script, ad esempio un percorso.

Alcune variabili di sistema speciali vengono impostate da Qlik Sense all'inizio dell'esecuzione dello script, indipendentemente dai loro valori precedenti.

2.8 Definizione di una variabile

Le variabili forniscono la possibilità di memorizzare valori statici o il risultato di un calcolo. Quando si definisce una variabile, utilizzare la sintassi seguente:

```
set variablename = string
```

oppure

```
let variable = expression
```

L'istruzione **Set** viene utilizzata per la valutazione della stringa. Assegna il testo a destra del segno di uguale alla variabile. L'istruzione **Let** valuta un'espressione a destra del segno di uguale al momento di eseguire lo script e assegna il risultato dell'espressione alla variabile.

Le variabili sono soggette alla distinzione tra maiuscole e minuscole.

Si sconsiglia di denominare una variabile con lo stesso nome utilizzato per un campo o una funzione in Qlik Sense.

Esempi:

`set x = 3 + 4; //` la variabile otterrà la stringa '3 + 4' come valore.

`let x = 3 + 4; //` restituisce 7 come valore.

`set x = Today(); //` restituisce 'Today()' come valore.

`let x = Today(); //` restituisce la data di oggi come valore, per esempio, '27/9/2021'.

2.9 Cancellare una variabile

Rimuovendo una variabile dallo script e ricaricando i dati, la variabile resta nell'app. Se si desidera rimuovere completamente la variabile dall'app, è necessario eliminarla anche dalla finestra di dialogo delle variabili.

2.10 Caricamento di un valore della variabile come valore di campo

Se si desidera caricare un valore della variabile come valore di campo in un'istruzione **LOAD** e il risultato dell'espansione del simbolo del dollaro restituisce del testo invece che dei numeri o un'espressione, la variabile espansa deve essere racchiusa tra virgolette singole.

Esempio:

In questo esempio, la variabile di sistema contenente l'elenco degli errori di script viene caricata in una tabella. Si può notare che l'espansione di ScriptErrorCount nella clausola **If** non richiede virgolette, mentre l'espansione di ScriptErrorList richiede le virgolette.

```
IF $(ScriptErrorCount) >= 1 THEN

   LOAD '$(ScriptErrorList)' AS Error AutoGenerate 1; END IF
```

2.11 Calcolo della variabile

Esistono diversi modi per utilizzare le variabili con i valori calcolati in Qlik Sense e il risultato dipende dal metodo di definizione e dal metodo di richiamo in un'espressione.

In questo esempio, sono stati caricati dei dati in linea:

```
LOAD * INLINE [
    Dim, Sales
    A, 150
    A, 200
    B, 240
```

```
    B, 230
    C, 410
    C, 330
];
```

Si definiscano due variabili:

```
Let vSales  = 'Sum(Sales)' ;
Let vSales2  = '=Sum(Sales)' ;
```

Nella seconda variabile un segno di uguale verrà aggiunto prima dell'espressione. Ciò attiverà il calcolo della variabile prima che ne venga eseguita l'espansione e prima che l'espressione venga valutata.

Se si utilizza la variabile vSales senza modifiche, ad esempio in una misura, il risultato sarà la stringa Sum (Sales), vale a dire non verrà eseguito alcun calcolo.

Se si aggiunge un'espansione con simbolo del dollaro e si richiama $(vSales) nell'espressione, la variabile viene estesa e viene visualizzata la somma di Sales.

Infine, se si richiama $(vSales2), la variabile verrà calcolata prima di essere espansa. Ciò significa che il risultato visualizzato è la somma totale di Sales. In questo grafico, la differenza tra l'uso di =$(vSales) e =$(vSales2) come espressioni di misura restituisce i risultati:

Risultati

Dim	$(vSales)	$(vSales2)
A	350	1560
B	470	1560
C	740	1560

Come si può vedere, $(vSales) risulta nella somma parziale di un valore di dimensione, mentre $(vSales2) risulta nella somma totale.

Sono disponibili le seguenti variabili di script:

- *Variabili di errore (page 272)*
- *Variabili di interpretazione numerica (page 208)*
- *Variabili di sistema (page 199)*
- *Variabili di gestione del valore (page 206)*

2.12 Variabili di sistema

Le variabili di sistema, talvolta definite dal sistema stesso, forniscono informazioni sul sistema e sull'app Qlik Sense.

Prospetto delle variabili di sistema

Alcune funzioni vengono ulteriormente descritte dopo la panoramica. Per tali funzioni, è inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Floppy

Restituisce la lettera relativa alla prima unità floppy rilevata, in genere *a:*. Questa è una variabile definita dal sistema.

```
Floppy
```

Questa variabile non è supportata in modalità standard.

CD

Restituisce la lettera relativa alla prima unità CD-ROM rilevata. Se non viene rilevata alcuna unità CD-ROM, viene restituito *c:*. Questa è una variabile definita dal sistema.

```
CD
```

Questa variabile non è supportata in modalità standard.

Include

La variabile **Include/Must_Include** specifica un file contenente del testo che deve essere inserito nello script e valutato come codice di script. Non è utilizzato per aggiungere dati. È possibile memorizzare parti del codice di script in un file di testo separato e riutilizzarlo in diverse app. Questa è una variabile definita dall'utente.

```
$(Include=filename)
$(Must_Include=filename)
```

HidePrefix

Tutti i nomi di campo che iniziano con questa stringa di testo verranno nascosti nella stessa maniera dei campi di sistema. Questa è una variabile definita dall'utente.

```
HidePrefix
```

HideSuffix

Tutti i nomi di campo che finiscono con questa stringa di testo verranno nascosti nella stessa maniera dei campi di sistema. Questa è una variabile definita dall'utente.

```
HideSuffix
```

QvPath

Restituisce la stringa costituita dal percorso del file eseguibile di Qlik Sense. Questa è una variabile definita dal sistema.

```
QvPath
```

Questa variabile non è supportata in modalità standard.

QvRoot

Restituisce la directory principale del file eseguibile di Qlik Sense. Questa è una variabile definita dal sistema.

```
QvRoot
```

Questa variabile non è supportata in modalità standard.

QvWorkPath

Restituisce la stringa costituita dal percorso dell'app Qlik Sense attuale. Questa è una variabile definita dal sistema.

```
QvWorkPath
```

Questa variabile non è supportata in modalità standard.

QvWorkRoot

Restituisce la directory principale dell'app Qlik Sense attuale. Questa è una variabile definita dal sistema.

```
QvWorkRoot
```

Questa variabile non è supportata in modalità standard.

StripComments

Se questa variabile è impostata su 0, la rimozione dei commenti /*..*/ e // nello script viene bloccata. Se questa variabile non è definita, la rimozione dei commenti verrà sempre eseguita.

```
StripComments
```

Verbatim

In genere, gli spazi vuoti (ASCII 32) iniziali e finali vengono rimossi automaticamente in tutti i valori di campo prima di essere caricati nel database di Qlik Sense. L'impostazione di questa variabile su 1 sospende la rimozione dei caratteri vuoti. La tabulazione (ASCII 9) e lo spazio unificatore (ANSI 160) non vengono mai rimossi.

```
Verbatim
```

OpenUrlTimeout

Questa variabile definisce l'attesa in secondi che Qlik Sense deve rispettare quando recupera i dati da sorgenti URL (ad esempio HTML pagine). Se viene omessa, l'attesa è di circa 20 minuti.

```
OpenUrlTimeout
```

WinPath

Restituisce la stringa costituita dal percorso di Windows. Questa è una variabile definita dal sistema.

```
WinPath
```

Questa variabile non è supportata in modalità standard.

WinRoot

Restituisce la directory principale di Windows. Questa è una variabile definita dal sistema.

```
WinRoot
```

Questa variabile non è supportata in modalità standard.

CollationLocale

Specifica le impostazioni locali da utilizzare per l'ordinamento e le corrispondenze della ricerca. Il valore corrisponde al nome della lingua di un'impostazione locale, ad esempio 'en-US'.Questa è una variabile definita dal sistema.

```
CollationLocale
```

CreateSearchIndexOnReload

Questa variabile definisce se durante il ricaricamento dati devono essere creati i file dell'indice di ricerca.

```
CreateSearchIndexOnReload
```

CreateSearchIndexOnReload

Questa variabile definisce se durante il ricaricamento dati devono essere creati i file dell'indice di ricerca.

Sintassi:

```
CreateSearchIndexOnReload
```

È possibile definire se i file dell'indice di ricerca devono essere creati durante il ricaricamento dei dati oppure dopo la prima richiesta di ricerca fatta da un utente. Il vantaggio della creazione dei file dell'indice di ricerca durante il ricaricamento dei dati è rappresentato dall'eliminazione dei tempi di attesa per il primo utente che esegue una ricerca. D'altra parte, però, la creazione dell'indice di ricerca aumenterà i tempi di ricaricamento dei dati.

Se questa variabile è omessa, i file dell'indice di ricerca non verranno creati durante il ricaricamento dei dati.

Per le app della sessione, i file dell'indice di ricerca non verranno creati durante il ricaricamento dei dati, indipendentemente dall'impostazione di questa variabile.

Example 1: Creazione dei campi dell'indice di ricerca durante il ricaricamento dei dati

```
set CreateSearchIndexOnReload=1;
```

Example 2: Creazione dei campi dell'indice di ricerca dopo la prima richiesta di ricerca

```
set CreateSearchIndexOnReload=0;
```

HidePrefix

Tutti i nomi di campo che iniziano con questa stringa di testo verranno nascosti nella stessa maniera dei campi di sistema. Questa è una variabile definita dall'utente.

Sintassi:

```
HidePrefix
```

Esempio:

```
set HidePrefix='_' ;
```

Se si utilizza questa istruzione, i nomi di campo che iniziano con un carattere di sottolineatura non vengono visualizzati negli elenchi dei nomi di campo quando i campi del sistema sono nascosti.

HideSuffix

Tutti i nomi di campo che finiscono con questa stringa di testo verranno nascosti nella stessa maniera dei campi di sistema. Questa è una variabile definita dall'utente.

Sintassi:

```
HideSuffix
```

Esempio:

```
set HideSuffix='%';
```

Se si utilizza questa istruzione, i nomi di campo che terminano con un segno percentuale non vengono visualizzati nell'elenco dei nomi di campo quando i campi del sistema sono nascosti.

Include

La variabile **Include/Must_Include** specifica un file contenente del testo che deve essere inserito nello script e valutato come codice di script. Non è utilizzato per aggiungere dati. È possibile memorizzare parti del codice di script in un file di testo separato e riutilizzarlo in diverse app. Questa è una variabile definita dall'utente.

Questa variabile supporta esclusivamente le connessioni dati cartella in modalità standard.

Sintassi:

```
$(Include=filename)
```

```
$(Must_Include=filename)
```

Esistono due versioni della variabile:

- **Include** non genera un errore quando non viene individuato il file e non visualizza alcun messaggio.
- **Must_Include** genera un errore quando non viene individuato il file.

Se non si specifica un percorso, il nome del file sarà relativo rispetto alla directory di lavoro dell'app di Qlik Sense. È inoltre possibile specificare un percorso del file assoluto o un percorso della connessione della cartella lib://. Non inserire uno spazio prima o dopo il segno di uguale.

*La costruzione **set Include =**filename non è applicabile.*

Esempi:

```
$(Include=abc.txt);
```

```
$(Must_Include=lib://DataFiles/abc.txt);
```

Limiti

Compatibilità incrociata limitata tra i file codificati UTF-8 sotto Windows rispetto a Linux.

In via opzionale è possibile usare UTF-8 con il BOM (Byte Order Mark). Il BOM può interferire con l'uso di UTF-8 nei software che non si aspettano byte non-ASCII all'inizio di un file, ma che potrebbe altrimenti gestire il flusso di testo.

- I sistemi Windows usano il BOM in UTF-8 per identificare che un file è codificato in UTF-8, nonostante non ci sia ambiguità nella memorizzazione dei byte.
- Unix/Linux usa UTF-8 per Unicode, ma non usano i BOM perché ciò interferisce con la sintassi dei file di comando.

Sono presenti alcune implicazioni per Qlik Sense.

- In Windows qualsiasi file che inizia con un BOM UTF-8 è considerato un file di script UTF-8. Altrimenti si presume l'uso della codifica ANSI.
- In Linux, la pagina di codice a 8 bit predefinita del sistema è UTF-8. È per questo che UTF-8 funziona anche se non contiene il BOM.

Di conseguenza, la portabilità non può essere garantita. Non è sempre possibile creare un file su Windows che possa essere interpretato da Linux e viceversa. Non esiste alcuna compatibilità incrociata tra i due sistemi per quanto riguarda i file codificati UTF-8, a causa della diversa gestione del BOM.

OpenUrlTimeout

Questa variabile definisce l'attesa in secondi che Qlik Sense deve rispettare quando recupera i dati da sorgenti URL (ad esempio HTML pagine). Se viene omessa, l'attesa è di circa 20 minuti.

Sintassi:

```
OpenUrlTimeout
```

Esempio:

```
set OpenUrlTimeout=10;
```

StripComments

Se questa variabile è impostata su 0, la rimozione dei commenti /*..*/ e // nello script viene bloccata. Se questa variabile non è definita, la rimozione dei commenti verrà sempre eseguita.

Sintassi:

```
StripComments
```

Alcuni driver del database utilizzano /*..*/ come suggerimento di ottimizzazione nelle istruzioni **SELECT**. In questo caso, non rimuovere i commenti prima di inviare l'istruzione **SELECT** al driver del database.

È consigliabile reimpostare questa variabile su 1 subito dopo l'istruzione o le istruzioni per le quali è stata necessaria.

Esempio:

```
set StripComments=0;
SQL SELECT * /* <optimization directive> */ FROM Table ;
set StripComments=1;
```

Verbatim

In genere, gli spazi vuoti (ASCII 32) iniziali e finali vengono rimossi automaticamente in tutti i valori di campo prima di essere caricati nel database di Qlik Sense. L'impostazione di questa variabile su 1 sospende la rimozione dei caratteri vuoti. La tabulazione (ASCII 9) e lo spazio unificatore (ANSI 160) non vengono mai rimossi.

Sintassi:

```
Verbatim
```

Esempio:

```
set Verbatim = 1;
```

2.13 Variabili di gestione del valore

In questa sezione vengono descritte le variabili utilizzate per gestire il valore NULL e gli altri valori.

Prospetto delle variabili di gestione del valore

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

NullDisplay
Il simbolo definito sostituisce tutti i valori NULL da ODBC e i connettori al livello di dati più basso. Questa è una variabile definita dall'utente.

```
NullDisplay
```

NullInterpret
Quando il simbolo definito compare in un file di testo, un file Excel o in un'istruzione inline, viene interpretato come NULL. Questa è una variabile definita dall'utente.

```
NullInterpret
```

NullValue
Se viene utilizzata l'istruzione **NullAsValue**, il simbolo definito sostituirà tutti i valori NULL nei campi **NullAsValue** specificati con la stringa specificata.

```
NullValue
```

OtherSymbol
Fa in modo che un simbolo venga trattato come 'tutti gli altri valori' prima di un'istruzione **LOAD/SELECT**. Questa è una variabile definita dall'utente.

```
OtherSymbol
```

NullDisplay

Il simbolo definito sostituisce tutti i valori NULL da ODBC e i connettori al livello di dati più basso. Questa è una variabile definita dall'utente.

Sintassi:

```
NullDisplay
```

Esempio:

```
set NullDisplay='<NULL>';
```

NullInterpret

Quando il simbolo definito compare in un file di testo, un file Excel o in un'istruzione inline, viene interpretato come NULL. Questa è una variabile definita dall'utente.

Sintassi:

```
NullInterpret
```

Esempi:

```
set NullInterpret=' ';
set NullInterpret =;
```

non restituisce valori NULL per i valori vuoti in Excel, tuttavia li restituisce nel caso di un file di testo CSV.

```
set NullInterpret ='';
```

restituisce valori NULL per i valori vuoti in Excel.

NullValue

Se viene utilizzata l'istruzione **NullAsValue**, il simbolo definito sostituirà tutti i valori NULL nei campi **NullAsValue** specificati con la stringa specificata.

Sintassi:

```
NullValue
```

Esempio:

```
NullAsValue Field1, Field2;
set NullValue='<NULL>';
```

OtherSymbol

Fa in modo che un simbolo venga trattato come 'tutti gli altri valori' prima di un'istruzione **LOAD/SELECT**. Questa è una variabile definita dall'utente.

Sintassi:

```
OtherSymbol
```

Esempio:

```
set OtherSymbol='+';
LOAD * inline
[X, Y
a, a
b, b];
LOAD * inline
[X, Z
a, a
+, c];
```

Il valore di campo Y='b' sarà ora collegato al valore Z='c' mediante l'altro simbolo.

2.14 Variabili di interpretazione numerica

Le variabili di interpretazione del numero sono definite dal sistema. Le variabili sono incluse all'inizio dello script di caricamento e applicano le impostazioni di formattazione dei numeri al momento dell'esecuzione dello script. Possono essere eliminate, modificate o duplicate.

Le variabili di interpretazione numerica sono generate automaticamente in base alle impostazioni regionali attuali del sistema operativo al momento della creazione di una nuova app. In Qlik Sense Desktop, questo dipende dalle impostazioni del sistema operativo del computer. In Qlik Sense riflettono le impostazioni del sistema operativo del server dove è installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Formattazione valuta

MoneyDecimalSep

Il separatore dei decimali specificato sostituisce il simbolo decimale per la valuta impostato nelle impostazioni locali.

```
MoneyDecimalSep
```

MoneyFormat

Il simbolo specificato sostituisce il simbolo della valuta impostato nelle impostazioni locali.

```
MoneyFormat
```

MoneyThousandSep

Il separatore delle migliaia specificato sostituisce il simbolo di raggruppamento delle cifre per la valuta configurata nelle impostazioni locali.

```
MoneyThousandSep
```

Formattazione numero

DecimalSep

Il separatore dei decimali specificato sostituisce il simbolo decimale impostato nelle impostazioni locali.

```
DecimalSep
```

ThousandSep

Il separatore delle migliaia specificato sostituisce il simbolo di raggruppamento delle cifre del sistema operativo (impostazioni locali).

```
ThousandSep
```

NumericalAbbreviation

L'abbreviazione numerica definisce l'abbreviazione da utilizzare per i prefissi in scala dei numeri, ad esempio M per mega o un milione (10^6) e µ per micro (10^{-6}).

```
NumericalAbbreviation
```

Formattazione dell'ora

DateFormat

Questa variabile d'ambiente definisce il formato di data utilizzato come predefinito nell'app. Il formato viene utilizzato sia per interpretare che per formattare le date. Se la variabile non è definita, il formato della data delle impostazioni regionali del sistema operativo sarà recuperato quando lo script viene eseguito.

```
DateFormat
```

TimeFormat

Il formato specificato sostituisce il formato dell'ora del sistema operativo (impostazioni locali).

```
TimeFormat
```

TimestampFormat

Il formato specificato sostituisce il formato della data e dell'ora del sistema operativo (impostazioni locali).

```
TimestampFormat
```

MonthNames

Il formato specificato sostituisce la convenzione usata per i nomi dei mesi nelle impostazioni locali.

```
MonthNames
```

LongMonthNames

Il formato specificato sostituisce la convenzione usata per i nomi lunghi dei mesi nelle impostazioni locali.

```
LongMonthNames
```

DayNames

Il formato specificato sostituisce la convenzione usata per i nomi dei giorni della settimana configurata nelle impostazioni locali.

```
DayNames
```

LongDayNames

Il formato specificato sostituisce la convenzione usata per i nomi lunghi dei giorni della settimana nelle impostazioni locali.

```
LongDayNames
```

FirstWeekDay

Numero intero che definisce il giorno da utilizzare come primo giorno della settimana.

```
FirstWeekDay
```

BrokenWeeks

Questa impostazione definisce se le settimane sono interrotte o meno.

```
BrokenWeeks
```

ReferenceDay

L'impostazione definisce il giorno del mese di gennaio da impostare come giorno di riferimento per definire la settimana 1.

```
ReferenceDay
```

FirstMonthOfYear

L'impostazione definisce il mese da utilizzare come primo mese dell'anno, un'opzione utile per definire gli anni finanziari che fanno uso di un offset per i mesi, iniziando ad esempio il 1 aprile.

Attualmente questa impostazione non viene utilizzata, tuttavia è stata riservata per un uso futuro.

Le impostazioni valide vanno dal 1 (gennaio) al 12 (dicembre). L'impostazione predefinita è 1.

Sintassi:

```
FirstMonthOfYear
```

Esempio:

```
Set FirstMonthOfYear=4; //Sets the year to start in April
```

BrokenWeeks

Questa impostazione definisce se le settimane sono interrotte o meno.

Sintassi:

```
BrokenWeeks
```

In Qlik Sense, le impostazioni regionali vengono recuperate alla creazione dell'app e le impostazioni corrispondenti vengono memorizzate nello script come variabili d'ambiente.

Uno sviluppatore di app nordamericano ottiene spesso `Set BrokenWeeks=1;` nello script, corrispondente a settimane interrotte. Uno sviluppatore di app europeo ottiene spesso `Set BrokenWeeks=0;` nello script, corrispondente a settimane intere.

Settimane ininterrotte significa che:

- In alcuni anni, la settimana 1 inizia a dicembre, mentre in altri anni l'ultima settimana dell'anno precedente continua a gennaio.
- Secondo la norma ISO 8601, la settimana 1 ha sempre almeno 4 giorni in gennaio. In Qlik Sense, questo può essere configurato utilizzando la variabile `ReferenceDay`.

Settimane interrotte significa che:

- L'ultima settimana dell'anno non prosegue mai a gennaio.
- La settimana 1 inizia il primo gennaio e, nella maggior parte dei casi, non è una settimana completa.

È possibile utilizzare i seguenti valori:

- 0 (= utilizzo di settimane intere)
- 1 (= utilizzo di settimane suddivise)

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi:

Se si desiderano le impostazioni ISO per le settimane e i numeri di settimana, assicurarsi di inserire nello script quanto segue:

```
Set FirstWeekDay=0;
Set BrokenWeeks=0;   //(use unbroken weeks)
Set ReferenceDay=4;
```

Se si desiderano le impostazioni USA, assicurarsi che nello script sia presente quanto segue:

```
Set FirstWeekDay=6;
Set BrokenWeeks=1;   //(use broken weeks)
Set ReferenceDay=1;
```

DateFormat

Questa variabile di ambiente definisce il formato della data utilizzato come predefinito nell'app e per data che restituisce funzioni come `date()` e `date#()`. Il formato viene utilizzato per interpretare e formattare le date. Se la variabile non è definita, il formato della data delle impostazioni regionali sarà recuperato quando viene eseguito lo script.

Sintassi:

```
DateFormat
```

Esempi di funzioni DateFormat

Esempio	Risultato
`Set DateFormat='M/D/YY'; //`	Quando viene utilizzata in questo modo, la funzione `DateFormat`

Esempio	Risultato
`(US format)`	definisce la data con il formato USA, mese/giorno/anno.
`Set DateFormat='DD/MM/YY'; //(UK date format)`	Quando viene utilizzata in questo modo, la funzione `DateFormat` definisce la data con il formato GB, giorno/mese/anno.
`Set DateFormat='YYYY/MM/DD'; //(ISO date format)`	Quando viene utilizzata in questo modo, la funzione `DateFormat` definisce la data con il formato ISO, anno/mese/giorno.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Variabili di sistema predefinite

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di date.
- La funzione `DateFormat`, che utilizzerà il formato data USA.

Nel seguente esempio, un set di dati è caricato in una tabella denominata '`Transactions`'. Include un campo `date`. Viene utilizzata la definizione USA `DateFormat`. Questo modello verrà utilizzato per il testo implicito per la conversione data quando vengono caricate date di testo.

Script di caricamento

```
Set DateFormat='MM/DD/YYYY';

Transactions:
LOAD
date,
month(date) as month,
id,
```

```
amount
INLINE
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `month`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	mese	=sum(amount)
01/01/2022	Jan	1000
02/01/2022	Feb	2123
03/01/2022	Mar	4124
04/01/2022	Apr	2431

La definizione `DateFormat` MM/GG/AAAA è usata per la conversione implicita del testo in date, per cui il campo `date` è correttamente interpretato come una data. Lo stesso formato è utilizzato per visualizzare la data, come mostrato nella tabella dei risultati.

Esempio 2 – Modifica della variabile di sistema

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dall'esempio precedente.
- La funzione `DateFormat`, che utilizzerà il formato 'GG/MM/AAAA'.

Script di caricamento

```
SET DateFormat='DD/MM/YYYY';
Transactions:
LOAD
```

```
date,
month(date) as month,
id,
amount
INLINE
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `month`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	mese	=sum(amount)
01/01/2022	Jan	1000
02/01/2022	Jan	2123
03/01/2022	Jan	4124
04/01/2022	Jan	2431

Poiché la definizione di `DateFormat` è stata impostata su 'GG/MM/AAAA', si può notare che le due cifre dopo il primo simbolo "/" sono state interpretate come il mese, con il risultato che tutti i record sono del mese di gennaio.

Esempio 3 – Interpretazione delle date

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati con date in formato numerico.
- La variabile `DateFormat`, che utilizzerà il formato 'GG-MM-AAAA'.
- La variabile `date()`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
date(numerical_date),
month(date(numerical_date)) as month,
id,
amount
Inline
[
numerical_date,id,amount
43254,1,1000
43255,2,2123
43256,3,4124
43258,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `month`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	mese	=sum(amount)
06/03/2022	Jun	1000
06/04/2022	Jun	2123
06/05/2022	Jun	4124
06/07/2022	Jun	2431

Nello script di caricamento, utilizzare la funzione `date()` per convertire la data numerica in un formato data. Dato che non si fornisce un formato specificato come argomento secondario nella funzione, viene utilizzato `DateFormat`. Ciò fa sì che il campo data utilizzi il formato 'MM/GG/AAAA'.

Esempio 4 – Formattazione di una data straniera

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di date.
- La variabile `DateFormat`, che utilizza il formato 'GG/MM/AAAA' ma non presenta commenti con barre terminali.

Script di caricamento

```
// SET DateFormat='DD/MM/YYYY';


Transactions:
Load
date,
month(date) as month,
id,
amount
Inline
[
date,id,amount
22-05-2022,1,1000
23-05-2022,2,2123
24-05-2022,3,4124
25-05-2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `month`

Creare questa misura:

```
=sum(amount)
```

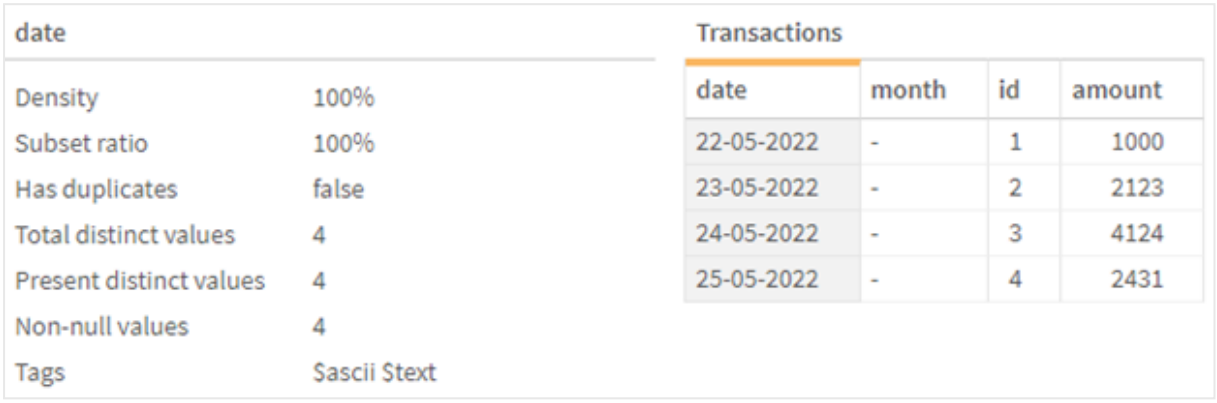
Tabella dei risultati

date	mese	=sum(amount)
22-05-2022	-	1000
23-05-2022	-	2123
24-05-2022	-	4124
25-05-2022	-	2431

Nello script di caricamento iniziale, il valore `DateFormat` in uso è il formato predefinito 'MM/GG/AAAA'. Poiché il campo `date` nel set di dati delle transazioni non è in questo formato, il campo non viene interpretato come una data. Ciò è mostrato nella tabella dei risultati in cui i valori del campo `month` sono NULL.

È possibile verificare i tipi di dati interpretati nel sistema di visualizzazione modello dati verificando le proprietà "Tag" del campo `date`:

Anteprima della tabella `Transactions`. Notare i "Tag" per il campo `date` che indicano che i dati di input testuali non sono stati convertiti implicitamente in una data/data e ora.



date

Density	100%
Subset ratio	100%
Has duplicates	false
Total distinct values	4
Present distinct values	4
Non-null values	4
Tags	$ascii $text

Transactions

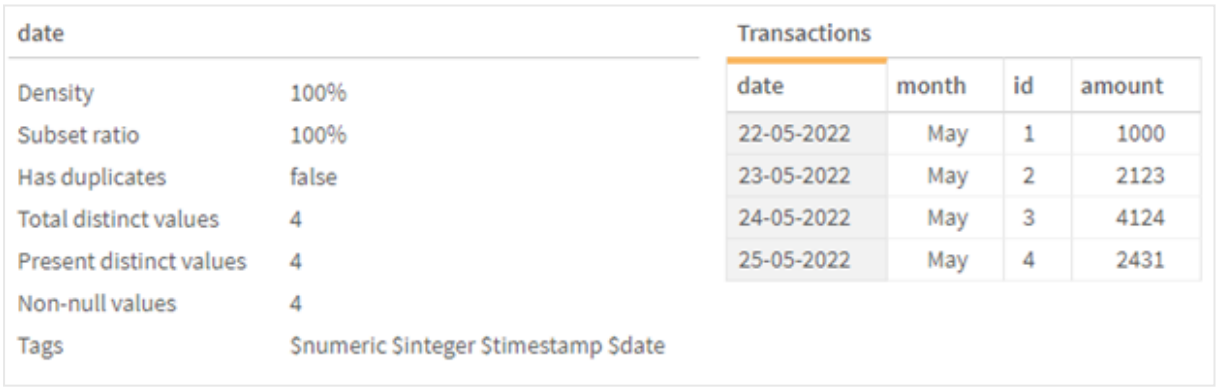
date	month	id	amount
22-05-2022	-	1	1000
23-05-2022	-	2	2123
24-05-2022	-	3	4124
25-05-2022	-	4	2431

Questo problema può essere risolto abilitando la variabile di sistema `DateFormat` :

```
// SET DateFormat='DD/MM/YYYY';
```

Rimuovere le doppie barre terminali e ricaricare i dati.

Anteprima della tabella `Transactions`. Notare i "Tag" per il campo `date` che indicano che i dati di input testuali sono stati convertiti implicitamente in una data/data e ora.



date

Density	100%
Subset ratio	100%
Has duplicates	false
Total distinct values	4
Present distinct values	4
Non-null values	4
Tags	$numeric $integer $timestamp $date

Transactions

date	month	id	amount
22-05-2022	May	1	1000
23-05-2022	May	2	2123
24-05-2022	May	3	4124
25-05-2022	May	4	2431

DayNames

Il formato specificato sostituisce la convenzione usata per i nomi dei giorni della settimana configurata nelle impostazioni locali.

Sintassi:

```
DayNames
```

Quando si modifica la variabile, è necessario inserire un punto e virgola ; per separare i singoli valori.

Esempi della funzione DayName

Esempio di funzione	Definizione del risultato
`Set DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';`	La funzione DayNames consente di definire i nomi dei giorni nella loro forma abbreviata.
`Set DayNames='M;Tu;W;Th;F;Sa;Su';`	La funzione DayNames consente di definire i nomi dei giorni utilizzando le lettere iniziali.

La funzione `DayNames` viene spesso utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
weekday (page 1064)	Funzione script per restituire `DayNames` come valori di campo.
Date (page 1223)	Funzione script per restituire `DayNames` come valori di campo.
LongDayNames (page 229)	Valori nel formato lungo di `DayNames`.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Variabili di sistema predefinite

Script di caricamento e risultati

Panoramica

In questo esempio, le date nel set di dati sono impostate nel formato MM/GG/AAAA.

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati con date, che verrà caricato in una tabella denominata `Transactions`.
- Un campo `date`.
- La definizione predefinita `DayNames`.

Script di caricamento

```
SET DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';

Transactions:
LOAD
date,
WeekDay(date) as dayname,
id,
amount
INLINE
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `dayname`

Creare questa misura:

```
sum(amount)
```

Tabella dei risultati

date	dayname	sum(amount)
01/01/2022	Sat	1000
02/01/2022	Tue	2123
03/01/2022	Tue	4124
04/01/2022	Fri	2431

Nello script di caricamento, la funzione `WeekDay` viene utilizzata con il campo `date` come argomento fornito. Nella tabella dei risultati, l'output di questa funzione `WeekDay` visualizza i giorni della settimana nel formato della definizione `DayNames`.

Esempio 2 - Modifica della variabile di sistema

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento di seguito in una nuova scheda. Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, all'inizio dello script, la definizione `DayNames` viene modificata per utilizzare i giorni abbreviati della settimana in afrikaans.

Script di caricamento

```
SET DayNames='Ma;Di;Wo;Do;Vr;Sa;So';

Transactions:
Load
date,
WeekDay(date) as dayname,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `dayname`

Creare questa misura:

```
sum(amount)
```

Tabella dei risultati

date	dayname	sum(amount)
01/01/2022	Sa	1000
02/01/2022	Di	2123
03/01/2022	Di	4124
04/01/2022	Vr	2431

Nella tabella dei risultati, l'output di questa funzione `WeekDay` visualizza i giorni della settimana nel formato della definizione `DayNames`.

È importante ricordare che se la lingua per `DayNames` viene modificata come in questo esempio, `LongDayNames` conterrà comunque i giorni della settimana in inglese. È necessario modificare anche questo valore se entrambe le variabili vengono utilizzate nell'applicazione.

Esempio 3 – Funzione data

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati con date, che verrà caricato in una tabella denominata `Transactions`.
- Un campo `date`.
- La definizione predefinita `DayNames`.

Script di caricamento

```
SET DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';

Transactions:
Load
date,
Date(date,'WWW') as dayname,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `dayname`

Creare questa misura:

```
sum(amount)
```

Tabella dei risultati

date	dayname	sum(amount)
01/01/2022	Sat	1000
02/01/2022	Tue	2123
03/01/2022	Tue	4124
04/01/2022	Fri	2431

Viene utilizzata la definizione predefinita `DayNames`. Nello script di caricamento, la funzione `Date` viene utilizzata con il campo `date` come primo argomento. Il secondo argomento è `WWW`. Questa formattazione converte il risultato nei valori archiviati nella definizione `DayNames`. Quest'ultimo viene visualizzato nell'output della tabella dei risultati.

DecimalSep

Il separatore dei decimali specificato sostituisce il simbolo decimale impostato nelle impostazioni locali.

Qlik Sense interpreta automaticamente il testo come numeri ogni volta che rileva uno schema numerico riconoscibile. Le variabili di sistema `ThousandSep` e `DecimalSep` determinano la composizione dei modelli applicati durante l'analisi del testo come numeri. Le variabili `ThousandSep` e `DecimalSep` impostano il modello di formattazione numerico predefinito durante la visualizzazione di contenuti numerici nei grafici e nelle tabelle front-end. Quindi, ha un impatto diretto sulle opzioni di **Formattazione numero** per qualsiasi espressione front-end.

L'adozione di un separatore delle migliaia di virgola ',' e un separatore decimale di '.', sono esempi di modelli che verrebbero convertiti implicitamente in valori numerici equivalenti:

```
0,000.00
```

```
0000.00
```

```
0,000
```

Questi sono esempi di modelli che rimarrebbero invariati come testo; cioè, non verrebbero convertiti in un valore numerico:

```
0.000,00
```

```
0,00
```

Sintassi:

```
DecimalSep
```

Esempi di funzioni

Esempio	Risultato
`Set DecimalSep='.';`	Imposta '.' come separatore decimale.
`Set DecimalSep=',';`	Imposta ',' come separatore decimale.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio – Effetto dell'impostazione delle variabili di separazione dei numeri su dati di input diversi

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di somme e date con le somme impostate in modelli di formato differenti.
- Una tabella denominata `Transactions`.
- La variabile di sistema `DecimalSep` che è impostata su '.'.
- La variabile di sistema `ThousandSep` che è impostata su ','.
- La variabile `delimiter` che è impostata come carattere '|' per separare i diversi campi in una riga.

Script di caricamento

```
Set ThousandSep=',';
Set DecimalSep='.';

Transactions:
Load date,
id,
amount as amount
Inline
[
date|id|amount
01/01/2022|1|1.000-45
01/02/2022|2|23.344
01/03/2022|3|4124,35
01/04/2022|4|2431.36
01/05/2022|5|4,787
01/06/2022|6|2431.84
```

```
01/07/2022|7|4132.5246
01/08/2022|8|3554.284
01/09/2022|9|3.756,178
01/10/2022|10|3,454.356
] (delimiter is '|');
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `amount`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

Importo	=Sum(amount)	
Totali		20814.7086
1.000-45		
3.756,178		
4124,35		
	23.344	23.344
	2431.36	2431.36
	2431.84	2431.84
	3,454.356	3454.356
	3554.284	3554.284
	4132.5246	4132.5246
	4,787	4787

Qualsiasi valore non interpretato come numero viene trattato come testo ed è allineato a sinistra per impostazione predefinita. Tutti i valori convertiti correttamente vengono allineati a destra, mantenendo il formato di input originale.

La colonna dell'espressione mostra l'equivalente numerico, che per impostazione predefinita è formattato solo con un separatore decimale '.'. È possibile modificare la formattazione tramite l'impostazione del menu a discesa **Formattazione numero** nella configurazione dell'espressione.

FirstWeekDay

Numero intero che definisce il giorno da utilizzare come primo giorno della settimana.

Sintassi:

```
FirstWeekDay
```

Il lunedì è il primo giorno della settimana secondo la norma ISO 8601, lo standard internazionale per la rappresentazione di date e orari. Il lunedì è anche utilizzato come primo giorno della settimana in diversi Paesi, ad esempio nel Regno Unito, in Francia, in Germania e in Svezia.

Ma in altri Paesi, come gli Stati Uniti e il Canada, la domenica è considerata l'inizio della settimana.

In Qlik Sense, le impostazioni regionali vengono recuperate alla creazione dell'app e le impostazioni corrispondenti vengono memorizzate nello script come variabili d'ambiente.

Uno sviluppatore di app nordamericano ottiene spesso `Set FirstWeekDay=6;` nello script, corrispondente alla domenica. Uno sviluppatore di app europeo ottiene spesso `Set FirstWeekDay=0;` nello script, corrispondente al lunedì.

Valori impostabili per FirstWeekDay

Valore	Giorno
0	Lunedì
1	Martedì
2	Mercoledì
3	Giovedì
4	Venerdì
5	Sabato
6	Domenica

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi:

Se si desiderano le impostazioni ISO per le settimane e i numeri di settimana, assicurarsi di inserire nello script quanto segue:

```
Set FirstWeekDay=0;  // Monday as first week day
Set BrokenWeeks=0;
Set ReferenceDay=4;
```

Se si desiderano le impostazioni USA, assicurarsi che nello script sia presente quanto segue:

```
Set FirstWeekDay=6;  // Sunday as first week day
Set BrokenWeeks=1;
Set ReferenceDay=1;
```

Esempio 1: utilizzo del valore predefinito (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

In questo esempio, lo script di caricamento utilizzare il valore della variabile di sistema Qlik Sense predefinita, `FirstWeekDay=6`. Questi dati contengono i dati per i primi 14 giorni nel 2020.

Script di caricamento

```
// Example 1: Load Script using the default value of FirstWeekDay=6, i.e. Sunday

SET FirstWeekDay = 6;

Sales:
LOAD
        date,
        sales,
        week(date) as week,
        weekday(date) as weekday
Inline [
date,sales
01/01/2021,6000
01/02/2021,3000
01/03/2021,6000
01/04/2021,8000
01/05/2021,5000
01/06/2020,7000
01/07/2020,3000
01/08/2020,5000
01/09/2020,9000
01/10/2020,5000
01/11/2020,7000
01/12/2020,7000
01/13/2020,7000
01/14/2020,7000
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week`
- `weekday`

Tabella dei risultati

Data	settimana	weekday
01/01/2021	1	Mer
01/02/2021	1	Gio
01/03/2021	1	Fri
01/04/2021	1	Sat
01/05/2021	2	Sun
01/06/2020	2	Mon
01/07/2020	2	Tue
01/08/2020	2	Wed
01/09/2020	2	Thu
01/10/2020	2	Fri
01/11/2020	2	Sat
01/12/2020	3	Dom
01/13/2020	3	Lun
01/14/2020	3	Mar

Dato che vengono utilizzate le impostazioni predefinite, la variabile di sistema `FirstWeekDay` è impostata su 6. Nella tabella dei risultati, ciascuna nuova settimana può essere visualizzata iniziando dalla domenica (il 5 e il 12 di gennaio).

Esempio 2 – Cambiamento della variabile FirstWeekDay (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

In questo esempio, i dati contengono i primi 14 giorni nel 2020. All'inizio dello script, abbiamo impostato la variabile `FirstWeekDay` a 3.

Script di caricamento

```
// Example 2: Load Script setting the value of FirstWeekDay=3, i.e. Thursday

SET FirstWeekDay = 3;
```

```
Sales:
LOAD
       date,
       sales,
       week(date) as week,
       weekday(date) as weekday
Inline [
date,sales
01/01/2021,6000
01/02/2021,3000
01/03/2021,6000
01/04/2021,8000
01/05/2021,5000
01/06/2020,7000
01/07/2020,3000
01/08/2020,5000
01/09/2020,9000
01/10/2020,5000
01/11/2020,7000
01/12/2020,7000
01/13/2020,7000
01/14/2020,7000
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week`
- `weekday`

Tabella dei risultati

Data	settimana	weekday
01/01/2021	52	Mer
01/02/2021	1	Gio
01/03/2021	1	Fri
01/04/2021	1	Sat
01/05/2021	1	Sun
01/06/2020	1	Mon
01/07/2020	1	Tue
01/08/2020	1	Wed
01/09/2020	2	Thu

Data	settimana	weekday
01/10/2020	2	Fri
01/11/2020	2	Sat
01/12/2020	2	Dom
01/13/2020	2	Lun
01/14/2020	2	Mar

Dato che la variabile di sistema `FirstWeekDay` è impostata su 3, il primo giorno di ogni settimana sarà un giovedì. Nella tabella dei risultati, ciascuna nuova settimana può essere vista iniziare il giovedì (il 2 e il 9 di gennaio).

LongDayNames

Il formato specificato sostituisce la convenzione usata per i nomi lunghi dei giorni della settimana nelle impostazioni locali.

Sintassi:

```
LongDayNames
```

Il seguente esempio della funzione `LongDayNames` definisce i nomi dei giorni per intero:

```
Set LongDayNames='Monday;Tuesday;Wednesday;Thursday;Friday;Saturday;Sunday';
```

Quando si modifica la variabile, è necessario inserire un punto e virgola `;` per separare i singoli valori.

La funzione `LongDayNames` può essere utilizzata in combinazione con la funzione *Date (page 1223)* che restituisce `DayNames` come valori di campo.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Variabili di sistema predefinite

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati con date, che verrà caricato in una tabella denominata `Transactions`.
- Un campo `date`.
- La definizione predefinita `LongDayNames`.

Script di caricamento

```
SET LongDayNames='Monday;Tuesday;Wednesday;Thursday;Friday;Saturday;Sunday';

Transactions:
LOAD
date,
Date(date,'WWWW') as dayname,
id,
amount
INLINE
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `dayname`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	dayname	=sum(amount)
01/01/2022	Saturday	1000
02/01/2022	Tuesday	2123
03/01/2022	Tuesday	4124
04/01/2022	Friday	2431

Nello script di caricamento, per creare un campo denominato `dayname`, viene utilizzata la funzione `Date` viene utilizzata con il campo `date` come primo argomento. Il secondo argomento nella funzione è la formattazione `WWWW`.

L'utilizzo di questa formattazione converte i valori del primo argomento nel nome del giorno completo corrispondente impostato nella variabile `LongDayNames`. Questo viene mostrato nella tabella dei risultati, nei valori di campo per il campo creato `dayname`.

Esempio 2 – Modifica della variabile di sistema

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento di seguito in una nuova scheda.

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, all'inizio dello script, la definizione `LongDayNames` viene modificata per utilizzare i giorni della settimana in spagnolo.

Script di caricamento

```
SET LongDayNames='Lunes;Martes;Miércoles;Jueves;Viernes;Sábado;Domingo';

Transactions:
LOAD
date,
Date(date,'WWWW') as dayname,
id,
amount
INLINE
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `dayname`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	dayname	=sum(amount)
01/01/2022	Sábado	1000
02/01/2022	Martes	2123
03/01/2022	Martes	4124
04/01/2022	Viernes	2431

Nello script di caricamento, la variabile `LongDayNames` viene modificata per elencare i giorni della settimana in spagnolo.

Quindi, è necessario creare un campo denominato `dayname`, che è la funzione `Date` utilizzata con il campo `date` come primo argomento.

Il secondo argomento nella funzione è la formattazione `WWWW`. L'utilizzo della formattazione Qlik Sense converte i valori del primo argomento nel nome del giorno completo corrispondente impostato nella variabile `LongDayNames`.

Nella tabella dei risultati, i valori di campo per il campo creato `dayname` visualizza i giorni della settimana scritti per intero in spagnolo.

LongMonthNames

Il formato specificato sostituisce la convenzione usata per i nomi lunghi dei mesi nelle impostazioni locali.

Sintassi:

```
LongMonthNames
```

Quando si modifica la variabile, è necessario utilizzare ; per separare i singoli valori.

Il seguente esempio della funzione LongMonthNames definisce i nomi completi dei mesi:

```
Set
LongMonthNames='January;February;March;April;May;June;July;August;September;October;November;D
ecember';
```

La funzione `LongMonthNames` viene spesso utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
Date (page 1223)	Funzione script per restituire `DayNames` come valori di campo.
LongDayNames (page 229)	Valori nel formato lungo di `DayNames`.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema,

a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Variabili di sistema predefinite

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Un campo `date`.
- La definizione predefinita `LongMonthNames`.

Script di caricamento

```
SET
LongMonthNames='January;February;March;April;May;June;July;August;September;October;November;D
ecember';

Transactions:
Load
date,
Date(date,'MMMM') as monthname,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,1000.45
01/02/2022,2,2123.34
01/03/2022,3,4124.35
01/04/2022,4,2431.36
01/05/2022,5,4787.78
01/06/2022,6,2431.84
01/07/2022,7,2854.83
01/08/2022,8,3554.28
01/09/2022,9,3756.17
01/10/2022,10,3454.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `monthname`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	monthname	sum(amount)
01/01/2022	Gennaio	1000.45
01/02/2022	Gennaio	2123.34
01/03/2022	Gennaio	4124.35
01/04/2022	Gennaio	2431.36
01/05/2022	Gennaio	4787.78
01/06/2022	Gennaio	2431.84
01/07/2022	Gennaio	2854.83
01/08/2022	Gennaio	3554.28
01/09/2022	Gennaio	3756.17
01/10/2022	Gennaio	3454.35

Viene utilizzata la definizione predefinita `LongMonthNames`. Nello script di caricamento, per creare un campo denominato `month`, viene utilizzata la funzione `Date` viene utilizzata con il campo `date` come primo argomento. Il secondo argomento nella funzione è la formattazione `MMMM`.

L'utilizzo della formattazione Qlik Sense converte i valori del primo argomento nel nome completo del mese corrispondente impostato nella variabile `LongMonthNames`. Questo viene mostrato nella tabella dei risultati, nei valori di campo per il campo creato `month`.

Esempio 2 - Modifica della variabile di sistema

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Un campo `date`.

- La variabile `LongMonthNames` modificata per utilizzare i giorni abbreviati della settimana in spagnolo.

Script di caricamento

```
SET
LongMonthNames='Enero;Febrero;Marzo;Abril;Mayo;Junio;Julio;Agosto;Septiembre;OctubreNoviembre;
Diciembre';

Transactions:
LOAD
date,
Date(date,'MMMM') as monthname,
id,
amount
INLINE
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere `sum(amount)` come misura e i seguenti campi come dimensioni:

- `date`
- `monthname`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	monthname	sum(amount)
01/01/2022	Enero	1000.45
01/02/2022	Enero	2123.34
01/03/2022	Enero	4124.35
01/04/2022	Enero	2431.36
01/05/2022	Enero	4787.78
01/06/2022	Enero	2431.84
01/07/2022	Enero	2854.83
01/08/2022	Enero	3554.28
01/09/2022	Enero	3756.17
01/10/2022	Enero	3454.35

Nello script di caricamento, la variabile LongMonthNames viene modificata per elencare i mesi dell'anno in spagnolo. Quindi, per creare un campo denominato monthname, la funzione Date viene utilizzata con il campo date come primo argomento. Il secondo argomento nella funzione è la formattazione MMMM.

L'utilizzo della formattazione Qlik Sense converte i valori del primo argomento nel nome completo del mese corrispondente impostato nella variabile LongMonthNames. Nella tabella dei risultati, i valori di campo per il campo creato monthname visualizzano il nome del mese in spagnolo.

MoneyDecimalSep

Il separatore dei decimali specificato sostituisce il simbolo decimale per la valuta impostato nelle impostazioni locali.

Per impostazione predefinita, in Qlik Sense i numeri e il testo vengono visualizzati in modo diverso nei grafici a tabella. I numeri sono allineati a destra e il testo è allineato a sinistra. In questo modo è facile individuare i problemi di conversione da testo a numero. Tutte le tabelle di questa pagina che mostrano i risultati Qlik Sense utilizzeranno questa formattazione.

Sintassi:

MoneyDecimalSep

Le applicazioni Qlik Sense interpretano i campi di testo conformi a questa formattazione come valori monetari. Il campo di testo deve contenere il simbolo di valuta definito nella variabile di sistema MoneyFormat. MoneyDecimalSep è particolarmente utile quando si gestiscono sorgenti dati ricevute da più impostazioni regionali diverse.

Il seguente esempio mostra un possibile utilizzo della variabile di sistema MoneyDecimalSep:

```
Set MoneyDecimalSep='.';
```

Questa funzione viene spesso utilizzata insieme alle funzioni seguenti:

Funzioni correlate

Funzione	Interazione
MoneyFormat	Nei casi di interpretazione di campi di testo, il simbolo MoneyFormat verrà utilizzato come parte dell'interpretazione. Per la Formattazione dei numeri, la formattazione MoneyFormat verrà utilizzata da Qlik Sense negli Oggetti grafico.
MoneyThousandSep	Nei casi di interpretazione di campi di testo, è necessario rispettare anche la funzione MoneyThousandSep.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione SET DateFormat nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema,

a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Notazione MoneyDecimalSep punto (.)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Dati forniti che hanno il loro campo monetario in formato testo con un punto '.' usato come separatore decimale. Ogni record è inoltre preceduto dal simbolo "$", tranne l'ultimo record, che è preceduto dal simbolo "£".

Tenere presente che la variabile di sistema `MoneyFormat` definisce il dollaro '$' come valuta predefinita.

Script di caricamento

```
SET MoneyThousandSep=',';
SET MoneyDecimalSep='.';
SET MoneyFormat='$###0.00;-$###0.00';

Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,'$14.41'
01/02/2022,2,'$2,814.32'
01/03/2022,3,'$249.36'
01/04/2022,4,'$24.37'
01/05/2022,5,'$7.54'
01/06/2022,6,'$243.63'
01/07/2022,7,'$545.36'
01/08/2022,8,'$3.55'
01/09/2022,9,'$3.436'
01/10/2022,10,'£345.66'
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`amount`.

Aggiungere le seguenti misure:

- `isNum(amount)`
- `sum(amount)`

Esaminare i risultati qui sotto, dimostrando la corretta interpretazione di tutti i valori del dollaro "$".

Tabella dei risultati

importo	=isNum(amount)	=Sum(amount)
Totali	0	$3905.98
£345.66	0	$0.00
$3.436	-1	$3.44
$3.55	-1	$3.55
$7.54	-1	$7.54
$14.41	-1	$14.41
$24.37	-1	$24.37
243.63	-1	$243.63
$249.36	-1	$249.36
$545.36	-1	$545.36
$2,814.32	-1	$2814.32

La tabella dei risultati sopra mostra come il campo `amount` sia stato interpretato correttamente per tutti i valori con prefisso dollaro ($), mentre il prefisso sterlina (£) `amount` non è stato convertito in un valore monetario.

Esempio 2 - Notazione MoneyDecimalSep virgola (,)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Dati forniti che hanno il loro campo monetario in formato testo con una virgola '.' usato come separatore decimale. Ogni record è inoltre preceduto dal simbolo '$', ad eccezione dell'ultimo record, che utilizza erroneamente il separatore decimale a punti '.'.

Tenere presente che la variabile di sistema `MoneyFormat` definisce il dollaro '$' come valuta predefinita.

Script di caricamento

```
SET MoneyThousandSep='.';
SET MoneyDecimalSep=',';
SET MoneyFormat='$###0.00;-$###0.00';

Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,'$14,41'
01/02/2022,2,'$2.814,32'
01/03/2022,3,'$249,36'
01/04/2022,4,'$24,37'
01/05/2022,5,'$7,54'
01/06/2022,6,'$243,63'
01/07/2022,7,'$545,36'
01/08/2022,8,'$3,55'
01/09/2022,9,'$3,436'
01/10/2022,10,'$345.66'
];
```

Risultati

Testo del paragrafo per i Risultati.

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`amount`.

Aggiungere le seguenti misure:

- `isNum(amount)`
- `sum(amount)`

Esaminare i risultati qui sotto, dimostrando la corretta interpretazione di tutti i valori, ad eccezione dell'importo in cui il separatore decimale utilizza la notazione del punto '. In questo caso si sarebbe dovuta usare una virgola.

Tabella dei risultati

importo	=isNum(amount)	=Sum(amount)
Totali	0	$3905.98
$345.66	0	$0.00
$3,436	-1	$3.44
$3,55	-1	$3.55
$7,54	-1	$7.54
$14,41	-1	$14.41
$24,37	-1	$24.37
$243,63	-1	$243.63
$249,36	-1	$249.36
$545,36	-1	$545.36
$2.814,32	-1	$2814.32

MoneyFormat

Questa variabile di sistema definisce il modello di formattazione utilizzato da Qlikper la traduzione automatica di testo in numero quando il numero è preceduto da un simbolo monetario. Definisce anche il modo in cui le misure le cui proprietà di formattazione dei numeri sono impostate su "denaro" saranno visualizzate negli oggetti grafici.

Il simbolo definito come parte del modello di formattazione nella variabile di sistema `MoneyFormat` sostituisce il simbolo di valuta impostato dalle impostazioni regionali.

Per impostazione predefinita, in Qlik Sense i numeri e il testo vengono visualizzati in modo diverso nei grafici a tabella. I numeri sono allineati a destra e il testo è allineato a sinistra. In questo modo è facile individuare i problemi di conversione da testo a numero. Tutte le tabelle di questa pagina che mostrano i risultati Qlik Sense utilizzeranno questa formattazione.

Sintassi:

MoneyFormat

```
Set MoneyFormat='$ #,##0.00; ($ #,##0.00)';
```

Questa formattazione viene visualizzata negli oggetti grafico quando la proprietà di un campo numerico `Number Formatting` è impostata su `Money`. Inoltre, quando i campi di testo numerici sono interpretati da Qlik Sense, se il simbolo di valuta del campo di testo corrisponde a quello del simbolo definito nella variabile `MoneyFormat`, Qlik Sense interpreterà il campo come un valore monetario.

Questa funzione viene spesso utilizzata insieme alle funzioni seguenti:

Funzioni correlate

Funzione	Interazione
MoneyDecimalSep (page 236)	Per la formattazione dei numeri, verrà utilizzato `MoneyDecimalSep` nella formattazione dei campi degli oggetti.
MoneyThousandSep (page 244)	Per la formattazione dei numeri, verrà utilizzato `MoneyThousandSep` nella formattazione dei campi degli oggetti.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - MoneyFormat

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene un set di dati che viene caricato in una tabella denominata `Transactions`. Viene utilizzata la definizione di variabile `MoneyFormat` predefinita.

Script di caricamento

```
SET MoneyThousandSep=',';
SET MoneyDecimalSep='.';
SET MoneyFormat='$###0.00;-$###0.00';

Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,$10000000441
01/02/2022,2,$21237492432
```

```
01/03/2022,3,$249475336
01/04/2022,4,$24313369837
01/05/2022,5,$7873578754
01/06/2022,6,$24313884663
01/07/2022,7,$545883436
01/08/2022,8,$35545828255
01/09/2022,9,$37565817436
01/10/2022,10,$3454343566
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `amount`

Aggiungere questa misura:

```
=Sum(amount)
```

Sotto **Formattazione numeri**, selezionare **Denaro** per configurare `Sum(amount)` come valore monetario.

Tabella dei risultati

date	Importo	=Sum(amount)
Totali		$165099674156.00
01/01/2022	$10000000441	$10000000441.00
01/02/2022	$21237492432	$21237492432.00
01/03/2022	$249475336	$249475336.00
01/04/2022	$24313369837	$24313369837.00
01/05/2022	$7873578754	$7873578754.00
01/06/2022	$24313884663	$24313884663.00
01/07/2022	$545883436	$545883436.00
01/08/2022	$35545828255	$35545828255.00
01/09/2022	$37565817436	$37565817436.00
01/10/2022	$3454343566	$3454343566.00

Viene utilizzata la definizione predefinita `MoneyFormat`. L'aspetto sarà il seguente: `$###0.00;-$###0.00`. Nella tabella dei risultati, il formato del campo `amount` mostra il simbolo della valuta e la virgola e i decimali sono stati inclusi.

Esempio 2 - MoneyFormat con separatore delle migliaia e formati di input misti

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati in formato misto, che viene caricato in una tabella denominata `Transactions` con separatori di migliaia e separatori decimali intercalati.
- Una modifica della definizione `MoneyFormat` è stata apportata per includere una virgola come separatore delle migliaia.
- Una delle righe di dati è stata erroneamente delimitata con migliaia di virgole di separazione nei punti sbagliati. Si noti che questo importo viene lasciato come testo e non è interpretabile come numero.

Script di caricamento

```
SET MoneyThousandSep=',';
SET MoneyDecimalSep='.';
SET MoneyFormat = '$#,##0.00;-$#,##0.00';

Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,'$10,000,000,441.45'
01/02/2022,2,'$212,3749,24,32.23'
01/03/2022,3,$249475336.45
01/04/2022,4,$24,313,369,837
01/05/2022,5,$7873578754
01/06/2022,6,$24313884663
01/07/2022,7,$545883436
01/08/2022,8,$35545828255
01/09/2022,9,$37565817436
01/10/2022,10,$3454343566
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `amount`

Aggiungere questa misura:

```
=Sum(amount)
```

Sotto **Formattazione numeri**, selezionare **Denaro** per configurare `Sum(amount)` come valore monetario.

Tabella dei risultati

date	Importo	=Sum(amount)
Totali		$119,548,811,911.90
01/01/2022	$10,000,000,441.45	$10,000,000,441.45
01/02/2022	$212,3749,24,32.23	$0.00
01/03/2022	$249475336.45	$249,475,336.45
01/04/2022	$24	$24.00
01/05/2022	$7873578754	$7,873,578,754.00
01/06/2022	$24313884663	$24,313,884,663.00
01/07/2022	$545883436	$545,883,436.00
01/08/2022	$35545828255	$35,545,828,255.00
01/09/2022	$37565817436	$37,565,817,436.00
01/10/2022	$3454343566	$3,454,343,566.00

All'inizio dello script, la variabile di sistema `MoneyFormat` viene modificata per includere una virgola come separatore delle migliaia. Nella tabella Qlik Sense, si può notare che la formattazione include questo separatore. Inoltre, la riga con il separatore errato non è stata interpretata correttamente e rimane come testo. Per questo motivo non contribuisce alla somma dell'importo.

MoneyThousandSep

Il separatore delle migliaia specificato sostituisce il simbolo di raggruppamento delle cifre per la valuta configurata nelle impostazioni locali.

Per impostazione predefinita, in Qlik Sense i numeri e il testo vengono visualizzati in modo diverso nei grafici a tabella. I numeri sono allineati a destra e il testo è allineato a sinistra. In questo modo è facile individuare i problemi di conversione da testo a numero. Tutte le tabelle di questa pagina che mostrano i risultati Qlik Sense utilizzeranno questa formattazione.

Sintassi:

```
MoneyThousandSep
```

Le applicazioni Qlik Sense interpretano i campi di testo conformi a questa formattazione come valori monetari. Il campo di testo deve contenere il simbolo di valuta definito nella variabile di sistema `MoneyFormat`. `MoneyThousandSep` è particolarmente utile quando si gestiscono sorgenti dati ricevute da più impostazioni regionali diverse.

Il seguente esempio mostra un possibile utilizzo della variabile di sistema `MoneyThousandSep`:

```
Set MoneyDecimalSep=',';
```

Questa funzione viene spesso utilizzata insieme alle funzioni seguenti:

Funzioni correlate

Funzione	Interazione
`MoneyFormat`	Nei casi di interpretazione di campi di testo, il simbolo `MoneyFormat` verrà utilizzato come parte dell'interpretazione. Per la Formattazione dei numeri, la formattazione `MoneyFormat` verrà utilizzata da Qlik Sense negli Oggetti grafico.
`MoneyDecimalSep`	Nei casi di interpretazione di campi di testo, è necessario rispettare anche la funzione `MoneyDecimalSep`.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Notazione MoneyThousandSep virgola (,)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Dati forniti che hanno il loro campo monetario in formato testo con una virgola usato come separatore delle migliaia. Ogni record è inoltre preceduto dal simbolo "$".

Tenere presente che la variabile di sistema `MoneyFormat` definisce il dollaro '$' come valuta predefinita.

Script di caricamento

```
SET MoneyThousandSep=',';
SET MoneyDecimalSep='.';
SET MoneyFormat='$###0.00;-$###0.00';
```

```
Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,'$10,000,000,441'
01/02/2022,2,'$21,237,492,432'
01/03/2022,3,'$249,475,336'
01/04/2022,4,'$24,313,369,837'
01/05/2022,5,'$7,873,578,754'
01/06/2022,6,'$24,313,884,663'
01/07/2022,7,'$545,883,436'
01/08/2022,8,'$35,545,828,255'
01/09/2022,9,'$37,565,817,436'
01/10/2022,10,'$3.454.343.566'
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`amount`.

Aggiungere le seguenti misure:

- `isNum(amount)`
- `sum(amount)`

Esaminare i risultati di seguito. La tabella mostra la corretta interpretazione di tutti i valori utilizzando la virgola ',' come separatore delle migliaia.

Il campo `amount` è stato interpretato correttamente per tutti i valori, ad eccezione di un valore che utilizzava un punto '.' come separatore delle migliaia.

Tabella dei risultati

importo	=isNum(amount)	=Sum(amount)
Totali	0	$161645330590.00
$3.454.343.566	0	$0.00
$249,475,336	-1	$249475336.00
$545,883,436	-1	$545883436.00
$7,873,578,754	-1	$7873578754.00
$10,000,000,441	-1	$10000000441.00
$21,237,492,432	-1	$21237492432.00

importo	=isNum(amount)	=Sum(amount)
$24,313,369,837	-1	$24313369837.00
$24,33,884,663	-1	$24313884663.00
$35,545,828,255	-1	$35545828255.00
$37,565,817,436	-1	$37565817436.00

Esempio 2 - Notazione MoneyThousandSep punto (.)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Dati forniti che hanno il loro campo monetario in formato testo con un punto '.' usato come separatore delle migliaia. Ogni record è inoltre preceduto dal simbolo "$".

Tenere presente che la variabile di sistema `MoneyFormat` definisce il dollaro '$' come valuta predefinita.

Script di caricamento

```
SET MoneyThousandSep='.';
SET MoneyDecimalSep=',';
SET MoneyFormat='$###0.00;-$###0.00';

Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,'$10.000.000.441'
01/02/2022,2,'$21.237.492.432'
01/03/2022,3,'$249.475.336'
01/04/2022,4,'$24.313.369.837'
01/05/2022,5,'$7.873.578.754'
01/06/2022,6,'$24.313.884.663'
01/07/2022,7,'$545.883.436'
01/08/2022,8,'$35.545.828.255'
01/09/2022,9,'$37.565.817.436'
01/10/2022,10,'$3,454,343,566'
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`amount`.

Aggiungere le seguenti misure:

- `isNum(amount)`
- `sum(amount)`

Esaminare i risultati qui sotto, dimostrando la corretta interpretazione di tutti i valori utilizzando la notazione del punto '.' come separatore delle migliaia.

Il campo `amount` è stato interpretato correttamente per tutti i valori, ad eccezione di un valore che utilizzava una virgola ',' come separatore delle migliaia.

Tabella dei risultati

importo	=isNum(amount)	=Sum(amount)
Totali	0	$161645330590.00
$3,545,343,566	0	$0.00
$249.475.336	-1	$249475336.00
$545.883.436	-1	545883436.00
$7.873.578.754	-1	$7873578754.00
$10.000.000.441	-1	$10000000441.00
$21.237.492.432	-1	$21237492432.00
$24.313.884.663	-1	$24313884663.00
$24.313.884.663	-1	$24313884663.00
$35.545.828.255	-1	$35545828255.00
$37.565.817.436	-1	$37565817436.00

MonthNames

Il formato specificato sostituisce la convenzione usata per i nomi dei mesi nelle impostazioni locali.

Sintassi:

```
MonthNames
```

Quando si modifica la variabile, è necessario utilizzare ; per separare i singoli valori.

Esempi di funzioni

Esempio	Risultati

Esempio	Risultati
`Set MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';`	L'utilizzo della funzione `MonthNames` consente di definire i nomi dei mesi in inglese e nella loro forma abbreviata.
`Set MonthNames='Enero;Feb;Marzo;Abr;Mayo;Jun;Jul;Agosto;Set;Oct;Nov;Dic';`	L'utilizzo della funzione `MonthNames` consente di definire i nomi dei mesi in spagnolo e nella loro forma abbreviata.

La funzione `MonthNames` può essere utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
month (page 905)	Funzione script per restituire i valori definiti in `MonthNames` come valori di campo
Date (page 1223)	Funzione script per restituire i valori definiti in `MonthNames` come valori di campo in base all'argomento di formattazione fornito
LongMonthNames (page 232)	Valori nel formato lungo di `MonthNames`

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Variabili di sistema predefinite

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Un campo `date`.
- La definizione predefinita `MonthNames`.

Script di caricamento

```
SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';

Transactions:
LOAD
date,
Month(date) as monthname,
id,
amount
INLINE
[
date,id,amount
01/01/2022,1,1000.45
01/02/2022,2,2123.34
01/03/2022,3,4124.35
01/04/2022,4,2431.36
01/05/2022,5,4787.78
01/06/2022,6,2431.84
01/07/2022,7,2854.83
01/08/2022,8,3554.28
01/09/2022,9,3756.17
01/10/2022,10,3454.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `monthname`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	monthname	sum(amount)
01/01/2022	Jan	1000.45

date	monthname	sum(amount)
01/02/2022	Jan	2123.34
01/03/2022	Jan	4124.35
01/04/2022	Jan	2431.36
01/05/2022	Jan	4787.78
01/06/2022	Jan	2431.84
01/07/2022	Jan	2854.83
01/08/2022	Jan	3554.28
01/09/2022	Jan	3756.17
01/10/2022	Jan	3454.35

Viene utilizzata la definizione predefinita `MonthNames`. Nello script di caricamento, la funzione `Month` viene utilizzata con il campo `date` come argomento fornito.

Nella tabella dei risultati, l'output della funzione `Month` visualizza i giorni della settimana nel formato della definizione `MonthNames`.

Esempio 2 - Modifica della variabile di sistema

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Un campo `date`.
- La variabile `MonthNames` modificata per utilizzare i mesi abbreviati in spagnolo.

Script di caricamento

```
Set MonthNames='Enero;Feb;Marzo;Abr;Mayo;Jun;Jul;Agosto;Set;Oct;Nov;Dic';

Transactions:
LOAD
date,
month(date) as month,
id,
amount
INLINE
[
date,id,amount
01/01/2022,1,1000
02/01/2022,2,2123
03/01/2022,3,4124
```

```
04/01/2022,4,2431
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `monthname`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	monthname	sum(amount)
01/01/2022	Enero	1000.45
01/02/2022	Enero	2123.34
01/03/2022	Enero	4124.35
01/04/2022	Enero	2431.36
01/05/2022	Enero	4787.78
01/06/2022	Enero	2431.84
01/07/2022	Enero	2854.83
01/08/2022	Enero	3554.28
01/09/2022	Enero	3756.17
01/10/2022	Enero	3454.35

Nello script di caricamento, per prima cosa la variabile `MonthNames` viene modificata per elencare i mesi dell'anno abbreviati in spagnolo. La funzione `Month` viene utilizzata con il campo `date` come argomento fornito.

Nella tabella dei risultati, l'output della funzione `Month` visualizza i giorni della settimana nel formato della definizione `MonthNames`.

È importante ricordare che se la lingua per la variabile `MonthNames` viene modificata come in questo esempio, la variabile `LongMonthNames` conterrà comunque i mesi dell'anno in inglese. Se entrambe le variabili vengono utilizzate nell'applicazione, è necessario modificare la variabile `LongMonthNames`.

Esempio 3 – Funzione data

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Un campo `date`.
- La definizione predefinita `MonthNames`.

Script di caricamento

```
SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';

Transactions:
LOAD
date,
Month(date, 'MMM') as monthname,
id,
amount
INLINE
[
date,id,amount
01/01/2022,1,1000.45
01/02/2022,2,2123.34
01/03/2022,3,4124.35
01/04/2022,4,2431.36
01/05/2022,5,4787.78
01/06/2022,6,2431.84
01/07/2022,7,2854.83
01/08/2022,8,3554.28
01/09/2022,9,3756.17
01/10/2022,10,3454.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `monthname`

Creare questa misura:

```
=sum(amount)
```

Tabella dei risultati

date	monthname	sum(amount)
01/01/2022	Jan	1000.45
01/02/2022	Jan	2123.34
01/03/2022	Jan	4124.35
01/04/2022	Jan	2431.36

date	monthname	sum(amount)
01/05/2022	Jan	4787.78
01/06/2022	Jan	2431.84
01/07/2022	Jan	2854.83
01/08/2022	Jan	3554.28
01/09/2022	Jan	3756.17
01/10/2022	Jan	3454.35

Viene utilizzata la definizione predefinita `MonthNames`. Nello script di caricamento, la funzione `Date` viene utilizzata con il campo `date` come primo argomento. Il secondo argomento è `MMM`.

L'utilizzo della formattazione Qlik Sense converte i valori del primo argomento nel nome del mese corrispondente impostato nella variabile `MonthNames`. Questo viene mostrato nella tabella dei risultati, nei valori di campo per il campo creato `month`.

NumericalAbbreviation

L'abbreviazione numerica definisce l'abbreviazione da utilizzare per i prefissi in scala dei numeri, ad esempio M per mega o un milione (10^6) e µ per micro (10^{-6}).

Sintassi:

```
NumericalAbbreviation
```

La variabile `NumericalAbbreviation` deve essere impostata su una stringa contenente un elenco di coppie che definiscono le abbreviazioni, delimitate da punti e virgola. Ogni coppia di definizione abbreviazioni deve contenere la scala (l'esponente in base decimale) e l'abbreviazione separata da due punti, ad esempio `6:M` per i milioni.

L'impostazione predefinita è `'3:k;6:M;9:G;12:T;15:P;18:E;21:Z;24:Y;-3:m;-6:µ;-9:n;-12:p;-15:f;-18:a;-21:z;-24:y'`.

Esempi:

Questa impostazione cambia il prefisso delle migliaia in t e il prefisso dei miliardi in B. Ciò è utile in applicazioni finanziarie statunitensi, dove si utilizzano abbreviazioni come t$, M$ e B$.

```
Set NumericalAbbreviation='3:t;6:M;9:B;12:T;15:P;18:E;21:Z;24:Y;-3:m;-6:µ;-9:n;-12:p;-15:f;-18:a;-21:z;-24:y';
```

ReferenceDay

L'impostazione definisce quale giorno di gennaio impostare come giorno di riferimento per definire la settimana 1. In altre parole, questa impostazione stabilisce il numero di giorni della settimana 1 che devono essere inclusi a gennaio.

Sintassi:

```
ReferenceDay
```

`ReferenceDay` imposta il numero di giorni da includere nella prima settimana dell'anno. È possibile impostare `ReferenceDay` su qualsiasi valore compreso tra `1` e `7`. Qualsiasi valore al di fuori dell'intervallo `1-7` viene interpretato come il punto centrale della settimana (`4`), che equivale a `ReferenceDay` impostato su `4`.

Se non si seleziona un valore per l'impostazione `ReferenceDay`, il valore predefinito mostrerà `ReferenceDay=0`, che verrà interpretato come il punto centrale della settimana (`4`), come mostrato nella tabella dei valori `ReferenceDay` di seguito.

La funzione `ReferenceDay` viene spesso utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Variabile	Interazione
BrokenWeeks (page 210)	Se nell'app Qlik Sense vengono utilizzate settimane ininterrotte, verrà applicata l'impostazione della variabile `ReferenceDay`. Tuttavia, se vengono utilizzate settimane interrotte, la settimana 1 inizierà il 1 gennaio e terminerà contemporaneamente all'impostazione della variabile `FirstWeekDay` e ignorerà il contrassegno `ReferenceDay`.
FirstWeekDay (page 224)	Numero intero che definisce il giorno da utilizzare come primo giorno della settimana.

Qlik Sense consente di impostare i seguenti valori per `ReferenceDay`:

Valori ReferenceDay

Value	Giorno di riferimento
0 (predefinito)	January 4
1	January 1
2	Gennaio 2
3	January 3
4	January 4
5	January 5
6	January 6
7	January 7

Nell'esempio seguente il valore `ReferenceDay = 3` definisce il 3 gennaio come giorno di riferimento:

```
SET ReferenceDay=3; //(set January 3 as the reference day)
```

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi:

Se si desiderano le impostazioni ISO per le settimane e i numeri di settimana, assicurarsi di inserire nello script quanto segue:

```
Set FirstWeekDay=0;
Set BrokenWeeks=0;
Set ReferenceDay=4;    // Jan 4th is always in week 1
```

Se si desiderano le impostazioni USA, assicurarsi che nello script sia presente quanto segue:

```
Set FirstWeekDay=6;
Set BrokenWeeks=1;
Set ReferenceDay=1;     // Jan 1st is always in week 1
```

Esempio 1 - Script di caricamento con il valore predefinito; ReferenceDay=0

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- La variabile di sistema `ReferenceDay` è impostata su `0`.
- La variabile `BrokenWeeks` impostata su `0` che fa in modo che l'app utilizzi settimane ininterrotte.
- Un set di dati delle date dalla fine del 2019 all'inizio del 2020.

Script di caricamento

```
SET BrokenWeeks = 0;
SET ReferenceDay = 0;

Sales:
LOAD
date,
sales,
week(date) as week,
weekday(date) as weekday
Inline [
date,sales
12/27/2019,5000
12/28/2019,6000
12/29/2019,7000
12/30/2019,4000
12/31/2019,3000
01/01/2020,6000
```

```
01/02/2020,3000
01/03/2020,6000
01/04/2020,8000
01/05/2020,5000
01/06/2020,7000
01/07/2020,3000
01/08/2020,5000
01/09/2020,9000
01/10/2020,5000
01/11/2020,7000
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week`
- `weekday`

Tabella dei risultati

date	settimana	weekday
12/27/2019	52	Fri
12/28/2019	52	Sat
12/29/2019	1	Sun
12/30/2019	1	Mon
12/31/2019	1	Tue
01/01/2020	1	Wed
01/02/2020	1	Thu
01/03/2020	1	Fri
01/04/2020	1	Sat
01/05/2020	2	Sun
01/06/2020	2	Mon
01/07/2020	2	Tue
01/08/2020	2	Wed
01/09/2020	2	Thu
01/10/2020	2	Fri
01/11/2020	2	Sat

La settimana 52 si conclude sabato 28 dicembre. Poiché `ReferenceDay` richiede che il 4 gennaio sia incluso nella settimana 1, pertanto la settimana 1 inizia il 29 dicembre e si conclude sabato 4 gennaio.

Esempio: Variabile ReferenceDay impostata su 5

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- La variabile di sistema `ReferenceDay` è impostata su `5`.
- La variabile `BrokenWeeks` impostata su `0` che fa in modo che l'app utilizzi settimane ininterrotte.
- Un set di dati delle date dalla fine del 2019 all'inizio del 2020.

Script di caricamento

```
SET BrokenWeeks = 0;
SET ReferenceDay = 5;

Sales:
LOAD
date,
sales,
week(date) as week,
weekday(date) as weekday
Inline [
date,sales
12/27/2019,5000
12/28/2019,6000
12/29/2019,7000
12/30/2019,4000
12/31/2019,3000
01/01/2020,6000
01/02/2020,3000
01/03/2020,6000
01/04/2020,8000
01/05/2020,5000
01/06/2020,7000
01/07/2020,3000
01/08/2020,5000
01/09/2020,9000
01/10/2020,5000
01/11/2020,7000
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week`

- weekday

Tabella dei risultati

date	settimana	weekday
12/27/2019	52	Fri
12/28/2019	52	Sat
12/29/2019	53	Sun
12/30/2019	53	Mon
12/31/2019	53	Tue
01/01/2020	53	Wed
01/02/2020	53	Thu
01/03/2020	53	Fri
01/04/2020	53	Sat
01/05/2020	1	Sun
01/06/2020	1	Mon
01/07/2020	1	Tue
01/08/2020	1	Wed
01/09/2020	1	Thu
01/10/2020	1	Fri
01/11/2020	1	Sat

La settimana 52 si conclude sabato 28 dicembre. La variabile `BrokenWeeks` fa in modo che l'app utilizzi settimane ininterrotte. Il valore del giorno di riferimento di `5` richiede che il 5 gennaio sia incluso nella settimana 1.

Tuttavia, questo giorno è otto giorni dopo la conclusione della settimana 52 dell'anno precedente. Pertanto, la settimana 53 inizia il 29 dicembre e si conclude il 4 gennaio. La settimana 1 inizia domenica 5 gennaio.

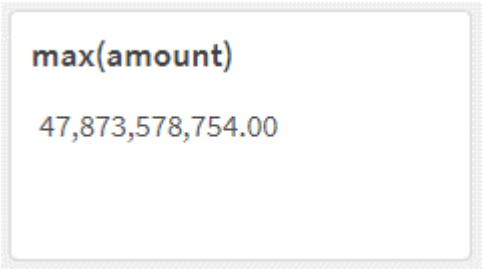
ThousandSep

Il separatore delle migliaia specificato sostituisce il simbolo di raggruppamento delle cifre del sistema operativo (impostazioni locali).

Sintassi:

```
ThousandSep
```

Oggetto Qlik Sense che utilizza la variabile `ThousandSep` (con separatore delle migliaia)



Le app Qlik Sense interpretano i campi di testo conformi a questa formattazione come numeri. Questa formattazione viene visualizzata negli oggetti grafico quando la proprietà di **formattazione numerica** di un campo numerico è impostata su **Numero**.

`ThousandSep` è utile quando si gestiscono sorgenti dati ricevute da più impostazioni regionali.

> *Se la variabile `ThousandSep` viene modificata dopo che gli oggetti sono già stati creati e formattati nell'applicazione, l'utente dovrà riformattare ogni campo pertinente deselezionando e poi riselezionando la proprietà di **formattazione numerica Numero**.*

I seguenti esempi mostrano i possibili utilizzi della variabile di sistema `ThousandSep`:

```
Set ThousandSep=','; //(for example, seven billion will be displayed as: 7,000,000,000)
```

```
Set ThousandSep=' '; //(for example, seven billion will be displayed as: 7 000 000 000)
```

I seguenti argomenti possono aiutarti a lavorare con questa funzione:

Argomenti correlati

Argomento	Descrizione
DecimalSep (page 222)	Nei casi di interpretazione di campi di testo, devono essere rispettate anche le impostazioni del separatore decimale fornite da questa funzione. Per la formattazione dei numeri, **DecimalSep** verrà utilizzato da Qlik Sense laddove necessario.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Variabili di sistema predefinite

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Utilizzo della definizione di variabile `ThousandSep` predefinita.

Script di caricamento

```
Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,10000000441
01/02/2022,2,21237492432
01/03/2022,3,41249475336
01/04/2022,4,24313369837
01/05/2022,5,47873578754
01/06/2022,6,24313884663
01/07/2022,7,28545883436
01/08/2022,8,35545828255
01/09/2022,9,37565817436
01/10/2022,10,3454343566
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`date`.
2. Aggiungere la misura seguente:
   `=sum(amount)`
3. Nel pannello delle proprietà, alla voce **Dati**, selezionare la misura.
4. Sotto **Formattazione numeri**, selezionare **Numero**.

Regolazione della formattazione dei numeri per una misura del grafico

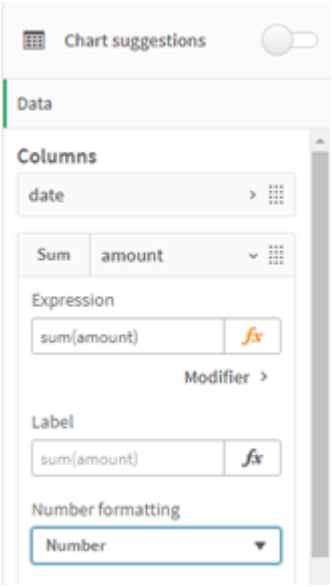


Tabella dei risultati

date	=sum(amount)
01/01/2022	10,000,000,441.00
01/02/2022	21,237,492,432.00
01/03/2022	41,249,475,336.00
01/04/2022	24,313,369,837.00
01/05/2022	47,873,578,754.00
01/06/2022	24,313,884,663.00
01/07/2022	28,545,883,436.00
01/08/2022	35,545,828,255.00
01/09/2022	37,565,817,436.00
01/10/2022	3,454,343,566.00

In questo esempio, viene utilizzata la definizione predefinita `ThousandSep`, impostata sul formato virgola (','). Nella tabella dei risultati, il formato del campo importo visualizza una virgola tra i raggruppamenti delle migliaia.

Esempio 2 - Modifica della variabile di sistema

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati del primo esempio, caricato in una tabella denominata `Transactions`.
- Modifica della definizione `ThousandSep`, all'inizio dello script, per visualizzare un carattere '*' come separatore delle migliaia. Si tratta di un esempio estremo, utilizzato esclusivamente per dimostrare la funzionalità della variabile.

La modifica utilizzata in questo esempio è estrema e non comunemente utilizzata, ma viene mostrata per dimostrare la funzionalità della variabile.

Script di caricamento

```
SET ThousandSep='*';

Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,10000000441
01/02/2022,2,21237492432
01/03/2022,3,41249475336
01/04/2022,4,24313369837
01/05/2022,5,47873578754
01/06/2022,6,24313884663
01/07/2022,7,28545883436
01/08/2022,8,35545828255
01/09/2022,9,37565817436
01/10/2022,10,3454343566
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`date`.
2. Aggiungere la misura seguente:
   `=sum(amount)`
3. Nel pannello delle proprietà, alla voce **Dati**, selezionare la misura.
4. Sotto **Formattazione numeri**, selezionare **Personalizzata**.

Tabella dei risultati

date	=sum(amount)
01/01/2022	10*000*000*441.00

date	=sum(amount)
01/02/2022	21*237*492*432.00
01/03/2022	41*249*475*336.00
01/04/2022	24*313*369*837.00
01/05/2022	47*873*578*754.00
01/06/2022	24*313*884*663.00
01/07/2022	28*545*883*436.00
01/08/2022	35*545*828*255.00
01/09/2022	37*565*817*436.00
01/10/2022	3*454*343*566.00

All'inizio dello script, la variabile di sistema `ThousandSep` viene modificata con un "*". Nella tabella dei risultati, il formato del campo importo può essere visualizzato come un "*" tra il raggruppamento delle migliaia.

Esempio 3 - Interpretazione del testo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata `Transactions`.
- Dati che hanno il loro campo numerico in formato testo, con una virgola come separatore delle migliaia.
- Utilizzo della variabile di sistema predefinita `ThousandSep`.

Script di caricamento

```
Transactions:
Load
date,
id,
amount
Inline
[
date,id,amount
01/01/2022,1,'10,000,000,441'
01/02/2022,2,'21,492,432'
01/03/2022,3,'4,249,475,336'
01/04/2022,4,'24,313,369,837'
01/05/2022,5,'4,873,578,754'
```

```
01/06/2022,6,'313,884,663'
01/07/2022,7,'2,545,883,436'
01/08/2022,8,'545,828,255'
01/09/2022,9,'37,565,817,436'
01/10/2022,10,'3,454,343,566'
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`date`.
2. Aggiungere la misura seguente:
   `=sum(amount)`
3. Nel pannello delle proprietà, alla voce **Dati**, selezionare la misura.
4. Sotto **Formattazione numeri**, selezionare **Numero**.
5. Aggiungere la seguente misura per valutare se il campo importo è un valore numerico o meno:
   `=isnum(amount)`

Tabella dei risultati

date	=sum(amount)	=isnum(amount)
01/01/2022	10,000,000,441.00	-1
01/02/2022	21,492,432.00	-1
01/03/2022	4,249,475,336.00	-1
01/04/2022	24,313,369,837.00	-1
01/05/2022	4,873,578,754.00	-1
01/06/2022	313,884,663.00	-1
01/07/2022	2,545,883,436.00	-1
01/08/2022	545,828,255.00	-1
01/09/2022	37,565,817,436.00	-1
01/10/2022	3*454*343*566.00	-1

Una volta caricati i dati, si può notare che il campo importo è stato interpretato da Qlik Sense come un valore numerico, grazie alla conformità dei dati alla variabile `ThousandSep`. Ciò è dimostrato dalla funzione `isnum()`, che valuta ogni voce a -1, o a `TRUE`.

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

TimeFormat

Il formato specificato sostituisce il formato dell'ora del sistema operativo (impostazioni locali).

Sintassi:

```
TimeFormat
```

Esempio:

```
Set TimeFormat='hh:mm:ss';
```

TimestampFormat

Il formato specificato sostituisce il formato della data e dell'ora del sistema operativo (impostazioni locali).

Sintassi:

```
TimestampFormat
```

Esempio:

Gli esempi seguenti utilizzano *1983-12-14T13:15:30Z* come data e ora per mostrare i risultati di diverse istruzioni **SET TimestampFormat**. Il formato data utilizzato è **YYYYMMDD** e il formato ora è **h:mm:ss TT**. Il formato data è specificato nell'istruzione **SET DateFormat** e il formato ora è specificato nell'istruzione **SET TimeFormat** all'inizio dello script di caricamento dei dati.

Risultati

Esempio	Risultato
`SET TimestampFormat='YYYYMMDD';`	19831214
`SET TimestampFormat='M/D/YY hh:mm:ss[.fff]';`	12/14/83 13:15:30
`SET TimestampFormat='DD/MM/YYYY hh:mm:ss[.fff]';`	14/12/1983 13:15:30
`SET TimestampFormat='DD/MM/YYYY hh:mm:ss[.fff] TT';`	14/12/1983 1:15:30 PM
`SET TimestampFormat='YYYY-MM-DD hh:mm:ss[.fff] TT';`	1983-12-14 01:15:30

Esempi: Script di caricamento

Esempio: Script di caricamento
Nel primo script di caricamento viene utilizzata l'istruzione *SET TimestampFormat='DD/MM/YYYY h:mm:ss[.fff] TT'*. Nel secondo script di caricamento, il formato di data e ora viene modificato con l'istruzione *SET TimestampFormat='MM/DD/YYYY hh:mm:ss[.fff]'*. I diversi risultati mostrano come funziona l'istruzione **SET TimeFormat** con diversi formati di dati temporali.

La tabella seguente mostra la serie di dati utilizzata negli script di caricamento che seguono. La seconda colonna della tabella mostra il formato di ogni indicatore di data e ora nella serie di dati. I primi cinque indicatori di data e ora seguono le regole ISO 8601, a differenza del sesto.

Serie di dati

Tabella che mostra i dati temporali utilizzati e il formato di ogni indicatore di data e ora nella serie di dati.

transaction_timestamp	time data format
2018-08-30	YYYY-MM-DD
20180830T193614.857	YYYYMMDDhhmmss.sss
20180830T193614.857+0200	YYYYMMDDhhmmss.sss±hhmm
2018-09-16T12:30-02:00	YYYY-MM-DDhh:mm±hh:mm
2018-09-16T13:15:30Z	YYYY-MM-DDhh:mmZ
9/30/18 19:36:14	M/D/YY hh:mm:ss

Nell'**editor caricamento dati** creare una nuova sezione, aggiungere lo script di esempio ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Script di caricamento

```
SET FirstWeekDay=0; SET BrokenWeeks=1; SET ReferenceDay=0; SET
DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun'; SET
LongDayNames='Monday;Tuesday;Wednesday;Thursday;Friday;Saturday;Sunday'; SET
DateFormat='YYYYMMDD'; SET TimestampFormat='DD/MM/YYYY h:mm:ss[.fff] TT';   Transactions: Load
*, Timestamp(transaction_timestamp, 'YYYY-MM-DD hh:mm:ss[.fff]') as LogTimestamp ;   Load *
Inline [ transaction_id, transaction_timestamp, transaction_amount, transaction_quantity,
discount, customer_id, size, color_code 3750, 2018-08-30, 12423.56, 23, 0,2038593, L, Red
3751, 20180830T193614.857, 5356.31, 6, 0.1, 203521, m, orange 3752, 20180830T193614.857+0200,
15.75, 1, 0.22, 5646471, S, blue 3753, 2018-09-16T12:30-02:00, 1251, 7, 0, 3036491, l, Black
3754, 2018-09-16T13:15:30Z, 21484.21, 1356, 75, 049681, xs, Red 3755, 9/30/18 19:36:14, -
59.18, 2, 0.333333333333333, 2038593, M, Blue ];
```

Risultati

Tabella Qlik Sense che mostra i risultati della variabile di interpretazione TimestampFormat utilizzata nello script di caricamento. L'ultimo indicatore di data e ora nella serie di dati non restituisce una data corretta.

transaction_id	transaction_timestamp	LogTimeStamp
3750	2018-08-30	2018-08-30 00:00:00
3751	20180830T193614.857	2018-08-30 19:36:14
3752	20180830T193614.857+0200	2018-08-30 17:36:14
3753	2018-09-16T12:30-02:00	2018-09-16 14:30:00
3754	2018-09-16T13:15:30Z	2018-09-16 13:15:30
3755	9/30/18 19:36:14	-

Lo script di caricamento successivo utilizza la stessa serie di dati, ma usando l'istruzione *SET TimestampFormat='MM/DD/YYYY hh:mm:ss[.fff]'* per consentire il riconoscimento del formato non conforme a ISO-8601 del sesto indicatore di data e ora.

Nell'**editor caricamento dati** sostituire lo script di esempio precedente con quello sottostante ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Script di caricamento

```
SET FirstWeekDay=0; SET BrokenWeeks=1; SET ReferenceDay=0; SET
DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun'; SET
LongDayNames='Monday;Tuesday;Wednesday;Thursday;Friday;Saturday;Sunday'; SET
DateFormat='YYYYMMDD'; SET TimestampFormat='MM/DD/YYYY hh:mm:ss[.fff]';   Transactions: Load
*, Timestamp(transaction_timestamp, 'YYYY-MM-DD hh:mm:ss[.fff]') as LogTimestamp ;   Load *
Inline [ transaction_id, transaction_timestamp, transaction_amount, transaction_quantity,
discount, customer_id, size, color_code 3750, 2018-08-30, 12423.56, 23, 0,2038593, L, Red
3751, 20180830T193614.857, 5356.31, 6, 0.1, 203521, m, orange 3752, 20180830T193614.857+0200,
15.75, 1, 0.22, 5646471, S, blue 3753, 2018-09-16T12:30-02:00, 1251, 7, 0, 3036491, l, Black
3754, 2018-09-16T13:15:30Z, 21484.21, 1356, 75, 049681, xs, Red 3755, 9/30/18 19:36:14, -
59.18, 2, 0.333333333333333, 2038593, M, Blue ];
```

Risultati

Tabella Qlik Sense che mostra i risultati della variabile di interpretazione TimestampFormat utilizzata nello script di caricamento.

transaction_id	transaction_timestamp	LogTimeStamp
3750	2018-08-30	2018-08-30 00:00:00
3751	20180830T193614.857	2018-08-30 19:36:14
3752	20180830T193614.857+0200	2018-08-30 17:36:14
3753	2018-09-16T12:30-02:00	2018-09-16 14:30:00
3754	2018-09-16T13:15:30Z	2018-09-16 13:15:30
3755	9/30/18 19:36:14	2018-09-16 19:36:14

2.15 Variabili di Direct Discovery

Variabili di sistema di Direct Discovery

DirectCacheSeconds

È possibile impostare un limite di memorizzazione nella cache per i risultati della query Direct Discovery per le visualizzazioni. Una volta raggiunto tale limite, Qlik Sense svuota la cache quando vengono eseguite nuove query Direct Discovery. Qlik Sense esegue una query sui dati sorgente per le selezioni e crea nuovamente la cache per il limite di tempo indicato. Il risultato per ciascuna combinazione di selezioni viene memorizzato nella cache in modo indipendente. Ciò significa che la cache viene aggiornata in modo indipendente per ciascuna selezione, quindi una selezione aggiorna la cache solo per i campi selezionati e

una seconda selezione la aggiorna per i campi rilevanti. Se la seconda selezione include i campi aggiornati nella prima selezione, questi non vengono aggiornati nuovamente nella cache se il limite di memorizzazione nella cache non è stato raggiunto.

La cache Direct Discovery non è applicabile alle visualizzazioni di **tabella**. Le selezioni di tabella interrogano la sorgente dei dati tutte le volte.

Il valore limite deve essere impostato in secondi. Il limite predefinito della cache è 1800 secondi (30 minuti).

Il valore utilizzato per **DirectCacheSeconds** è il valore impostato al momento dell'esecuzione dell'istruzione **DIRECT QUERY**. Tale valore non può essere modificato in fase di esecuzione.

Esempio:

```
SET DirectCacheSeconds=1800;
```

DirectConnectionMax
È possibile effettuare chiamate asincrone e parallele al database utilizzando la funzione di pooling delle connessioni. Di seguito è riportata la sintassi dello script di caricamento per impostare la funzionalità di pooling:

```
SET DirectConnectionMax=10;
```
L'impostazione numerica specifica il numero massimo di connessioni di database che il codice Direct Discovery deve utilizzare durante l'aggiornamento di un foglio. L'impostazione predefinita è 1.

Questa variabile deve essere utilizzata con attenzione. Se impostata su un valore superiore a 1 causa problemi di connessione a Microsoft SQL Server.

DirectUnicodeStrings
Direct Discovery è in grado di supportare la selezione dei dati Unicode estesi mediante l'utilizzo del formato standard SQL per i valori letterali delle stringhe di caratteri estese (N'<stringa estesa>') come richiesto da alcuni database (ad esempio SQL Server). È possibile attivare l'utilizzo della sintassi per Direct Discovery con la variabile di script **DirectUnicodeStrings**.

Se si imposta questa variabile su 'true', sarà possibile attivare l'utilizzo del marcatore di caratteri wide ANSI "N" standard davanti ai valori letterali di stringa. Non tutti i database supportano questo standard. L'impostazione predefinita è 'false'.

DirectDistinctSupport
Quando un valore di campo **DIMENSION** viene selezionato in un oggetto di Qlik Sense, viene generata una query per il database sorgente. Quando la query richiede il raggruppamento, Direct Discovery utilizza la parola chiave **DISTINCT** per selezionare solo valori univoci. Alcuni database, tuttavia, richiedono la parola chiave **GROUP BY**. Impostare **DirectDistinctSupport** su 'false' per generare **GROUP BY** anziché **DISTINCT** nelle query per i valori univoci.

```
SET DirectDistinctSupport='false';
```
Se DirectDistinctSupport è impostato su true, viene utilizzato **DISTINCT**. Se non è impostato su alcun valore, il comportamento predefinito prevede l'utilizzo di **DISTINCT**.

DirectEnableSubquery
Negli scenari con tabelle multiple ad alta cardinalità, è possibile generare sottoquery nella query SQL invece di generare una lunga clausola IN. Questa opzione viene attivata impostando **DirectEnableSubquery** su 'true'. Il valore predefinito è 'false'.

*Quando viene attivato **DirectEnableSubquery**, non è possibile caricare tabelle che non si trovino nella modalità Direct Discovery.*

```
SET DirectEnableSubquery='true';
```

Variabili di unione di query Teradata

L'unione di query Teradata è una funzione che consente alle applicazioni aziendali di collaborare con il database Teradata sottostante per migliorare la contabilità, l'assegnazione delle priorità e la gestione del carico di lavoro. L'utilizzo di un'unione di query permette di aggiungere a una query metadati come le credenziali utente.

Sono disponibili due variabili, entrambe sono stringhe che vengono valutate e inviate al database.

SQLSessionPrefix
Questa stringa viene inviata al momento della creazione di una connessione al database.

```
SET SQLSessionPrefix = 'SET QUERY_BAND = ' & Chr(39) & 'Who=' & OSuser() & ';' & Chr(39) & '
FOR SESSION;';
```

Se, ad esempio, **OSuser()** restituisce *WA\sbt*, questo sarà valutato come `SET QUERY_BAND = 'Who=WA\sbt;' FOR SESSION;`, che viene inviato al database quando viene creata la connessione.

SQLQueryPrefix
Questa stringa viene inviata per ciascuna singola query.

```
SET SQLSessionPrefix = 'SET QUERY_BAND = ' & Chr(39) & 'Who=' & OSuser() & ';' & Chr(39) & '
FOR TRANSACTION;';
```

Direct DiscoveryVariabili di carattere

DirectFieldColumnDelimiter
È possibile impostare il carattere utilizzato come delimitatore di campo nelle istruzioni **Direct Query** per i database che richiedono un carattere diverso dalla virgola come delimitatore di campo. Il carattere specificato deve essere racchiuso tra virgolette semplici nell'istruzione **SET**.

```
SET DirectFieldColumnDelimiter= '|'
```

DirectStringQuoteChar
È possibile specificare un carattere da utilizzare per racchiudere tra virgolette le stringhe in una query generata. Il valore predefinito è costituito dalle virgolette singole. Il carattere specificato deve essere racchiuso tra virgolette semplici nell'istruzione **SET**.

```
SET DirectStringQuoteChar= '"';
```

DirectIdentifierQuoteStyle
È possibile specificare l'utilizzo di virgolette non ANSI per racchiudere gli identificatori nelle query generate. In tal caso, le uniche virgolette non ANSI disponibili sono GoogleBQ. Il valore predefinito è ANSI. È possibile utilizzare lettere maiuscole, minuscole e miste (ANSI, ansi, Ansi).

```
SET DirectIdentifierQuoteStyle="GoogleBQ";
```
Ad esempio, le virgolette ANSI vengono utilizzate nella seguente istruzione **SELECT**:

```
SELECT [Quarter] FROM [qvTest].[sales] GROUP BY [Quarter]
```
Quando **DirectIdentifierQuoteStyle** è impostato su "GoogleBQ", l'istruzione **SELECT** utilizza le virgolette come indicato di seguito:

```
SELECT [Quarter] FROM [qvTest.sales] GROUP BY [Quarter]
```

DirectIdentifierQuoteChar
È possibile specificare un carattere per controllare le virgolette che racchiudono gli identificatori in una query generata. L'impostazione può includere uno o due caratteri: ad esempio, virgolette doppie o parentesi quadre. L'impostazione predefinita è costituita dalle virgolette doppie.

```
SET DirectIdentifierQuoteChar='[]';
SET DirectIdentifierQuoteChar='``';
SET DirectIdentifierQuoteChar=' ';
SET DirectIdentifierQuoteChar='""';
```

DirectTableBoxListThreshold
Quando si utilizzano i campi Direct Discovery in una visualizzazione di **tabella**, viene impostata una soglia per limitare il numero di righe visualizzate. La soglia predefinita è 1000 record. L'impostazione predefinita della soglia può essere modificata impostando la variabile **DirectTableBoxListThreshold** nello script di caricamento. Ad esempio:

```
SET DirectTableBoxListThreshold=5000;
```
L'impostazione della soglia è valida solo per le visualizzazioni di **tabella** che contengono campi Direct Discovery. Le visualizzazioni di **tabella** che contengono solo campi in memoria non sono limitate dall'impostazione **DirectTableBoxListThreshold**.

Nella visualizzazione di **tabella** non viene visualizzato alcun campo finché la selezione non include un numero di record inferiore al limite di soglia.

Variabili di interpretazione numerica Direct Discovery

DirectMoneyDecimalSep
Il separatore decimale definito sostituisce il simbolo decimale per la valuta nell'istruzione SQL generata per caricare i dati con Direct Discovery. Questo carattere deve corrispondere a quello utilizzato in **DirectMoneyFormat**.

Il valore predefinito è `'.'`

Esempio:

```
Set DirectMoneyDecimalSep='.';
```

DirectMoneyFormat
Il simbolo definito sostituisce il formato della valuta nell'istruzione SQL generata per caricare i dati con Direct Discovery. Il simbolo della valuta per il separatore delle migliaia non deve essere incluso.

Il valore predefinito è `'#.0000'`

Esempio:

```
Set DirectMoneyFormat='#.0000';
```

DirectTimeFormat
Il formato dell'ora definito sostituisce quello dell'istruzione SQL generata per caricare i dati con Direct Discovery.

Esempio:

```
Set DirectTimeFormat='hh:mm:ss';
```

DirectDateFormat
Il formato della data definito sostituisce quello dell'istruzione SQL generata per caricare i dati con Direct Discovery.

Esempio:

```
Set DirectDateFormat='MM/DD/YYYY';
```

DirectTimeStampFormat
Il formato definito sostituisce il formato di data e ora dell'istruzione SQL generata per caricare i dati con Direct Discovery.

Esempio:

```
Set DirectTimestampFormat='M/D/YY hh:mm:ss[.fff]';
```

2.16 Variabili di errore

I valori di tutte le variabili di errore vengono mantenuti anche dopo l'esecuzione dello script. La prima variabile, ErrorMode, è l'input dell'utente, mentre le ultime tre sono l'output di Qlik Sense con informazioni sugli errori rilevati nello script.

Prospetto delle variabili di errore

Ciascuna variabile è descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della variabile nella sintassi per accedere immediatamente ai dettagli per tale variabile specifica.

Per ulteriori informazioni sulle variabili, fare riferimento alla Guida in linea di Qlik Sense.

ErrorMode
Questa variabile di errore determina quale azione Qlik Sense deve intraprendere in caso di errore durante l'esecuzione dello script.

```
ErrorMode
```

ScriptError

Questa variabile di errore restituisce il codice di errore nell'istruzione dell'ultimo script eseguito.

```
ScriptError
```

ScriptErrorCount

Questa variabile di errore restituisce il numero totale di istruzioni che hanno provocato errori durante l'esecuzione dello script attuale. Questa variabile viene sempre reimpostata su 0 all'inizio dell'esecuzione dello script.

```
ScriptErrorCount
```

ScriptErrorList

Questa variabile di errore conterrà un elenco concatenato di tutti gli errori di script che si sono verificati durante l'ultima esecuzione dello script. Ogni errore è separato da un avanzamento riga.

```
ScriptErrorList
```

ErrorMode

Questa variabile di errore determina quale azione Qlik Sense deve intraprendere in caso di errore durante l'esecuzione dello script.

Sintassi:

```
ErrorMode
```

Argomenti:

Argomenti

Argomento	Descrizione
ErrorMode=1	Impostazione predefinita. L'esecuzione dello script viene interrotta e viene richiesto un intervento da parte dell'utente (modalità non batch).
ErrorMode =0	Qlik Sense ignora l'errore e prosegue nell'esecuzione dello script passando all'istruzione successiva.
ErrorMode =2	Qlik Sense restituirà immediatamente un messaggio di errore "Esecuzione dello script non riuscita…", senza richiedere prima alcuna azione all'utente.

Esempio:

```
set ErrorMode=0;
```

ScriptError

Questa variabile di errore restituisce il codice di errore nell'istruzione dell'ultimo script eseguito.

Sintassi:

```
ScriptError
```

Questa variabile verrà reimpostata su 0 dopo ogni esecuzione corretta delle istruzioni dello script. Se si verifica un errore, verrà impostato su un codice di errore di Qlik Sense interno. Tali codici sono valori doppi con un componente numerico e uno testuale. Esistono i seguenti codici di errore:

Codici di errore script

Codice di errore	Descrizione
0	Nessun errore. Il testo valore duale è vuoto.
1	Errore generico.
2	Errore di sintassi.
3	Errore generico ODBC.
4	Errore generico OLE DB.
5	Errore database personalizzato generico.
6	Errore generico XML.
7	Errore generico HTML.
8	File non trovato.
9	Database non trovato.
10	Tabella non trovata.
11	Campo non trovato.
12	Formato file errato.
16	Errore semantica.

Esempio:

```
set ErrorMode=0;

LOAD * from abc.qvf;

if ScriptError=8 then

exit script;

//no file;

end if
```

ScriptErrorCount

Questa variabile di errore restituisce il numero totale di istruzioni che hanno provocato errori durante l'esecuzione dello script attuale. Questa variabile viene sempre reimpostata su 0 all'inizio dell'esecuzione dello script.

Sintassi:

```
ScriptErrorCount
```

ScriptErrorList

Questa variabile di errore conterrà un elenco concatenato di tutti gli errori di script che si sono verificati durante l'ultima esecuzione dello script. Ogni errore è separato da un avanzamento riga.

Sintassi:

```
ScriptErrorList
```

2 Espressioni nello script

Le espressioni possono essere utilizzate sia nelle istruzioni **LOAD** che nelle istruzioni **SELECT**. La sintassi e le funzioni descritte in questa sezione riguardano l'istruzione **LOAD** e non l'istruzione **SELECT** poiché quest'ultima viene interpretata dal driver ODBC e non da Qlik Sense. In ogni caso, la maggior parte dei driver ODBC è spesso in grado di interpretare molte delle funzioni descritte di seguito.

Le espressioni sono costituite da funzioni, campi e operatori, combinati in una sintassi.

Tutte le espressioni in uno script di Qlik Sense restituiscono un numero e/o una stringa, a seconda di quale dei due risultati è appropriato. Le funzioni logiche e gli operatori restituiscono 0 per False e -1 per True. Le conversioni da numero a stringa e viceversa sono implicite. Gli operatori logici e le funzioni interpretano 0 come False e tutto il resto come True.

La sintassi generale di un'espressione è:

Sintassi generale

Espressione	Campi	Operatore
expression ::= (constant	constant	|
expression ::= (constant	fieldref	|
expression ::= (constant	operator1 expression	|
expression ::= (constant	expression operator2 expression	|
expression ::= (constant	function	|
expression ::= (constant	(expression)	)

dove:

- **constant** è una stringa (un testo, una data o un'ora) racchiusa tra virgolette singole diritte o un numero. Le costanti sono scritte senza separatore delle migliaia e con un punto decimale come separatore decimale.
- **fieldref** è un nome di campo della tabella caricata.
- **operator1** è un operatore unario (che agisce su un'unica espressione, quella a destra).
- **operator2** è un operatore binario (che agisce su due espressioni, una per ogni lato).
- **function ::= functionname(parameters)**
- **parameters ::= expression { , expression }**

Il numero e i tipi dei parametri non sono arbitrari. Dipendono dal tipo di funzione utilizzata.

Le espressioni e le funzioni possono essere nidificate liberamente. Finché l'espressione restituisce un valore interpretabile, in Qlik Sense non verrà visualizzato alcun messaggio di errore.

3 Espressioni del grafico

Un'espressione grafico (visualizzazione) è una combinazione di funzioni, campi e operatori matematici (+ * / =) e altre misure. Le espressioni vengono utilizzate per elaborare i dati nell'app al fine di produrre un risultato visibile all'interno di una visualizzazione. il loro utilizzo non è limitato alle misure. È possibile creare visualizzazioni più dinamiche e avanzate con espressioni per titoli, sottotitoli, note a piè di pagina e persino dimensioni.

Questo significa, ad esempio, che al posto del titolo di una visualizzazione costituito da testo statico, è possibile utilizzare un'espressione il cui risultato cambia in base alle selezioni effettuate.

Per un riferimento dettagliato sulle funzioni degli script e dei grafici, vedere Sintassi dello script e funzioni grafiche.

3.1 Definizione dell'ambito di aggregazione

Vi sono in genere due fattori che concorrono a determinare quali record vengono utilizzati per definire il valore dell'aggregazione in un'espressione. Quando si utilizzano le visualizzazioni, tali fattori sono:

- Valore dimensionale (dell'aggregazione in un'espressione di grafico)
- Selezioni

Insieme, questi fattori definiscono l'ambito dell'aggregazione. È possibile incontrare situazioni in cui si desidera che il calcolo ignori la selezione, la dimensione o entrambe. Nelle funzioni per grafici è possibile ottenere questo risultato utilizzando il qualificatore TOTAL, Set Analysis o una combinazione dei due.

Aggregazione: Metodo e descrizione

Metodo	Descrizione
Qualificatore TOTAL	Utilizzando il qualificatore TOTAL all'interno della funzione di aggregazione viene ignorato il valore dimensionale. L'aggregazione verrà eseguita su tutti i valori di campo possibili. Il qualificatore **TOTAL** può essere seguito da un elenco di uno o più nomi di campo tra parentesi acute. Questi nomi di campo devono essere un sottogruppo delle variabili di dimensione del grafico. In questo caso, il calcolo verrà effettuato ignorando tutte le variabili di dimensione del grafico eccetto quelle elencate, ad esempio un valore verrà restituito per ogni combinazione di valori di campo nei campi delle dimensioni elencati. Anche i campi che non sono attualmente una dimensione in un grafico possono essere inclusi nell'elenco. Questo può essere utile in caso di dimensioni di gruppo, dove i campi di dimensione non sono fissi. Elencando tutte le variabili nel gruppo viene attivata la funzione in corrispondenza delle modifiche del livello di drill-down.
Set Analysis	Se si utilizza l'analisi di gruppo all'interno dell'aggregazione, la selezione verrà ignorata. L'aggregazione verrà eseguita su tutti i valori suddivisi nelle dimensioni.
Qualificatore TOTAL e Set Analysis	Se si utilizza il qualificatore **TOTAL** e Set Analysis all'interno dell'aggregazione, la selezione e le dimensioni verranno ignorate.

Qualificatore ALL	Se si utilizza il qualificatore **ALL** all'interno dell'aggregazione, la selezione e le dimensioni verranno ignorate. È possibile ottenere gli stessi risultati con l'istruzione {1} Set Analysis e il qualificatore **TOTAL** :
	
	`=sum(All Sales)`
	
	`=sum({1} Total Sales)`

Esempio: Qualificatore TOTAL

Nell'esempio seguente viene mostrato come TOTAL può essere utilizzato per calcolare una quota relativa. Presupponendo che sia stato selezionato Q2, se si utilizza TOTAL viene calcolata la somma di tutti i valori ignorando le dimensioni.

Esempio: Qualificatore totale

Year	Quarter	Sum(Amount)	Sum(TOTAL Amount)	Sum(Amount)/Sum(TOTAL Amount)
		3000	3000	100%
2012	Q2	1700	3000	56,7%
2013	Q2	1300	3000	43,3%

Per visualizzare i numeri sotto forma di percentuale, nel pannello delle proprietà relativo alla misura che si desidera visualizzare come valore percentuale, in ***Number formatting*** *selezionare* ***Number*** *e da* ***Formatting*** *scegliere* ***Simple*** *e uno dei formati %.*

Esempio: Analisi di gruppo

Nell'esempio seguente viene mostrato come Set Analysis può essere utilizzata per eseguire un confronto tra serie di dati prima che venga eseguita qualsiasi selezione. Presupponendo che sia stato selezionato Q2, se si utilizza Set Analysis con la definizione di gruppo {1} verrà calcolata la somma di tutti i valori ignorando eventuali selezioni ma operando una suddivisione in base alle dimensioni.

Esempio: Analisi di gruppo

Year	Quarter	Sum(Amount)	Sum({1} Amount)	Sum(Amount)/Sum({1} Amount)
		3000	10800	27,8%
2012	Q1	0	1100	0%
2012	Q3	0	1400	0%
2012	Q4	0	1800	0%
2012	Q2	1700	1700	100%
2013	Q1	0	1000	0%
2013	Q3	0	1100	0%
2013	Q4	0	1400	0%
2013	Q2	1300	1300	100%

Esempio: Qualificatore TOTAL e analisi di gruppo

Nell'esempio seguente viene mostrato come Set Analysis e il qualificatore TOTAL possono essere combinati per eseguire un confronto tra le serie di dati prima che venga eseguita qualsiasi selezione e per tutte le dimensioni. Presupponendo che sia stato selezionato Q2, se si utilizza Set Analysis con la definizione set {1} e il qualificatore TOTAL viene calcolata la somma di tutti i valori ignorando eventuali selezioni e le dimensioni.

Esempio: Qualificatore TOTAL e analisi di gruppo

Year	Quarter	Sum (Amount)	Sum({1} TOTAL Amount)	Sum(Amount)/Sum({1} TOTAL Amount)
		3000	10800	27,8%
2012	Q2	1700	10800	15,7%
2013	Q2	1300	10800	12%

Dati utilizzati negli esempi:

```
AggregationScope: LOAD * inline [ Year Quarter Amount 2012 Q1 1100 2012 Q2 1700 2012 Q3 1400
2012 Q4 1800 2013 Q1 1000 2013 Q2 1300 2013 Q3 1100 2013 Q4 1400] (delimiter is ' ');
```

3.2 Analisi di gruppo

Quando si effettua una selezione in un'app, si definisce un sottogruppo di record nei dati. Le funzioni di aggregazione, come `Sum()`, `Max()`, `Min()`, `Avg()` e `Count()`, vengono calcolate in base a questo sottogruppo.

In altre parole, la selezione dell'utente definisce l'ambito dell'aggregazione; definisce il set di record su cui vengono effettuati i calcoli.

Set Analysis offre un modo per definire un ambito diverso dal set di record definito dalla selezione corrente. Tale nuovo ambito può anche essere considerato come una selezione alternativa.

Ciò può essere utile se si desidera confrontare la selezione corrente con un particolare valore, ad esempio il valore per lo scorso anno o la quota di mercato globale.

Espressioni set

Le espressioni set possono essere utilizzate all'interno e all'esterno delle funzioni di aggregazione, racchiudendole tra parentesi graffe.

Esempio: Espressione set interna

```
Sum( {$<Year={2021}>} Sales )
```

Esempio: Espressione set esterna

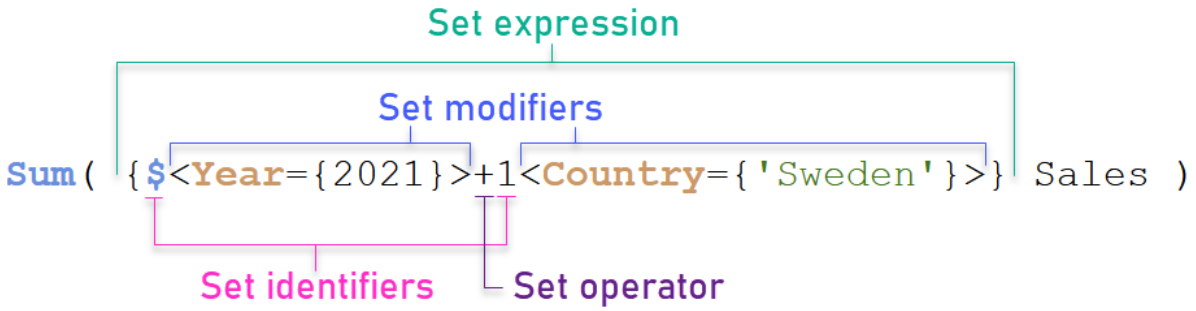
```
{<Year={2021}>} Sum(Sales) / Count(distinct Customer)
```

Un'espressione di gruppo è composta da una combinazione dei seguenti elementi:

- **Identifiers**. Un identificatore set rappresenta una selezione, definita altrove. Rappresenta anche un set specifico di record nei dati. Può trattarsi della selezione corrente, di una selezione da un preferito o di una selezione da uno stato alternato. Un'espressione di gruppo semplice è composta da un unico identificatore, come ad esempio il simbolo del dollaro, {$},che rappresenta tutti i record della selezione attuale.
  Esempi: `$`, `1`, `BookMark1`, `State2`
- **Operators**. Un operatore set può essere utilizzato per creare unioni, differenze o intersezioni tra diversi identificatori set. In questo modo, è possibile creare un sottogruppo o un sovragruppo delle selezioni definite dagli identificatori set.
  Esempi: `+`, `-`, `*`, `/`
- **Modifiers**. Un modificatore set può essere aggiunto all'identificatore set per modificarne la selezione. Un modificatore può anche essere utilizzato autonomamente, in modo che modifichi successivamente l'identificatore predefinito. Un modificatore deve essere racchiuso tra parentesi angolari `<…>`.
  Esempi: `<Year={2020}><Supplier={ACME}>`

Gli elementi sono combinati in modo da formare espressioni set.

Elementi in un'espressione set



L'espressione set di cui sopra, ad esempio, viene costruita dall'aggregazione `Sum(Sales)`.

Il primo operando restituisce le vendite per l'anno `2021` per la selezione corrente, indicato dall'identificatore set `$` e dal modificatore contenente la selezione dell'anno `2021`. Il secondo operando restituisce `Sales` per `Sweden`, e ignora la selezione corrente, indicata dall'identificatore set `1`.

Infine, l'espressione restituisce un set composto dai record appartenenti a uno qualsiasi dei due operandi set, come indicato dall'operatore set `+`.

Esempi

Esempi che combinano gli elementi espressione set sopra disponibili nei seguenti argomenti:

Set naturali

In genere, un'espressione set rappresenta sia un set di record nel modello dati, sia una selezione che definisce tale sottogruppo di dati. In questo caso, il set viene definito set naturale.

Gli identificatori set, con o senza modificatori set, rappresentano sempre set naturali.

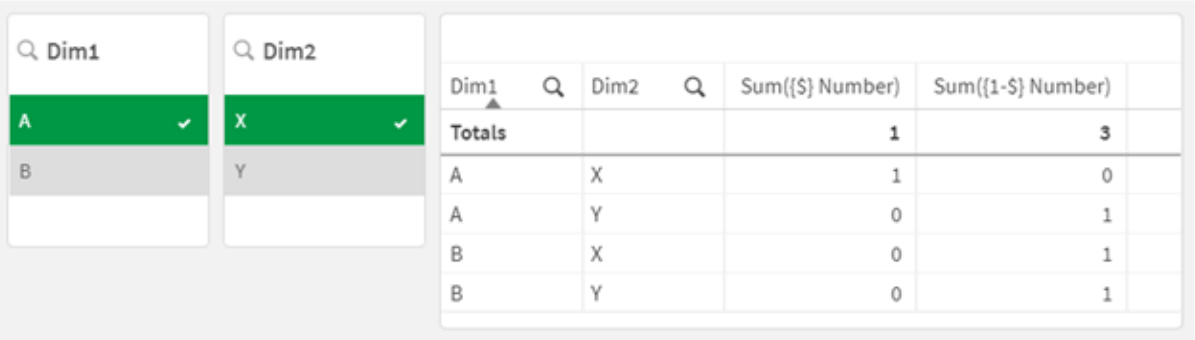
Tuttavia, un'espressione set che utilizza operatori set rappresenta anche un sottogruppo dei record, ma generalmente può comunque non essere descritta usando una selezione di valori di campo. Tale espressione rappresenta un set non naturale.

Ad esempio, il set fornito da `{1-$}` non può essere sempre definito da una selezione. Non rappresenta dunque un set naturale. Ciò può essere mostrato caricando i seguenti dati, aggiungendoli a una tabella ed effettuando quindi selezioni usando le caselle di filtro.

```
Load * Inline
[Dim1, Dim2, Number
A, X, 1
A, Y, 1
B, X, 1
B, Y, 1];
```

Effettuando selezioni per `Dim1` e `Dim2`, si ottiene la visualizzazione mostrata nella tabella seguente.

Tabella con set naturali e non naturali



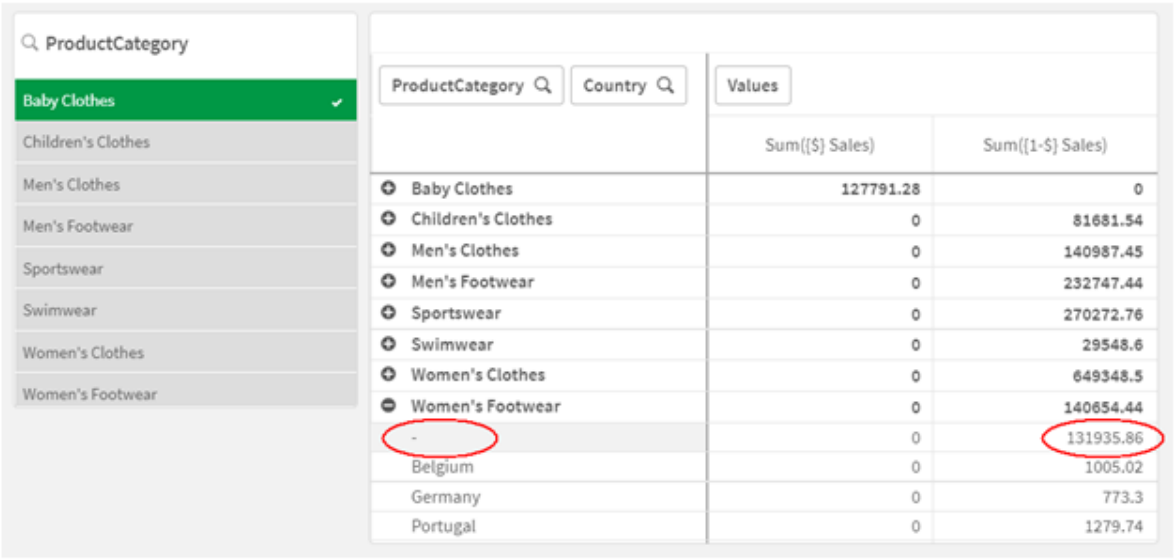
L'espressione set è la prima misura che utilizza un set naturale; corrisponde alla selezione effettuata `{$}`:

La seconda misura è diversa. Utilizza `{1-$}`. Non è possibile effettuare una selezione corrispondente a questo set, pertanto rappresenta un set non naturale.

Tale distinzione ha una serie di conseguenze:

- I modificatori set possono essere applicati solo agli identificatori set. Non possono essere applicati a un'espressione set arbitraria. Ad esempio, non è possibile utilizzare un'espressione set come:
  `{ (BM01 * BM02) <Field={x,y}> }`
  Qui, le normali parentesi (tonde) implicano che l'intersezione tra `BM01` e `BM02` debba essere valutata prima che sia possibile applicare il modificatore set. Il motivo è l'assenza di un set di elementi modificabili.
- Non è possibile utilizzare set non naturali all'interno delle funzioni di elementi `P()` e `E()`. Tali funzioni restituiscono un set di elementi, ma non è possibile dedurre il set di elementi da un set non naturali.
- Una misura che utilizza un set non naturale non sempre può essere attribuita al giusto valore dimensionale se il modello dati presenta molte tabelle. Ad esempio, nel grafico seguente, alcuni numeri esclusi sulle vendite sono attribuiti al giusto Country, mentre altri hanno NULL come Country.

Grafico con set non naturale



Il fatto che l'assegnazione venga effettuata correttamente o meno dipende dal modello dati. In questo caso, il numero non può essere assegnato se appartiene a un paese escluso dalla selezione.

Identificatore	Descrizione
1	Rappresenta la serie completa di tutti i record nell'applicazione, indipendentemente dalle selezioni eseguite.
$	Rappresenta i record della selezione attuale. L'espressione di gruppo **{$}** equivale pertanto alla non dichiarazione di un'espressione di gruppo.
$1	Rappresenta la selezione precedente. $2 rappresenta la selezione precedente, con uno scarto di uno, e così via.
$_1	Rappresenta la selezione successiva (in avanti). $_2 rappresenta la selezione successiva, con uno scarto di uno, e così via.
BM01	È possibile utilizzare qualsiasi ID o nome del preferito.
MyAltState	È possibile fare riferimento alle selezioni effettuate in uno stato alternativo utilizzando il nome del relativo stato.

Esempio	Risultato
sum ({1} Sales)	Restituisce le vendite totali per l'app, ignorando le selezioni, ma non la dimensione.
sum ({$} Sales)	Restituisce le vendite per la selezione attuale, vale a dire lo stesso di sum(Sales).
sum ({$1} Sales)	Restituisce le vendite per la selezione precedente.

Esempio	Risultato
sum ({BM01} Sales)	Restituisce le vendite per il preferito denominato *BM01*.

Esempio	Risultato
sum({$<OrderDate = DeliveryDate>} Sales)	Restituisce le vendite per la selezione attuale in cui OrderDate = DeliveryDate.
sum({1<Region = {US}>} Sales)	Restituisce le vendite per la regione US ignorando la selezione attuale.
sum({$<Region = >} Sales)	Restituisce le vendite per la selezione, da cui è stata però rimossa la selezione in '*Region*'.
sum({<Region = >} Sales)	Restituisce lo stesso risultato dell'esempio precedente. Quando il gruppo da modificare viene omesso, viene utilizzato $.
sum({$<Year={2000}, Region={"U*"}>} Sales)	Restituisce le vendite per la selezione corrente, ma con nuove selezioni sia in *Year* che in *Region*.

Identificatori set

Un identificatore set rappresenta un set di record nei dati; tutti i dati o un sottogruppo di essi. Rappresenta il set di record definiti da una selezione. Può trattarsi della selezione corrente, di tutti i dati (nessuna selezione), di una selezione da un preferito, o di una selezione da uno stato alternato.

Nell'esempio `Sum( {$<Year = {2009}>} Sales )`, l'identificatore è il simbolo del dollaro: `$`. Ciò rappresenta la selezione corrente. Rappresenta anche tutti i possibili record. Questo set può quindi essere alterato dalla parte con il modificatore dell'espressione set: la selezione `2009` in `Year` viene aggiunta.

In un'espressione set più complessa, è possibile utilizzare insieme due identificatori con un operatore per formare un'unione, oppure un'intersezione dei due set di record.

La tabella seguente mostra alcuni identificatori comuni.

Esempi con identificatori comuni

Identificatore	Descrizione
`1`	Rappresenta la serie completa di tutti i record nell'applicazione, indipendentemente dalle selezioni eseguite.
`$`	Rappresenta i record della selezione corrente nello stato predefinito. L'espressione set `{$}` equivale pertanto in genere alla non dichiarazione di un'espressione set.
`$1`	Rappresenta la selezione precedente nello stato predefinito. `$2` rappresenta la selezione precedente con uno scarto di uno, e così via.

Identificatore	Descrizione
`$_1`	Rappresenta la prossima selezione (in avanti). `$_2` rappresenta la selezione successiva, con uno scarto di uno, e così via.
`BM01`	È possibile utilizzare qualsiasi ID o nome del preferito.
`AltState`	È possibile fare riferimento a uno stato alternato mediante il proprio nome stato.
`AltState::BM01`	Un preferito contiene le selezioni di tutti gli stati, ed è possibile creare un riferimento a un preferito specifico qualificandone il nome.

La tabella seguente mostra esempi con identificatori diversi.

Esempi con identificatori diversi

Esempio	Risultato
`Sum ({1} Sales)`	Restituisce le vendite totali per l'app, ignorando le selezioni, ma non la dimensione.
`Sum ({$} Sales)`	Restituisce le vendite per la selezione attuale, vale a dire lo stesso di `Sum(Sales)`.
`Sum ({$1} Sales)`	Restituisce le vendite per la selezione precedente.
`Sum ({BM01} Sales)`	Restituisce le vendite per il preferito denominato `BM01`.

Operatori set

Gli operatori set sono utilizzati per includere, escludere o intersecare serie di dati. Tutti gli operatori utilizzano i gruppi come operandi e restituiscono un gruppo come risultato.

È possibile utilizzare gli operatori set in due situazioni diverse:

- Per eseguire un'operazione set sugli identificatori set, rappresentando i set di record nei dati.
- Per eseguire un'operazione sui set di elementi, sui valori di campo o all'interno di un modificatore set.

La tabella seguente mostra gli operatori utilizzabili nelle espressioni set.

Operatori

Operatore	Descrizione
`+`	Unione. Questa operazione binaria restituisce un gruppo costituito dai record o dagli elementi che appartengono a uno qualsiasi dei due operandi set.
`-`	Esclusione. Questa operazione binaria restituisce un gruppo costituito dai record o dagli elementi che appartengono solo al primo dei due operandi set e non al secondo. Inoltre, se utilizzata come operatore unario, restituisce il gruppo complementare.

Operatore	Descrizione
*	Intersezione. Questa operazione binaria restituisce un gruppo costituito dai record o dagli elementi che appartengono a entrambi gli operandi set.
/	Differenza simmetrica (XOR). Questa operazione binaria restituisce un gruppo costituito dai record o dagli elementi che appartengono a uno dei due operandi set, ma non a entrambi.

La tabella seguente mostra esempi con operatori.

Esempi con operatori

Esempio	Risultato
`Sum ({1-$} Sales)`	Restituisce le vendite per tutto ciò che è escluso dalla selezione corrente.
`Sum ({$*BM01} Sales)`	Restituisce le vendite per l'intersezione tra la selezione e il preferito `BM01`.
`Sum ({-($+BM01)} Sales)`	Restituisce le vendite escluse dalla selezione e dal preferito `BM01`.
`Sum ({$<Year={2009}>+1<Country={'Sweden'}>} Sales)`	Restituisce le vendite per l'anno 2009 associate alle selezioni correnti e aggiunge la serie di dati completa associata al paese `Sweden` per tutti gli anni.
`Sum ({$<Country={"S*"}+{"*land"}>} Sales)`	Restituisce le vendite per i paesi che iniziano con `S` o terminano con `land`.

Modificatori set

Le espressioni set vengono utilizzate per definire l'ambito di un calcolo. La parte centrale dell'espressione set è il modificatore set che specifica una selezione. Ciò è utilizzabile per modificare la selezione utente, oppure la selezione nell'identificatore set, e il risultato definisce un nuovo ambito per il calcolo.

Il modificatore set è composto da uno o più nomi di campo, ciascuno seguito da una selezione che dovrebbe essere effettuata sul campo. Il modificatore è racchiuso tra parentesi angolari: `< >`

Ad esempio:

- `Sum ( {$<Year = {2015}>} Sales )`
- `Count ( {1<Country = {Germany}>} distinct OrderID )`
- `Sum ( {$<Year = {2015}, Country = {Germany}>} Sales )`

Set di elementi

Un set di elementi può essere definito usando quanto segue:

- Un elenco di valori
- Una ricerca
- Un riferimento a un altro campo
- Una funzione set

Se la definizione del set elementi viene omessa, il modificatore set cancellerà qualsiasi selezione in questo campo. Ad esempio:

```
Sum( {$<Year = >} Sales )
```

Esempi: Espressioni del grafico per i modificatori set basati su set di elementi

Esempi - espressioni del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare gli esempi di espressione del grafico in basso.

```
MyTable:
Load * Inline [
Country, Year, Sales
Argentina, 2014, 66295.03
Argentina, 2015, 140037.89
Austria, 2014, 54166.09
Austria, 2015, 182739.87
Belgium, 2014, 182766.87
Belgium, 2015, 178042.33
Brazil, 2014, 174492.67
Brazil, 2015, 2104.22
Canada, 2014, 101801.33
Canada, 2015, 40288.25
Denmark, 2014, 45273.25
Denmark, 2015, 106938.41
Finland, 2014, 107565.55
Finland, 2015, 30583.44
France, 2014, 115644.26
France, 2015, 30696.98
Germany, 2014, 8775.18
Germany, 2015, 77185.68
];
```

Espressioni del grafico

Generare una tabella in un foglio Qlik Sense con le seguenti espressioni del grafico.

Tabella - modificatori set basati su set di elementi

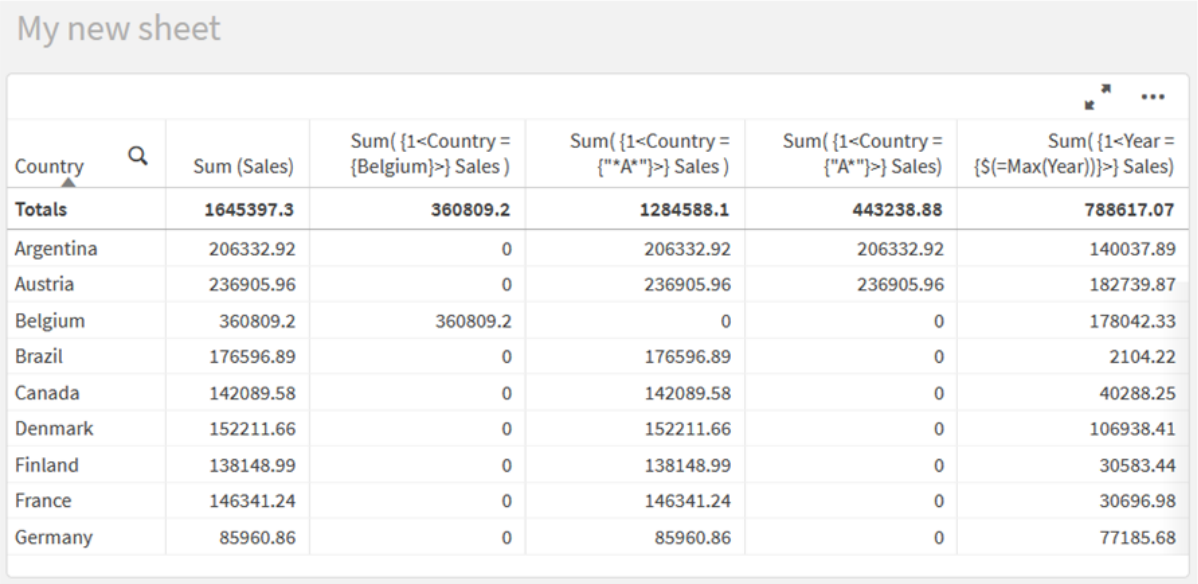
Paese	Sum(Sales)	Sum ({1<Country= {Belgium}>} Sales)	Sum ({1<Country= {"*A*"}>} Sales)	Sum ({1<Country= {"A*"}>} Sales)	Sum ({1<Year= {$(=Max (Year))}>} Sales)
Totali	1645397.3	360809.2	1284588.1	443238.88	788617.07
Argentina	206332.92	0	206332.92	206332.92	140037.89

Paese	Sum(Sales)	Sum ({1<Country={Belgium}>} Sales)	Sum ({1<Country={"*A*"}>} Sales)	Sum ({1<Country={"A*"}>} Sales)	Sum ({1<Year={$(=Max(Year))}>} Sales)
Austria	236905.96	0	236905.96	236905.96	182739.87
Belgio	360809.2	360809.2	0	0	178042.33
Brasile	176596.89	0	176596.89	0	2104.22
Canada	142089.58	0	142089.58	0	40288.25
Danimarca	152211.66	0	152211.66	0	106938.41
Finlandia	138148.99	0	138148.99	0	30583.44
France	146341.24	0	146341.24	0	30696.98
Germany	85960.86	0	85960.86	0	77185.68

Spiegazione

- Dimensioni:
  - `Country`
- Misure:
  - `Sum(Sales)`
    Sum `Sales` senza espressione set.
  - `Sum({1<Country={Belgium}>}Sales)`
    Selezionare `Belgium`, quindi sommare il corrispondente `Sales`.
  - `Sum({1<Country={"*A*"}>}Sales)`
    Selezionare tutti i paesi che hanno un `A`, quindi sommare il corrispondente `Sales`.
  - `Sum({1<Country={"A*"}>}Sales)`
    Selezionare tutti i paesi che iniziano con un `A`, quindi sommare il corrispondente `Sales`.
  - `Sum({1<Year={$(=Max(Year))}>}Sales)`
    Calcolare il valore `Max(Year)`, che corrisponde a `2015`, quindi sommare il valore corrispondente `Sales`.

Modificatori set basati su set di elementi


My new sheet

Country	Sum (Sales)	Sum({1<Country = {Belgium}>} Sales)	Sum({1<Country = {"*A*"}>} Sales)	Sum({1<Country = {"A*"}>} Sales)	Sum({1<Year = {$(=Max(Year))}>} Sales)
Totals	**1645397.3**	**360809.2**	**1284588.1**	**443238.88**	**788617.07**
Argentina	206332.92	0	206332.92	206332.92	140037.89
Austria	236905.96	0	236905.96	236905.96	182739.87
Belgium	360809.2	360809.2	0	0	178042.33
Brazil	176596.89	0	176596.89	0	2104.22
Canada	142089.58	0	142089.58	0	40288.25
Denmark	152211.66	0	152211.66	0	106938.41
Finland	138148.99	0	138148.99	0	30583.44
France	146341.24	0	146341.24	0	30696.98
Germany	85960.86	0	85960.86	0	77185.68

Valori elencati

L'esempio più comune di un set di elementi è quello basato su un elenco di valori di campo racchiusi tra parentesi graffe. Ad esempio:

- `{$<Country = {Canada, Germany, Singapore}>}`
- `{$<Year = {2015, 2016}>}`

Le parentesi graffe interne definiscono il set di elementi. I singoli valori sono separati da virgole.

Distinzione per virgolette e maiuscole/minuscole

Se i valori contengono caratteri vuoti o speciali, i valori devono essere racchiusi tra virgolette. Le virgolette singole indicano una corrispondenza letterale, con distinzione tra maiuscole e minuscole, con un valore di campo singolo. Le virgolette doppie implicano una corrispondenza senza distinzione tra maiuscole e minuscole in uno o più valori di campo. Ad esempio:

- `<Country = {'New Zealand'}>`
  Corrisponde solo a `New Zealand`.
- `<Country = {"New Zealand"}>`
  Corrisponde a `New Zealand`, `NEW ZEALAND` e `new zealand`.

Le date devono essere racchiuse tra virgolette e utilizzare il formato data del campo in questione. Ad esempio:

- `<ISO_Date = {'2021-12-31'}>`
- `<US_Date = {'12/31/2021'}>`
- `<UK_Date = {'31/12/2021'}>`

Le virgolette doppie possono essere sostituite dalle parentesi quadre o da accenti gravi.

Ricerche

I set di elementi possono anche essere creati attraverso le ricerche. Ad esempio:

- `<Country = {"C*"}>`
- `<Ingredient = {"*garlic*"}>`
- `<Year = {">2015"}>`
- `<Date = {">12/31/2015"}>`

I caratteri jolly possono essere utilizzati in una ricerca testuale: Un asterisco (`*`) rappresenta un qualsiasi numero di caratteri, mentre un punto interrogativo (`?`) rappresenta un carattere singolo. Per definire le ricerche numeriche possono essere utilizzati gli operatori relazionali.

Utilizzare sempre le virgolette doppie per le ricerche. Le ricerche non distinguono tra maiuscole e minuscole.

Espansioni con dollaro

Le espansioni con dollaro sono necessarie se si desidera utilizzare un calcolo all'interno del proprio set di elementi. Ad esempio, se si desidera ricercare solo l'ultimo anno possibile, è possibile usare:

```
<Year = {$(=Max(Year))}>
```

Valori selezionati in altri campi

I modificatori possono essere basati sui valori selezionati di un altro campo. Ad esempio:

```
<OrderDate = DeliveryDate>
```

Questo modificatore prenderà i valori selezionati da `DeliveryDate` e li applicherà come una selezione a `OrderDate`. Se sono presenti molti valori diversi, più di duecento, evitare di eseguire questa operazione, in quanto prevede l'uso di una quantità elevata di CPU.

Funzioni dei set di elementi

Il set di elementi può anche basarsi sulle funzioni set `P()` (possibili valori) e `E()` (valori esclusi).

Ad esempio, se si desidera selezionare i paesi in cui è stato venduto il prodotto `Cap`, è possibile utilizzare:

```
<Country = P({1<Product={Cap}>} Country)>
```

In modo simile, se si desidera selezionare i paesi in cui il prodotto `Cap` non è stato venduto, è possibile utilizzare:

```
<Country = E({1<Product={Cap}>} Country)>
```

Modificatori set con ricerche

È possibile creare set di elementi attraverso le ricerche con i modificatori set.

Ad esempio:

- `<Country = {"C*"}>`
- `<Year = {">2015"}>`
- `<Ingredient = {"*garlic*"}>`

Le ricerche devono sempre essere racchiuse tra virgolette doppie, parentesi quadre o accenti gravi. È possibile utilizzare un elenco con un mix di stringhe letterali (virgolette singole) e ricerche (virgolette doppie). Ad esempio:

```
<Product = {'Nut', "*Bolt", Washer}>
```

Ricerche testuali

Nelle ricerche testuali possono essere utilizzati i caratteri jolly e altri simboli:

- Un asterisco (*) rappresenterà un numero qualsiasi di caratteri.
- Un punto interrogativo (?) rappresenterà un carattere singolo.
- Un accento circonflesso (^) contrassegnerà l'inizio di una parola.

Ad esempio:

- `<Country = {"C*", "*land"}>`
  Mostra le corrispondenze di tutti i paesi che iniziano con un `C` o terminano con `land`.
- `<Country = {"*^z*"}>`
  Ciò corrisponderà a tutti i paesi che hanno una parola che inizia con `z`, come `New Zealand`.

Ricerche numeriche

È possibile effettuare ricerche numeriche usando questi operatori relazionali: `>`, `>=`, `<`, `<=`

Una ricerca numerica inizia sempre con uno di questi operatori. Ad esempio:

- `<Year = {">2015"}>`
  Corrisponde al 2016 e a tutti gli anni successivi.
- `<Date = {">=1/1/2015<1/1/2016"}>`
  Corrisponde a tutte le date durante il 2015. Notare la sintassi per la descrizione di un intervallo temporale tra due date. Il formato data deve corrispondere al formato data del campo in questione.

Ricerche tramite espressioni

È possibile utilizzare le ricerche tramite espressioni per effettuare ricerche più avanzate. Un'aggregazione viene quindi valutata per ciascun valore di campo nel campo di ricerca. Verranno selezionati tutti i valori per i quali l'espressione di ricerca restituisce valori veri.

Una ricerca tramite espressione inizia sempre con un segno uguale: `=`

Ad esempio:

```
<Customer = {"=Sum(Sales)>1000"}>
```

Ciò restituirà tutti i clienti con un valore delle vendite superiore a 1000. `Sum(Sales)` è calcolato sulla selezione corrente. Ciò significa che se si dispone di una selezione in un altro campo, come il campo `Product`, si otterranno i clienti che hanno soddisfatto la condizione di vendita solo per i prodotti selezionati.

Se si desidera che la condizione sia indipendente dalla selezione, è necessario utilizzare Set Analysis all'interno della stringa di ricerca. Ad esempio:

```
<Customer = {"=Sum({1} Sales)>1000"}>
```

Le espressioni dopo il segno uguale verranno interpretate come un valore booleano. Ciò significa che in caso di valutazione di qualcosa di diverso, qualsiasi numero diverso da zero verrà interpretato come vero, mentre lo zero e le stringhe vengono interpretate come false.

Quotes

Utilizzare le virgolette quando le stringhe di ricerca contengono caratteri vuoti o speciali. Le virgolette singole implicano una corrispondenza letterale, con distinzione tra maiuscole e minuscole, con un valore di campo singolo. Le virgolette doppie implicano una ricerca senza distinzione tra maiuscole e minuscole che corrisponde potenzialmente a più valori di campo.

Ad esempio:

- `<Country = {'New Zealand'}>`
  Corrispondenza solo con `New Zealand`.
- `<Country = {"New Zealand"}>`
  Corrispondenza con `New Zealand`, `NEW ZEALAND` e `new zealand`

Le virgolette doppie possono essere sostituite dalle parentesi quadre o da accenti gravi.

> *Nelle precedenti versioni di Qlik Sense non vi era distinzione tra virgolette singole e virgolette doppie e tutte le stringhe racchiuse tra virgolette venivano trattate come ricerche. Per questione di compatibilità, le app create con le versioni precedenti di Qlik Sense continueranno a funzionare come prima. Le app create con Qlik Sense November 2017 o versioni successive rispetteranno la differenza tra i due tipi di virgolette.*

Esempi: Le espressioni del grafico per i modificatori set con ricerche

Esempi - espressioni del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare gli esempi di espressione del grafico in basso.

```
MyTable:
Load
Year(Date) as Year,
Date#(Date,'YYYY-MM-DD') as ISO_Date,
Date(Date#(Date,'YYYY-MM-DD'),'M/D/YYYY') as US_Date,
Country, Product, Amount
Inline
[Date, Country, Product, Amount
2018-02-20, Canada, Washer, 6
2018-07-08, Germany, Anchor bolt, 10
```

```
2018-07-14, Germany, Anchor bolt, 3
2018-08-31, France, Nut, 2
2018-09-02, Czech Republic, Bolt, 1
2019-02-11, Czech Republic, Bolt, 3
2019-07-31, Czech Republic, Washer, 6
2020-03-13, France, Anchor bolt, 1
2020-07-12, Canada, Anchor bolt, 8
2020-09-16, France, Washer, 1];
```

Esempio 1: Espressioni del grafico con ricerche testuali

Generare una tabella in un foglio Qlik Sense con le seguenti espressioni del grafico.

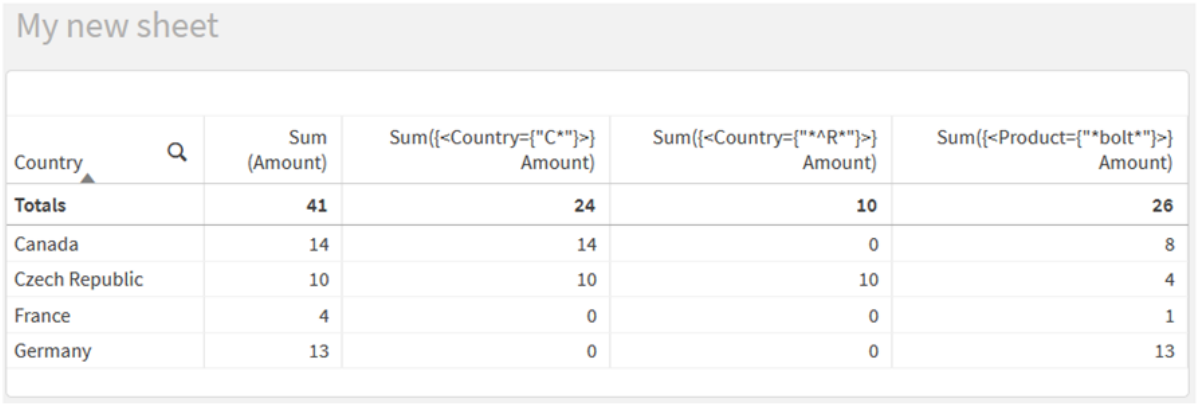
Tabella - Modificatori set con ricerche testuali

Paese	Sum (Amount)	Sum({<Country={"C*"}>} Amount)	Sum({<Country={"*^R*"}>} Amount)	Sum({<Product={"*bolt*"}>} Amount)
Totali	**41**	**24**	**10**	**26**
Canada	14	14	0	8
Repubblica Ceca	10	10	10	4
France	4	0	0	1
Germany	13	0	0	13

Spiegazione

- Dimensioni:
  - `Country`
- Misure:
  - `Sum(Amount)`
    Sum `Amount` senza espressione set.
  - `Sum({<Country={"C*"}>}Amount)`
    Sum `Amount` per tutti i paesi che iniziano con `C`, come `Canada` e `Czech Republic`.
  - `Sum({<Country={"*^R*"}>}Amount)`
    Sum `Amount` per tutti i paesi che presentano una parola che inizia con `R`, come `Czech Republic`.
  - `Sum({<Product={"*bolt*"}>}Amount)`
    Sum `Amount` per tutti i prodotti che contengono la stringa `bolt`, come `Bolt` e `Anchor bolt`.

Modificatori set con ricerche testuali


My new sheet

Country	Sum (Amount)	Sum({<Country={"C*"}>} Amount)	Sum({<Country={"*^R*"}>} Amount)	Sum({<Product={"*bolt*"}>} Amount)
Totals	**41**	**24**	**10**	**26**
Canada	14	14	0	8
Czech Republic	10	10	10	4
France	4	0	0	1
Germany	13	0	0	13

Esempio 2: Espressioni del grafico con ricerche numeriche

Generare una tabella in un foglio Qlik Sense con le seguenti espressioni del grafico.

Tabella - Modificatori set con ricerche numeriche

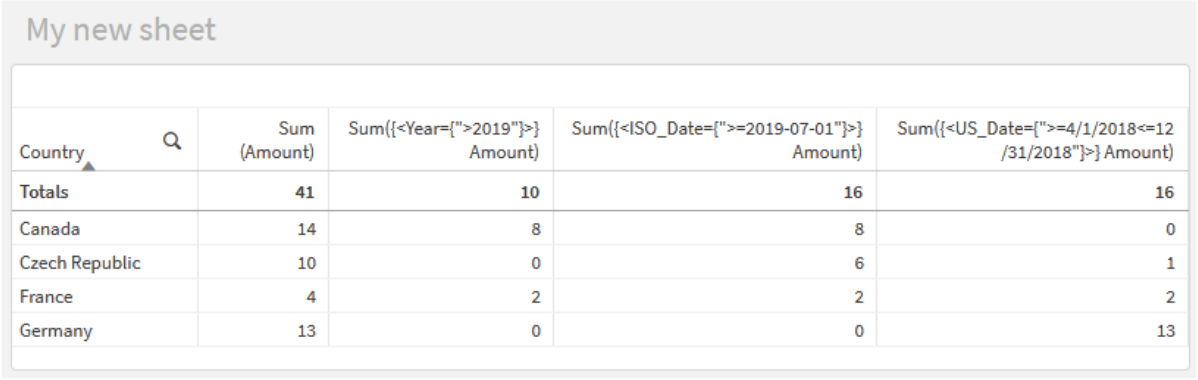
Paese	Sum (Amount)	Sum({<Year={">2019"}>} Amount)	Sum({<ISO_Date={">=2019-07-01"}>} Amount)	Sum({<US_Date={">=4/1/2018<=12/31/2018"}>} Amount)
Totali	**41**	**10**	**16**	**16**
Canada	14	8	8	0
Repubblica Ceca	10	0	6	1
France	4	2	2	2
Germany	13	0	0	13

Spiegazione

- Dimensioni:
  - `Country`
- Misure:
  - `Sum(Amount)`
    Sum `Amount` senza espressione set.
  - `Sum({<Year={">2019"}>}Amount)`
    Sum `Amount` per tutti gli anni dopo `2019`.
  - `Sum({<ISO_Date={">=2019-07-01"}>}Amount)`
    Sum `Amount` per tutte le date al o dopo il `2019-07-01`. Il formato della data nella ricerca deve corrispondere al formato del campo.
  - `Sum({<US_Date={">=4/1/2018<=12/31/2018"}>}Amount)`

Sum `Amount` per tutte le date dal `4/1/2018` al `12/31/2018`, incluse le date di inizio e fine. Il formato delle date nella ricerca deve corrispondere al formato del campo.

Modificatori set con ricerche numeriche


My new sheet

Country	Sum (Amount)	Sum({<Year={">2019"}>} Amount)	Sum({<ISO_Date={">=2019-07-01"}>} Amount)	Sum({<US_Date={">=4/1/2018<=12/31/2018"}>} Amount)
Totals	41	10	16	16
Canada	14	8	8	0
Czech Republic	10	0	6	1
France	4	2	2	2
Germany	13	0	0	13

Esempio 3: Espressioni del grafico con ricerche tramite espressione

Generare una tabella in un foglio Qlik Sense con le seguenti espressioni del grafico.

Table - Set modifiers with expression searches

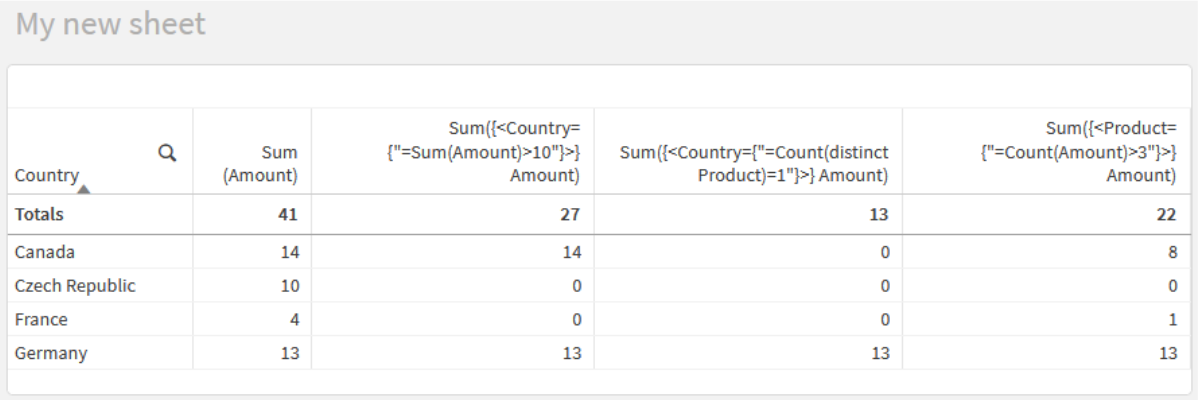
Country	Sum (Amount)	Sum({<Country={"=Sum (Amount)>10"}>} Amount)	Sum({<Country={"=Count(distinct Product)=1"}>} Amount)	Sum({<Product={"=Count (Amount)>3"}>} Amount)
Totals	**41**	**27**	**13**	**22**
Canada	14	14	0	8
Czech Republic	10	0	0	0
France	4	0	0	1
Germany	13	13	13	13

Spiegazione

- Dimensioni:
  - `Country`
- Misure:
  - `Sum(Amount)`
    Sum `Amount` senza espressione set.
  - `Sum({<Country={"=Sum(Amount)>10"}>}Amount)`
    Sum `Amount` per tutti i paesi che presentano una somma aggregata di `Amount` superiore a `10`.
  - `Sum({<Country={"=Count(distinct Product)=1"}>}Amount)`
    Sum `Amount` per tutti i paesi associati con esattamente un prodotto distinto.

- `Sum({<Product={"=Count(Amount)>3"}>}Amount)`
  Sum `Amount` per tutti i paesi che presentano più di tre transazioni nei dati.

Modificatori set con ricerche tramite espressione


My new sheet

Country	Sum (Amount)	Sum({<Country={"=Sum(Amount)>10"}>} Amount)	Sum({<Country={"=Count(distinct Product)=1"}>} Amount)	Sum({<Product={"=Count(Amount)>3"}>} Amount)
Totals	**41**	**27**	**13**	**22**
Canada	14	14	0	8
Czech Republic	10	0	0	0
France	4	0	0	1
Germany	13	13	13	13

Esempi	Risultati
sum({$-1<Product = {"*Internal*", "*Domestic*"}>} Sales)	Restituisce le vendite per la selezione corrente, escluse le transazioni relative ai prodotti con la stringa 'Internal' o 'Domestic' nel nome del prodotto.
sum({$<Customer = {"=Sum ({1<Year = {2007}>} Sales) > 1000000"}>} Sales)	Restituisce le vendite per la selezione corrente, ma con nuove selezione nel campo 'Customer': solo i clienti che nel 2007 hanno totalizzato un numero di vendite superiore a 1000000.

Modificatori di gruppo con espansioni con simbolo del dollaro

Le espansioni con simbolo del dollaro sono costrutti calcolati prima che l'espressione sia analizzata e valutata. Il risultato viene quindi inserito nell'espressione al posto di `$(…)`. Il calcolo dell'espressione viene quindi effettuato usando il risultato dell'espansione con simbolo del dollaro.

L'editor delle espressioni mostra un'anteprima dell'espansione con simbolo del dollaro, in modo che sia possibile verificare per cosa valuta la propria espressione con simbolo del dollaro.

Anteprima dell'espansione con simbolo del dollaro nell'editor delle espressioni

Utilizzare le espansioni con simbolo del dollaro quando si desidera utilizzare un calcolo all'interno del proprio set di elementi.

Ad esempio, se si desidera ricercare solo l'ultimo anno possibile, è possibile utilizzare la seguente costruzione:

```
<Year = {$(=Max(Year))}>
```

`Max(Year)` viene calcolato per primo e il risultato verrà inserito nell'espressione al posto di `$(…)`.

Il risultato dopo l'espansione con dollaro sarà un'espressione simile alla seguente:

```
<Year = {2021}>
```

L'espressione all'interno dell'espansione con simbolo del dollaro viene calcolata in base alla selezione corrente. Ciò significa che se si ha una selezione in un altro campo, il risultato dell'espressione ne verrà influenzato.

Se si desidera che il calcolo sia indipendente dalla selezione, utilizzare Set Analysis all'interno dell'espansione con simbolo del dollaro. Ad esempio:

```
<Year = {$(=Max({1} Year))}>
```

Stringhe

Quando si desidera che l'espansione con simbolo del dollaro risulti in una stringa, si applicano le normali regole associate alle citazioni. Ad esempio:

```
<Country = {'$(=FirstSortedValue(Country,Date)'}>
```

Il risultato dopo l'espansione con dollaro sarà un'espressione simile alla seguente:

```
<Country = {'New Zealand'}>
```

Si otterrà un errore di sintassi se non si utilizzano le virgolette.

Numeri

Quando si desidera che l'espansione con simbolo del dollaro risulti in un numero, assicurarsi che l'espansione abbia la stessa formattazione del campo. Questo significa che a volte è necessario disporre l'espressione in una funzione di formattazione.

Ad esempio:

```
<Amount = {$(=Num(Max(Amount), '###0.00'))}>
```

Il risultato dopo l'espansione con dollaro sarà un'espressione simile alla seguente:

```
<Amount = {12362.00}>
```

Usare un hash per forzare l'espansione a usare sempre il punto decimale e nessun separatore delle migliaia. Ad esempio:

```
<Amount = {$(#=Max(Amount))}>
```

Date

Quando si desidera che l'espansione con simbolo del dollaro risulti in una data, assicurarsi che l'espansione abbia la formattazione corretta. Questo significa che a volte è necessario disporre l'espressione in una funzione di formattazione.

Ad esempio:

```
<Date = {'$(=Date(Max(Date)))'}>
```

Il risultato dopo l'espansione con dollaro sarà un'espressione simile alla seguente:

```
<Date = {'12/31/2015'}>
```

Proprio come con le stringhe, è necessario utilizzare le virgolette corrette.

Un caso d'uso comune è quello di volere che il proprio calcolo sia limitato all'ultimo mese (o anno). Quindi è possibile utilizzare una ricerca numerica in combinazione con la funzione `AddMonths()`.

Ad esempio:

```
<Date = {">=$(=AddMonths(Today(),-1))"}>
```

Il risultato dopo l'espansione con dollaro sarà un'espressione simile alla seguente:

```
<Date = {">=9/31/2021"}>
```

Questa operazione sceglierà tutti gli eventi che si sono verificati nell'ultimo mese.

Esempio: Espressioni del grafico per i modificatori set con espansioni con simbolo del dollaro

Esempio: espressioni del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare gli esempi di espressione del grafico in basso.

```
Let vToday = Today();
MyTable:
Load
Year(Date) as Year,
Date#(Date,'YYYY-MM-DD') as ISO_Date,
Date(Date#(Date,'YYYY-MM-DD'),'M/D/YYYY') as US_Date,
Country, Product, Amount
Inline
[Date, Country, Product, Amount
2018-02-20, Canada, Washer, 6
2018-07-08, Germany, Anchor bolt, 10
2018-07-14, Germany, Anchor bolt, 3
2018-08-31, France, Nut, 2
2018-09-02, Czech Republic, Bolt, 1
2019-02-11, Czech Republic, Bolt, 3
2019-07-31, Czech Republic, Washer, 6
2020-03-13, France, Anchor bolt, 1
2020-07-12, Canada, Anchor bolt, 8
2021-10-15, France, Washer, 1];
```

Espressioni del grafico con espansioni con simbolo del dollaro

Generare una tabella in un foglio Qlik Sense con le seguenti espressioni del grafico.

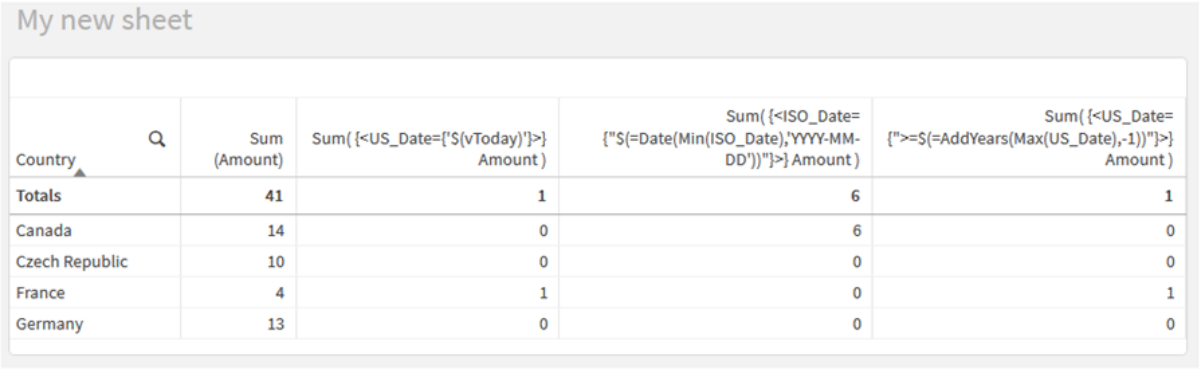
Tabella - Modificatori set con espansioni con simbolo del dollaro

Paese	Sum (Amount)	Sum({<US_ Date= {'$(vToday)'}>} Amount)	Sum({<ISO_Date= {"$(=Date(Min(ISO_ Date),'YYYY-MM-DD'))"}>} Amount)	Sum({<US_Date= {">=$(=AddYears (Max(US_Date),-1))"}>} Amount)
Totali	**41**	**1**	**6**	**1**
Canada	14	0	6	0
Repubblica Ceca	10	0	0	0
France	4	1	0	1
Germany	13	0	0	0

Spiegazione

- Dimensioni:
  - `Country`
- Misure:
  - `Sum(Amount)`
    Somma `Amount` senza espressione set.
  - `Sum({<US_Date={'$(vToday)'}>}Amount)`
    Sum `Amount` per tutti i record in cui `US_Date` è lo stesso della variabile `vToday`.
  - `Sum({<ISO_Date={"$(=Date(Min(ISO_Date),'YYYY-MM-DD'))"}>}Amount)`
    Sum `Amount` per tutti i record in cui `ISO_Date` è lo stesso del primo (più piccolo) `ISO_Date` possibile. La funzione `Date()` è necessaria per assicurare che il formato della data corrisponda a quello del campo.
  - `Sum({<US_Date={">=$(=AddYears(Max(US_Date),-1))"}>}Amount)`
    Sum `Amount` per tutti i record che presentano un `US_Date` dopo o nella data un anno prima del più recente (più grande) `US_Date` possibile. La funzione `AddYears()` restituirà una data nel formato specificato dalla variabile `DateFormat`, e questo deve corrispondere al formato del campo `US_Date`.

Modificatori di gruppo con espansioni con simbolo del dollaro


My new sheet

Country	Sum (Amount)	Sum({<US_Date={'$(vToday)'}>} Amount)	Sum({<ISO_Date={"$(=Date(Min(ISO_Date),'YYYY-MM-DD'))"}>} Amount)	Sum({<US_Date={">=$(=AddYears(Max(US_Date),-1))"}>} Amount)
Totals	**41**	**1**	**6**	**1**
Canada	14	0	6	0
Czech Republic	10	0	0	0
France	4	1	0	1
Germany	13	0	0	0

Esempi	Risultati
sum({$<Year = {$(#vLastYear)}>} Sales)	Restituisce le vendite per l'anno precedente in relazione alla selezione attuale. In questo esempio, una variabile vLastYear contenente l'anno pertinente viene utilizzata in un'espansione del simbolo del dollaro.
sum({$<Year = {$(#=Only(Year)-1)}>} Sales)	Restituisce le vendite per l'anno precedente in relazione alla selezione attuale. In questo esempio, viene utilizzata un'espansione del simbolo del dollaro per calcolare l'anno precedente.

Modificatori di gruppo con operatori di gruppo

Gli operatori set sono utilizzati per includere, escludere o intersecare vari set di elementi. Combinano i vari metodi per definire set di elementi.

Gli operatori corrispondono a quelli utilizzati per gli identificatori set.

Operatori

Operatore	Descrizione
+	Unione. Questa operazione binaria restituisce un gruppo costituito dai record o dagli elementi che appartengono a uno qualsiasi dei due operandi set.
-	Esclusione. Questa operazione binaria restituisce un gruppo costituito dai record o dagli elementi che appartengono solo al primo dei due operandi set e non al secondo. Inoltre, se utilizzata come operatore unario, restituisce il gruppo complementare.
*	Intersezione. Questa operazione binaria restituisce un gruppo costituito dai record o dagli elementi che appartengono a entrambi gli operandi set.
/	Differenza simmetrica (XOR). Questa operazione binaria restituisce un gruppo costituito dai record o dagli elementi che appartengono a uno dei due operandi set, ma non a entrambi.

Ad esempio, i due modificatori seguenti definiscono lo stesso set di valori di campo:

- `<Year = {1997, "20*"}>`
- `<Year = {1997} + {"20*"}>`

Entrambe le espressioni selezionano `1997` e gli anni che iniziano con `20`. In altre parole, si tratta dell'unione delle due condizioni.

Gli operatori set consentono anche definizioni più complesse. Ad esempio:

```
<Year = {1997, "20*"} - {2000}>
```

Questa espressione selezionerà gli stessi anni di quanto riportato sopra, ma in aggiunta esclude l'anno `2000`.

.

Esempi: Espressioni del grafico per i modificatori set con operatori set.

Esempi - espressioni del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare gli esempi di espressione del grafico in basso.

```
MyTable:
Load
Year(Date) as Year,
Date#(Date,'YYYY-MM-DD') as ISO_Date,
Date(Date#(Date,'YYYY-MM-DD'),'M/D/YYYY') as US_Date,
Country, Product, Amount
Inline
[Date, Country, Product, Amount
```

```
2018-02-20, Canada, Washer, 6
2018-07-08, Germany, Anchor bolt, 10
2018-07-14, Germany, Anchor bolt, 3
2018-08-31, France, Nut, 2
2018-09-02, Czech Republic, Bolt, 1
2019-02-11, Czech Republic, Bolt, 3
2019-07-31, Czech Republic, Washer, 6
2020-03-13, France, Anchor bolt, 1
2020-07-12, Canada, Anchor bolt, 8
2020-09-16, France, Washer, 1];
```

Espressioni del grafico

Generare una tabella in un foglio Qlik Sense con le seguenti espressioni del grafico.

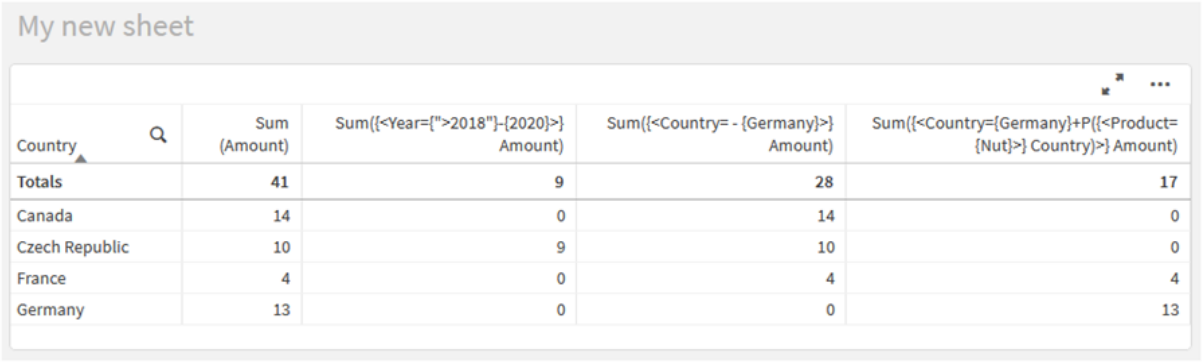
Tabella - Modificatori set con operatori set

Paese	Sum (Amount)	Sum({<Year={">2018"}-{2020}>} Amount)	Sum ({<Country=-{Germany}>} Amount)	Sum({<Country={Germany}+P({<Product={Nut}>}Country)>} Amount)
Totali	**41**	**9**	**28**	**17**
Canada	14	0	14	0
Repubblica Ceca	10	9	10	0
France	4	0	4	4
Germany	13	0	0	13

Spiegazione

- Dimensioni:
  - `Country`
- Misure:
  - `Sum(Amount)`
    Sum `Amount` senza espressione set.
  - `Sum({<Year={">2018"}-{2020}>}Amount)`
    Sum `Amount` per tutti gli anni dopo `2018`, tranne `2020`.
  - `Sum({<Country=-{Germany}>}Amount)`
    Sum `Amount` per tutti i paesi tranne la `Germany`. Notare l'operatore di esclusione unario.
  - `Sum({<Country={Germany}+P({<Product={Nut}>}Country)>}Amount)`
    Sum `Amount` per `Germany` e tutti i paesi associati al prodotto `Nut`.

Modificatori di gruppo con operatori di gruppo



Country	Sum (Amount)	Sum({<Year={">2018"}-{2020}>} Amount)	Sum({<Country= - {Germany}>} Amount)	Sum({<Country={Germany}+P({<Product={Nut}>} Country)>} Amount)
Totals	**41**	**9**	**28**	**17**
Canada	14	0	14	0
Czech Republic	10	9	10	0
France	4	0	4	4
Germany	13	0	0	13

Esempi	Risultati
sum({$<Product = Product + {OurProduct1} - {OurProduct2} >} Sales)	Restituisce le vendite per la selezione corrente, ma con il prodotto "OurProduct1" aggiunto all'elenco dei prodotti selezionati e "OurProduct2" rimosso dall'elenco dei prodotti selezionati.
sum({$<Year = Year + ({“20*”,1997} - {2000}) >} Sales)	Restituisce le vendite per la selezione corrente ma con selezioni aggiuntive nel campo "Year": 1997 e tutti gli anni che iniziano per "20", ad eccezione dell'anno 2000. Tenere presente che se 2000 è incluso nella selezione attuale, continuerà a essere incluso anche dopo la modifica.
sum({$<Year = (Year + {“20*”,1997}) - {2000} >} Sales)	Restituisce quasi lo stesso risultato dell'espressione precedente, ma l'anno 2000 verrà escluso, anche se inizialmente viene incluso nella selezione corrente. L'esempio dimostra l'importanza dello specifico utilizzo delle parentesi per definire un ordine di precedenza.
sum({$<Year = {“*”} - {2000}, Product = {“*bearing*”} >} Sales)	Restituisce le vendite per la selezione corrente, ma con una nuova selezione in "Year": tutti gli anni eccetto il 2000 e solo per i prodotti contenenti la stringa "bearing".

Modificatori set con operatori set impliciti

Il modo standard di scrivere le selezioni in un modificatore set consiste nell'utilizzare un segno uguale. Ad esempio:

```
Year = {">2015"}
```

L'espressione a destra del segno uguale nel modificatore set è chiamata set di elementi. Definisce un set di valori di campo distinti, in altre parole una selezione.

Questa notazione definisce una nuova selezione ignorando la selezione attuale nel campo. Pertanto, se l'identificatore set contiene una selezione in questo campo, la vecchia selezione verrà sostituita da quella nel set di elementi.

Quando si desidera basare la propria selezione sulla selezione corrente nel campo, è necessario usare un'espressione diversa

Per esempio, se si desidera rispettare la vecchia selezione e aggiungere il requisito che l'anno sia successivo al 2015, è possibile scrivere quanto segue:

```
Year = Year * {">2015"}
```

L'asterisco è un operatore set che definisce un'intersezione, quindi si otterrà l'intersezione tra la selezione corrente in `Year` e il requisito aggiuntivo che l'anno sia successivo al `2015`. Un modo alternativo per scriverlo è il seguente:

```
Year *= {">2015"}
```

Ovvero, l'operatore di assegnazione (`*=`) definisce implicitamente un'intersezione.

Allo stesso modo, le unioni, le esclusioni e le differenze simmetriche implicite possono essere definite utilizzando quanto segue: `+=`, `-=`, `/=`

Esempi: Espressioni del grafico per i modificatori set con operatori set impliciti

Esempi - espressioni del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare gli esempi di espressione del grafico in basso.

```
MyTable:
Load
Year(Date) as Year,
Date#(Date,'YYYY-MM-DD') as ISO_Date,
Date(Date#(Date,'YYYY-MM-DD'),'M/D/YYYY') as US_Date,
Country, Product, Amount
Inline
[Date, Country, Product, Amount
2018-02-20, Canada, Washer, 6
2018-07-08, Germany, Anchor bolt, 10
2018-07-14, Germany, Anchor bolt, 3
2018-08-31, France, Nut, 2
2018-09-02, Czech Republic, Bolt, 1
2019-02-11, Czech Republic, Bolt, 3
2019-07-31, Czech Republic, Washer, 6
2020-03-13, France, Anchor bolt, 1
2020-07-12, Canada, Anchor bolt, 8
2020-09-16, France, Washer, 1];
```

Espressioni del grafico con operatori set impliciti

Generare una tabella in un foglio Qlik Sense con le seguenti espressioni del grafico.

Selezionare `Canada` e `Czech Republic` da un elenco di paesi.

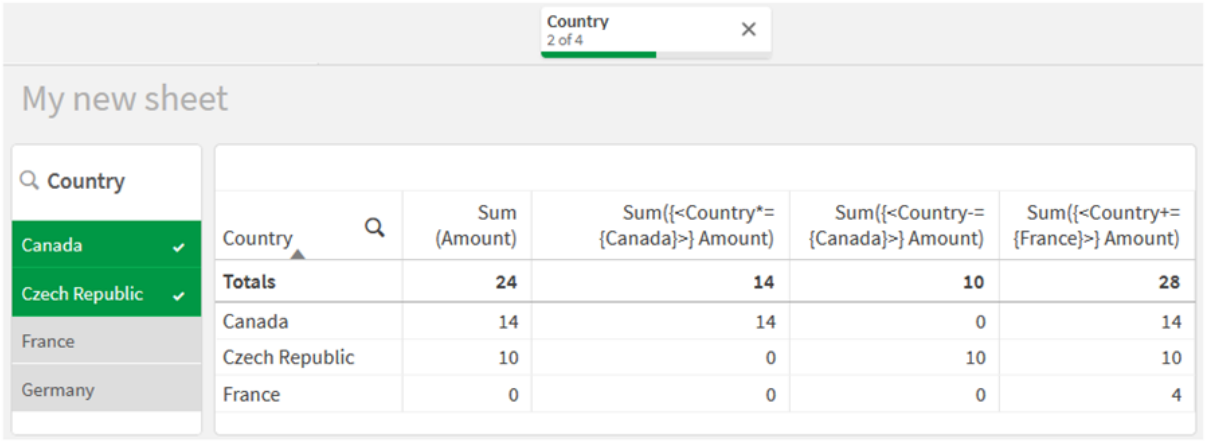
Tabella - Espressioni del grafico con operatori set impliciti

Paese	Sum (Amount)	Sum({<Country*= {Canada}>} Amount)	Sum({<Country-= {Canada}>} Amount)	Sum({<Country+= {France}>} Amount)
Totali	**24**	**14**	**10**	**28**
Canada	14	14	0	14
Repubblica Ceca	10	0	10	10
France	0	0	0	4

Spiegazione

- Dimensioni:
  - `Country`
- Misure:
  - `Sum(Amount)`
    Sum `Amount` per la selezione corrente. Notare che solo `Canada` e `Czech Republic` presentano valori diversi da zero.
  - `Sum({<Country*={Canada}>}Amount)`
    Sum `Amount` per la selezione corrente, intersecato con il requisito secondo cui il `Country` deve essere `Canada`. Se `Canada` non fa parte della selezione dell'utente, l'espressione set restituisce un set vuoto e la colonna avrà `0` su tutte le righe.
  - `Sum({<Country-={Canada}>}Amount)`
    Sum `Amount` per la selezione corrente, ma prima `Canada` viene escluso dalla selezione `Country`. Se `Canada` non fa parte della selezione utente, l'espressione set non cambierà alcun numero.
  - `Sum({<Country+={France}>}Amount)`
    Sum `Amount` per la selezione corrente, ma prima si aggiunge `France` alla selezione `Country`. Se `France` fa già parte della selezione utente, l'espressione set non cambierà alcun numero.

Modificatori set con operatori set impliciti.



Esempi	Risultati
sum({$<Product += {OurProduct1, OurProduct2} >} Sales)	Restituisce le vendite per la selezione attuale, ma utilizzando un'unione implicita per aggiungere i prodotti 'OurProduct1' e 'OurProduct2' all'elenco dei prodotti selezionati.
sum({$<Year += {“20*”,1997} - {2000} >} Sales)	Restituisce le vendite per la selezione attuale ma utilizzando un'unione implicita per aggiungere un numero di anni alla selezione: 1997 e tutti gli anni che iniziano per "20", ad eccezione dell'anno 2000. Tenere presente che se 2000 è incluso nella selezione attuale, continuerà a essere incluso anche dopo la modifica. Lo stesso di `<Year=Year + ({“20*”,1997}-{2000})>`.
sum({$<Product *= {OurProduct1} >} Sales)	Restituisce le vendite per la selezione attuale, ma solo per l'intersezione dei prodotti attualmente selezionati e del prodotto OurProduct1.

Modificatori set usando le funzioni set

A volte è necessario definire un set di valori di campo utilizzando una definizione di gruppo nidificata. Ad esempio, è possibile voler selezionare tutti i clienti che hanno portato un prodotto specifico, senza selezionare il prodotto.

In tali casi, usare le funzioni set di elementi `P()` e `E()`. Questi restituiscono set di elementi dei valori possibili e dei valori esclusi di un campo, rispettivamente. All'interno delle parentesi, è possibile specificare il campo in questione e un'espressione set che definisce l'ambito. Ad esempio:

```
P({1<Year = {2021}>} Customer)
```

Ciò restituirà il set di clienti che hanno avuto delle transazioni nel 2021. È quindi possibile utilizzarlo in un modificatore set. Ad esempio:

```
Sum({<Customer = P({1<Year = {2021}>} Customer)>} Amount)
```

Questa espressione set selezionerà tali clienti, ma non limiterà la restrizione della selezione al 2021.

Queste funzioni non possono essere utilizzate in altre espressioni.

In aggiunta, solo i set naturali possono essere utilizzati all'interno delle funzioni dei set di elementi. ossia un set di record che può essere definito mediante una semplice selezione.

Ad esempio, il set restituito da `{1-$}` non può essere sempre definito mediante una selezione e non è dunque un set naturale. L'utilizzo di queste funzioni su set non naturali restituirà risultati inattesi.

Esempi: Espressioni del grafico per i modificatori set usando funzioni set

Esempi - espressioni del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare gli esempi di espressione del grafico in basso.

```
MyTable:
Load
Year(Date) as Year,
Date#(Date,'YYYY-MM-DD') as ISO_Date,
Date(Date#(Date,'YYYY-MM-DD'),'M/D/YYYY') as US_Date,
Country, Product, Amount
Inline
[Date, Country, Product, Amount
2018-02-20, Canada, Washer, 6
2018-07-08, Germany, Anchor bolt, 10
2018-07-14, Germany, Anchor bolt, 3
2018-08-31, France, Nut, 2
2018-09-02, Czech Republic, Bolt, 1
2019-02-11, Czech Republic, Bolt, 3
2019-07-31, Czech Republic, Washer, 6
2020-03-13, France, Anchor bolt, 1
2020-07-12, Canada, Anchor bolt, 8
2020-09-16, France, Washer, 1];
```

Espressioni del grafico

Generare una tabella in un foglio Qlik Sense con le seguenti espressioni del grafico.

Tabella - Modificatori set usando le funzioni set

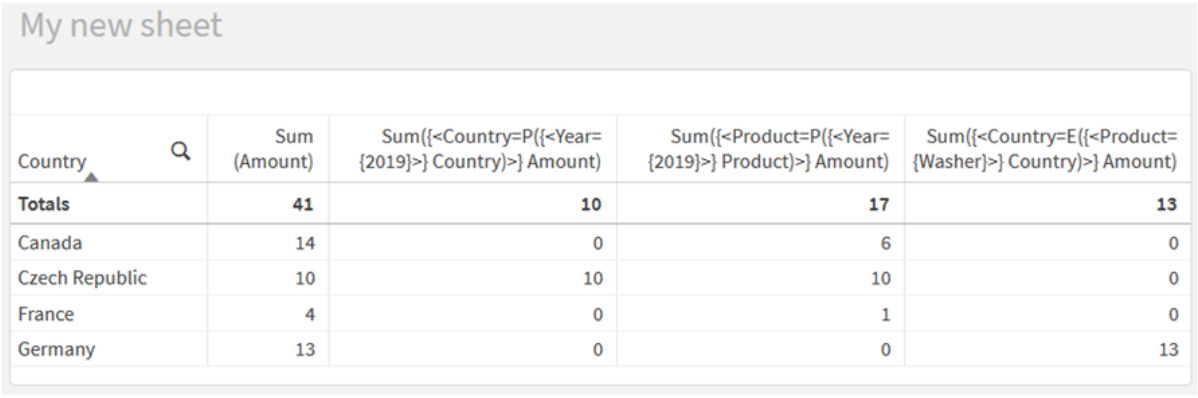
Paese	Sum (Amount)	Sum({<Country=P ({<Year= {2019}>}Country)>} Amount)	Sum({<Product=P ({<Year= {2019}>}Product)>} Amount)	Sum({<Country=E ({<Product= {Washer}>}Country)>} Amount)
Totali	**41**	**10**	**17**	**13**
Canada	14	0	6	0

Paese	Sum (Amount)	Sum({<Country=P({<Year={2019}>}Country)>} Amount)	Sum({<Product=P({<Year={2019}>}Product)>} Amount)	Sum({<Country=E({<Product={Washer}>}Country)>} Amount)
Totali	**41**	**10**	**17**	**13**
Repubblica Ceca	10	10	10	0
France	4	0	1	0
Germany	13	0	0	13

Spiegazione

- Dimensioni:
  - `Country`
- Misure:
  - `Sum(Amount)`
    Sum `Amount` senza espressione set.
  - `Sum({<Country=P({<Year={2019}>} Country)>} Amount)`
    Sum `Amount` per i paesi associati all'anno `2019`. Tuttavia, non limiterà il calcolo a `2019`.
  - `Sum({<Product=P({<Year={2019}>} Product)>} Amount)`
    Sum `Amount` per i prodotti associati all'anno `2019`. Tuttavia, non limiterà il calcolo a `2019`.
  - `Sum({<Country=E({<Product={Washer}>} Country)>} Amount)`
    Sum `Amount` per i paesi non associati al prodotto `Washer`.

Modificatori set usando le funzioni set



Esempi	Risultati
sum({$<Customer = P({1<Product= {'Shoe'}>} Customer)>} Sales)	Restituisce le vendite per la selezione corrente, ma solo i clienti che hanno acquistato il prodotto 'Shoe'. La funzione di elemento P( ) qui restituisce un elenco di possibili clienti; quelli che sono interessati dalla selezione 'Shoe' nel campo Product.
sum({$<Customer = P({1<Product= {'Shoe'}>})>} Sales)	Restituisce lo stesso risultato riportato in precedenza. Se il campo nella funzione di elemento viene omesso, la funzione restituirà i possibili valori del campo specificato nell'assegnazione esterna.
sum({$<Customer = P({1<Product= {'Shoe'}>} Supplier)>} Sales)	Restituisce le vendite per la selezione corrente, ma solo per i clienti che hanno fornito il prodotto 'Shoe', ovvero, laddove il cliente è anche un fornitore. La funzione di elemento P( ) qui restituisce un elenco di possibili fornitori; quelli che sono interessati dalla selezione 'Shoe' nel campo Product. L'elenco di fornitori viene utilizzato come selezione nel campo Customer.
sum({$<Customer = E({1<Product= {'Shoe'}>})>} Sales)	Restituisce le vendite per la selezione corrente, ma solo i clienti che non hanno mai acquistato il prodotto 'Shoe'. La funzione di elemento E( ) qui restituisce l'elenco di clienti esclusi; quelli che sono esclusi dalla selezione 'Shoe' nel campo Product.

Espressioni set interne ed esterne

Le espressioni set possono essere utilizzate all'interno e all'esterno delle funzioni di aggregazione, racchiudendole tra parentesi graffe.

Quando si utilizza un'espressione set all'interno di una funzione di aggregazione, l'aspetto può essere il seguente:

Esempio: Espressione set interna

```
Sum( {$<Year={2021}>} Sales )
```

Utilizzare un'espressione set al di fuori della funzione di aggregazione se si hanno espressioni con più aggregazioni e si vuole evitare di scrivere la stessa espressione set in ogni funzione di aggregazione.

Se si utilizza un'espressione di set esterna, questa deve essere posizionata all'inizio dell'ambito.

Esempio: Espressione set esterna

```
{<Year={2021}>} Sum(Sales) / Count(distinct Customer)
```

Se si utilizza un'espressione set al di fuori della funzione di aggregazione, è possibile applicarla anche alle misure principali esistenti.

Esempio: Espressione set esterna applicata alla misura principale

```
{<Year={2021}>} [Master Measure]
```

Un'espressione set utilizzata al di fuori delle funzioni di aggregazione influisce sull'intera espressione, a meno che non sia racchiusa tra parentesi; in tal caso le parentesi definiscono l'ambito. Nell'esempio di scoping lessicale che segue, l'espressione set viene applicata solo all'aggregazione all'interno delle parentesi.

Esempio: Scoping lessicale

```
( {<Year={2021}>} Sum(Amount) / Count(distinct Customer) ) - Avg(CustomerSales)
```

Regole

Ambito lessicale

L'espressione set influisce sull'intera espressione, a meno che non sia racchiusa tra parentesi. In tal caso, le parentesi definiscono l'ambito lessicale.

Posizione

L'espressione set deve essere collocata all'inizio dell'ambito lessicale.

Contesto

Il contesto è la selezione rilevante per l'espressione. Tradizionalmente, il contesto è sempre stato lo stato predefinito della selezione corrente. Ma se un oggetto è impostato su uno stato alternativo, il contesto è lo stato alternativo della selezione corrente.

È possibile definire un contesto anche sotto forma di espressione set esterna.

Eredità

Le espressioni set interne hanno la precedenza su quelle esterne. Se l'espressione set interna contiene un identificatore set, sostituisce il contesto. Altrimenti, il contesto e l'espressione set verranno uniti.

- `{$<SetExpression>}` - sovrascrive l'espressione set esterna
- `{<SetExpression>}` - viene unito all'espressione set esterna

Assegnazione set elementi

L'assegnazione set elementi determina il modo in cui le due selezioni vengono unite. Se viene utilizzato un normale segno di uguale, la selezione nell'espressione set interna ha la precedenza. Altrimenti, verrà utilizzato l'operatore set implicito.

- `{<Field={value}>}` - questa selezione interna sostituisce qualsiasi selezione esterna in "`Field`".
- `{<Field+={value}>}` - questa selezione interna viene unita alla selezione esterna in "`Field`", utilizzando l'operatore di unione.
- `{<Field*={value}>}` - questa selezione interna viene unita alla selezione esterna in "`Field`", utilizzando l'operatore di intersezione.

Ereditarietà in più fasi

L'ereditarietà può avvenire in più fasi. Esempi:

- Selezione corrente → `Sum(Amount)`
  La funzione di aggregazione utilizzerà il contesto, che in questo caso è la selezione corrente.
- Selezione corrente → `{<Set1>} Sum(Amount)`
  `Set1` erediterà dalla selezione corrente e il risultato sarà il contesto per la funzione di aggregazione.
- Selezione corrente → `{<Set1>} ({<Set2>} Sum(Amount))`
  `Set2` erediterà da `Set1`, che a sua volta eredita dalla selezione corrente e il risultato sarà il contesto per la funzione di aggregazione.

La funzione Aggr()

La funzione `Aggr()` crea un'aggregazione nidificata con due aggregazioni indipendenti. Nell'esempio in basso, viene calcolato un `Count()` per ciascun valore di `Dim`, e l'array risultante viene aggregato utilizzando la funzione `Sum()`.

Esempio:

`Sum(Aggr(Count(X),Dim))`
`Count()` è un'aggregazione interna mentre `Sum()` è un'aggregazione esterna.

- L'aggregazione interna non eredita alcun contesto dall'aggregazione esterna.
- L'aggregazione interna eredita il contesto dalla funzione `Aggr()`, che può contenere un'espressione set.
- Sia la funzione `Aggr()` sia la funzione di aggregazione esterna ereditano il contesto da un'espressione set esterna.

Tutorial - Creazione di un'espressione set

È possibile creare espressioni set per supportare l'analisi dati. In questo contesto, si fa spesso riferimento all'analisi come Set Analysis. Set Analysis offre un modo per definire un ambito diverso dal set di record definiti dalla selezione corrente in un'app.

Cosa si apprenderà

Questo tutorial fornisce i dati e le espressioni del grafico per costruire espressioni set usando modificatori, identificatori e operatori set.

Chi dovrebbe completare questo tutorial

Questo tutorial è destinato agli sviluppatori di app abituati a lavorare con l'editor di script e con le espressioni del grafico.

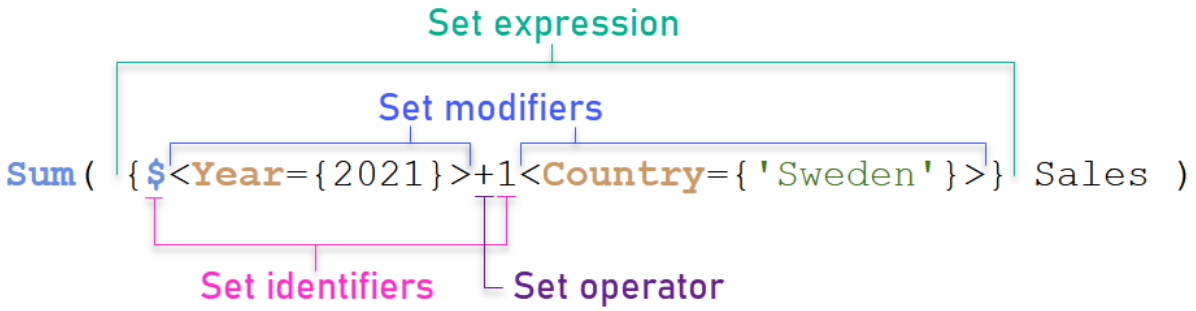
Cosa bisogna fare prima di iniziare

Un'allocazione per l'accesso a Qlik Sense Enterprise Professional, che consente di caricare dati e creare app.

Elementi in un'espressione set

Le espressioni set sono racchiuse in una funzione di aggregazione, come `Sum()`, `Max()`, `Min()`, `Avg()` o `Count()`. Le espressioni set sono formate da blocchi costitutivi noti come elementi. Tali elementi sono modificatori, identificatori e operatori set.

Elementi in un'espressione set



L'espressione set di cui sopra, ad esempio, viene costruita dall'aggregazione `Sum(Sales)`. L'espressione set è racchiusa in parentesi graffe esterne: `{ }`

Il primo operando nell'espressione è: `$<Year={2021}>`

Questo operando restituisce le vendite per l'anno `2021` per la selezione corrente. Il modificatore, `<Year={2021}>`, contiene la selezione dell'anno 2021. L'identificatore set `$` indica che l'espressione set è basata sulla selezione corrente.

Il secondo operando nell'espressione è: `1<Country={'Sweden'}>`

Questo operando restituisce Sales per Sweden. Il modificatore, `<Country={'Sweden'}>`, contiene la selezione del paese Sweden. L'identificatore set `1` indica che le selezioni effettuate nell'app verranno ignorate.

Infine, l'operatore set `+` indica che l'espressione restituisce un set composto dai record appartenenti a uno qualsiasi dei due operandi set.

Creazione di un tutorial di espressioni set

Completare le procedure seguenti per creare le espressioni set mostrate in questo tutorial.

Creare una nuova app e caricare i dati

Procedere come indicato di seguito:

1. Creare una nuova app.
2. Fare clic su **Editor di script**. In alternativa, fare clic su **Prepara** > **Editor caricamento dati** nella barra di navigazione.
3. Creare una nuova sezione nell'**Editor caricamento dati**.
4. Copiare i dati seguenti e incollarli nella nuova sezione: *Dati tutorial espressioni set (page 319)*
5. Fare clic su **Carica dati**. I dati vengono caricati come caricamento inline.

Creare espressioni set con i modificatori

Il modificatore set è composto da uno o più nomi di campo, ciascuno seguito da una selezione che dovrebbe essere effettuata sul campo. Il modificatore è racchiuso tra parentesi angolari. Ad esempio, in questa espressione set:

```
Sum ( {<Year = {2015}>} Sales )
```

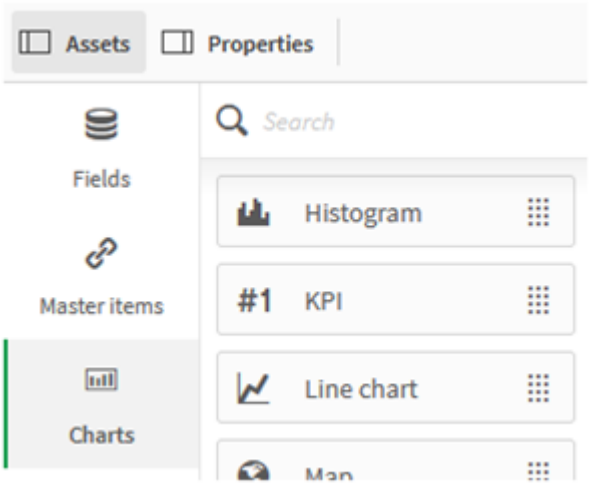
Il modificatore è:

```
<Year = {2015}>
```

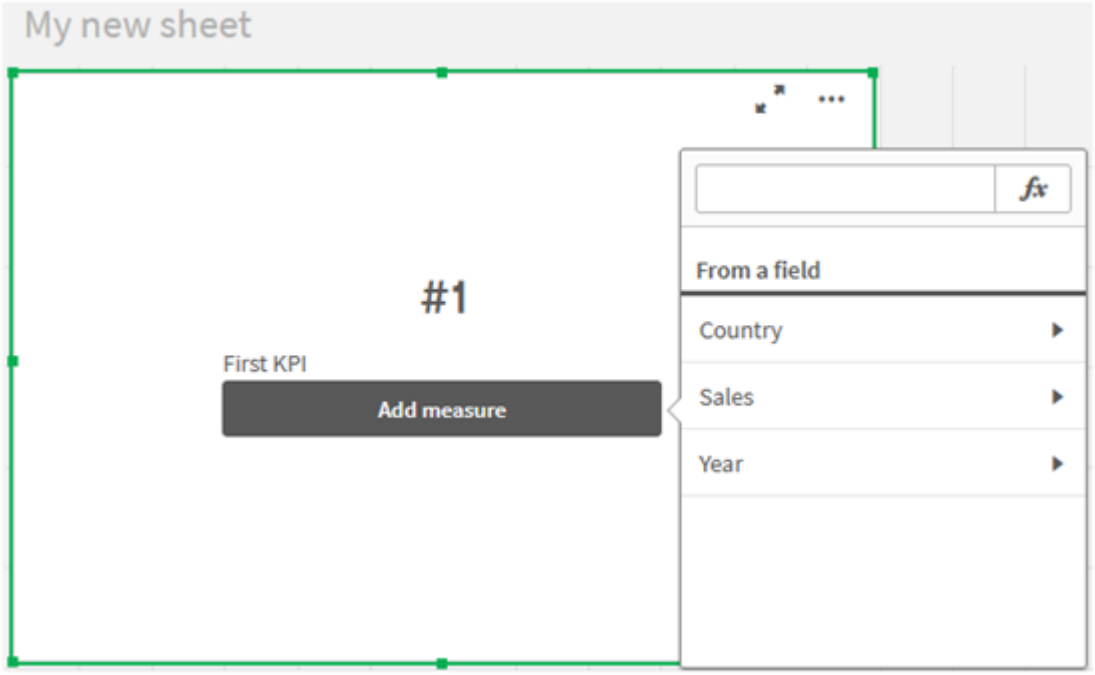
Questo modificatore specifica che verranno selezionati dati relativi all'anno 2015. Le parentesi graffe in cui il modificatore è racchiuso indicano un'espressione set.

Procedere come indicato di seguito:

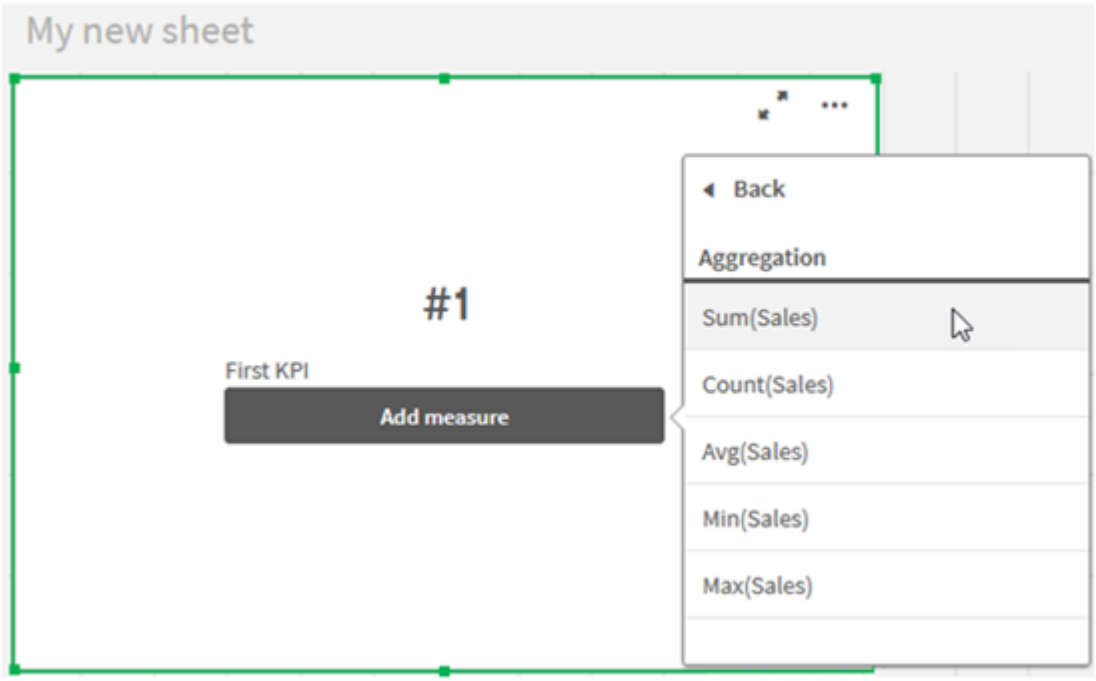
1. In un foglio, aprire il pannello **Asset** dalla barra di navigazione, quindi fare clic su **Grafici**.



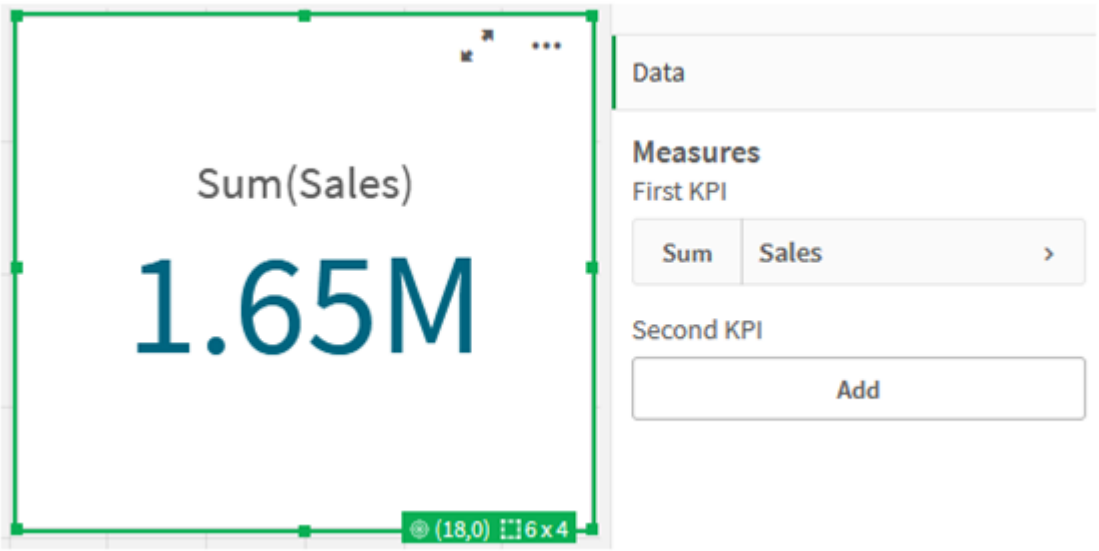
2. Trascinare un **KPI** sul foglio, quindi fare clic su **Aggiungi misura**.



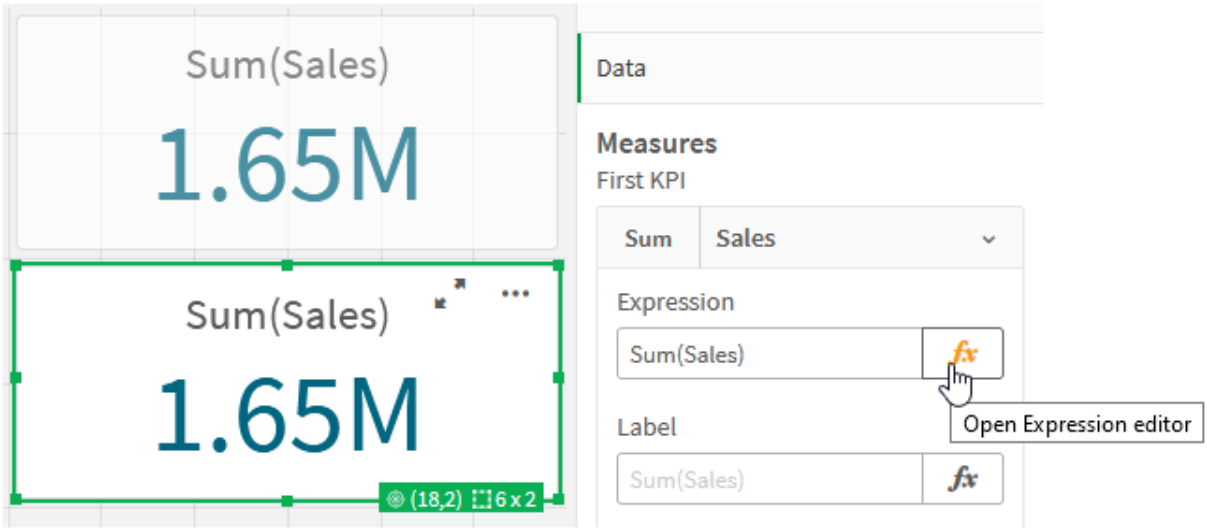
3. Fare clic su `Sales`, quindi selezionare `Sum(Sales)` per l'aggregazione.



KPI mostra la somma delle vendite per tutti gli anni.



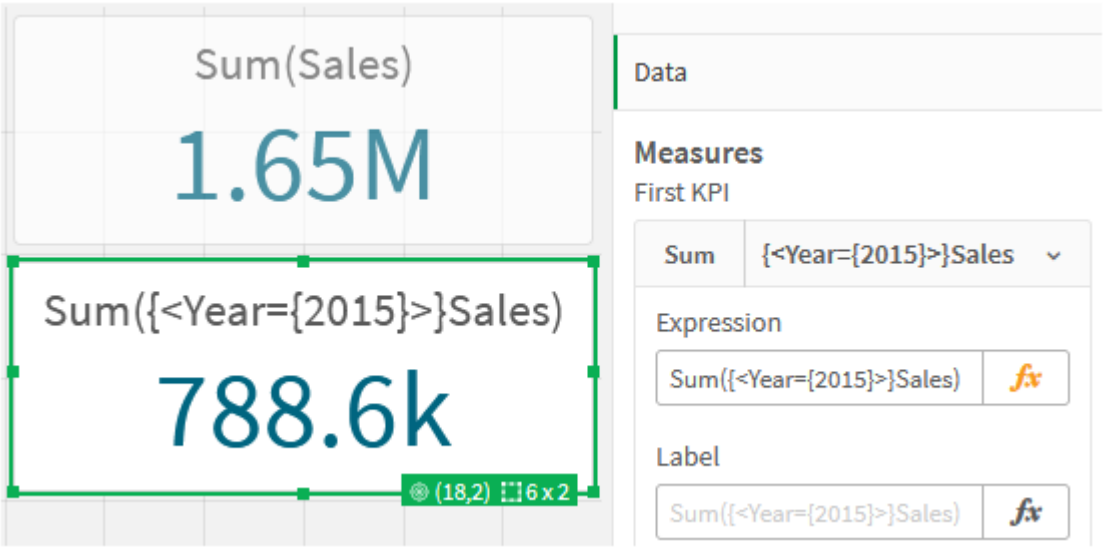
4. Copiare e incollare KPI per creare un nuovo KPI.
5. Fare clic sul nuovo KPI, fare clic su **Vendite** sotto **Misure** e quindi fare clic su **Apri editor delle espressioni**.



L'editor delle espressioni si apre con l'aggregazione `Sum(Sales)`.

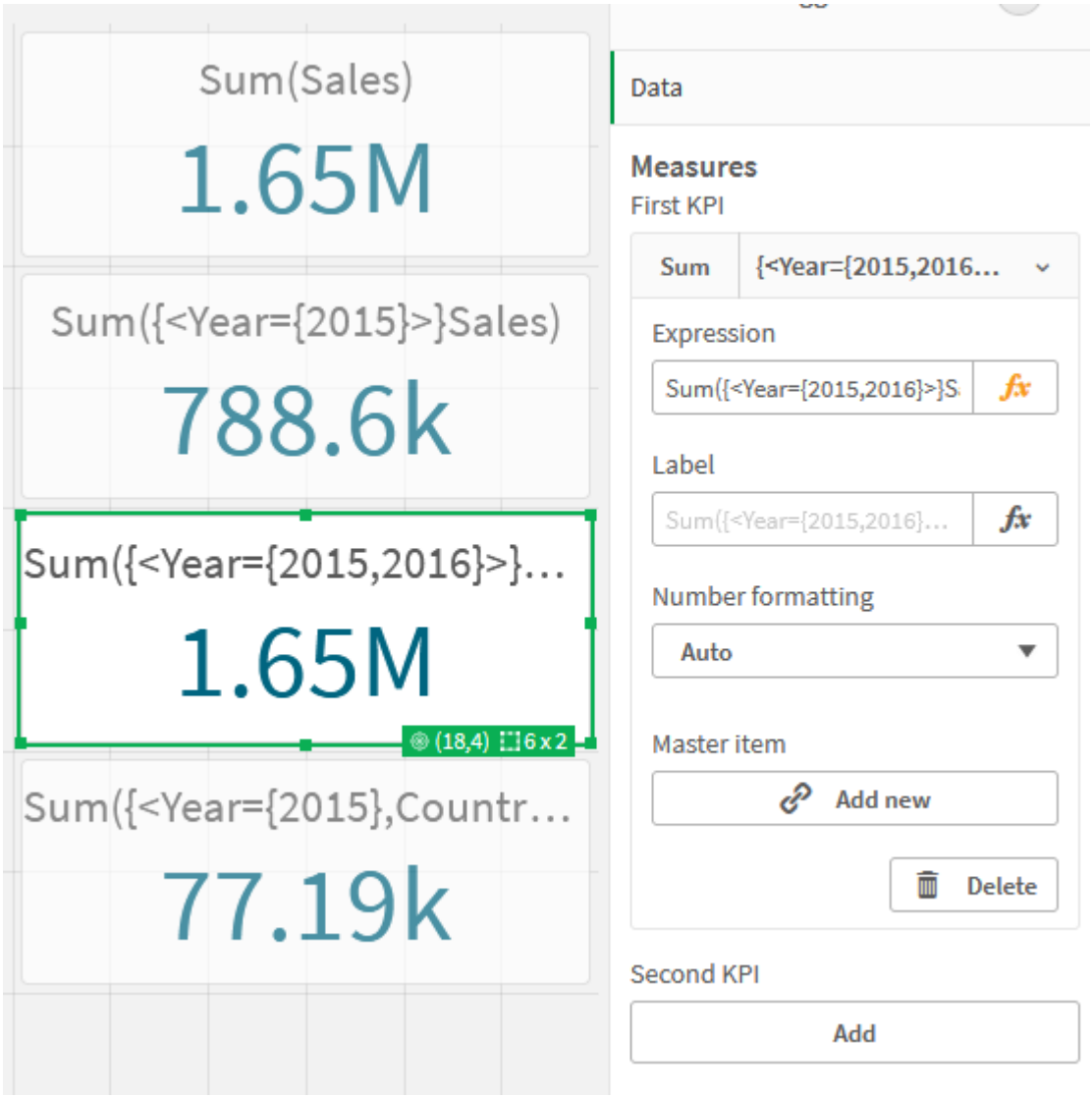
6. Nell'editor delle espressioni, creare un'espressione per sommare Sales solo per il 2015:
   i. Aggiungere parentesi graffe per indicare un'espressione set: `Sum({}Sales)`
   i. Aggiungere parentesi angolari per indicare un modificatore set: `Sum({<>}Sales)`
   ii. Nelle parentesi angolari, aggiungere il campo da selezionare, in questo caso il campo `Year`, seguito da un segno di uguale. Quindi, racchiudere 2015 in un altro set di parentesi graffe. Il modificatore set risultante è: `{<Year={2015}>}`.
   L'intera espressione è:
   `Sum({<Year={2015}>}Sales)`

iii. Fare clic su **Applica** per salvare l'espressione e chiudere l'editor delle espressioni. La somma di Sales per 2015 è mostrata in KPI.



7. Creare altri due KPI con le seguenti espressioni:
   `Sum({<Year={2015,2016}>}Sales)`
   Il modificatore di cui sopra è `<Year={2015,2016}>`. L'espressione restituirà la somma di Sales per il 2015 e il 2016.
   `Sum({<Year={2015},Country={'Germany'}>} Sales)`
   Il modificatore di cui sopra è `<Year={2015}, Country={'Germany'}>`. L'espressione restituirà la somma di Sales per il 2015, in cui il 2015 si interseca con Germany.

KPI che utilizzano i modificatori set



Aggiunta di identificatori set

Le espressioni set sopra utilizzeranno le selezioni correnti come base, dato che un identificatore non è stato utilizzato. Quindi, aggiungere identificatori per specificare il comportamento quando vengono effettuate selezioni.

Procedere come indicato di seguito:

Sul proprio foglio, creare o copiare le seguenti espressioni set:

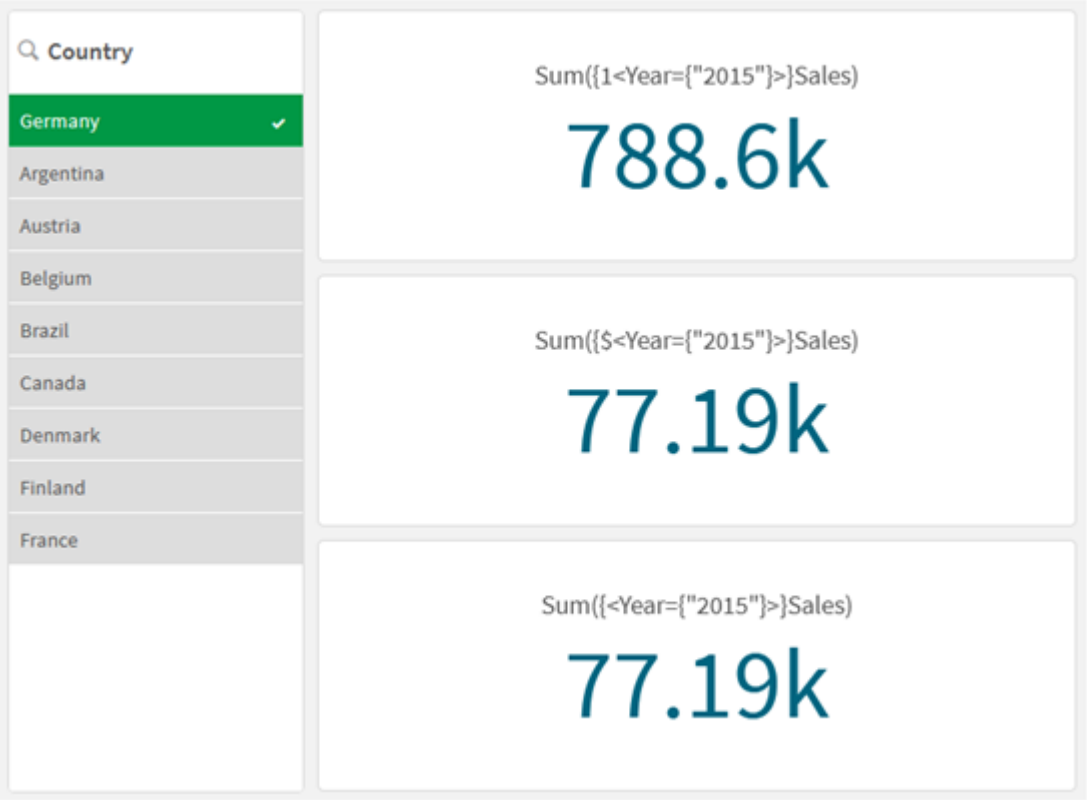
```
Sum({$<Year={"2015"}>}Sales
```

L'identificatore `$` baserà l'espressione set sulle selezioni correnti effettuate nei dati. Ciò rappresenta anche il comportamento predefinito quando non viene utilizzato un identificatore.

```
Sum({1<Year={"2015"}>}Sales)
```

L'identificatore `1` causerà l'aggregazione di `Sum(Sales)` sul 2015 per ignorare la selezione corrente. Il valore dell'aggregazione non cambierà quando l'utente effettua altre selezioni. Ad esempio, quando Germany viene selezionato in basso, il valore per la somma aggregata del 2015 non cambia.

KPI che utilizzano modificatori e identificatori set



Aggiunta di operatori

Gli operatori set sono utilizzati per includere, escludere o intersecare serie di dati. Tutti gli operatori utilizzano i gruppi come operandi e restituiscono un gruppo come risultato.

È possibile utilizzare gli operatori set in due situazioni diverse:

- Per eseguire un'operazione set sugli identificatori set, rappresentando i set di record nei dati.
- Per eseguire un'operazione sui set di elementi, sui valori di campo o all'interno di un modificatore set.

Procedere come indicato di seguito:

Sul proprio foglio, creare o copiare la seguente espressione set:

```
Sum({$<Year={2015}>+1<Country={'Germany'}>}Sales)
```

L'operatore segno positivo `(+)` produce un'unione dei set di dati per 2015 e Germany. Come spiegato con gli identificatori set sopra, l'identificatore con simbolo del dollaro `($)` indica che verranno rispettate le selezioni correnti utilizzate per il primo operando, `<Year={2015}>`. L'identificatore `1` che segnala la selezione verrà ignorato per il secondo operando, `<Country={'Germany'}>`.

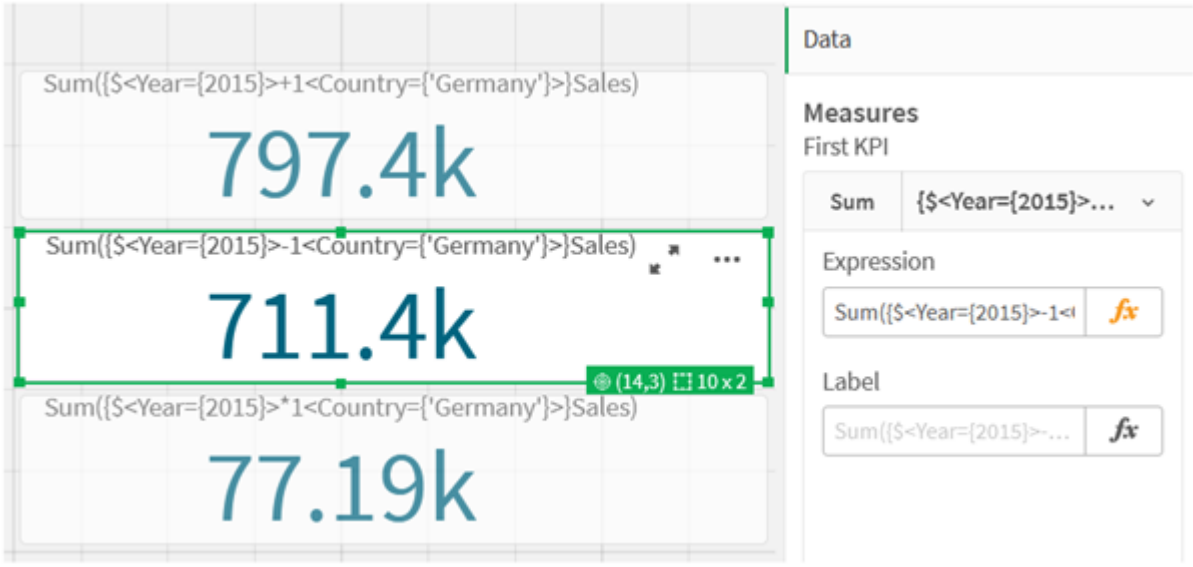
KPI che utilizza l'operatore con segno positivo (+)

In alternativa, utilizzare un segno meno `(-)` per restituire una serie di dati che consiste nei record appartenenti a 2015 ma non a Germany. Oppure, utilizzare un asterisco `(*)` per restituire un set composto dai record appartenenti a entrambi i set.

```
Sum({$<Year={2015}>-1<Country={'Germany'}>}Sales)
```

```
Sum({$<Year={2015}>*1<Country={'Germany'}>}Sales)
```

KPI che utilizzano operatori



Dati tutorial espressioni set

Script di caricamento

Caricare i dati seguenti come caricamento inline, quindi creare le espressioni del grafico nel tutorial.

```
//Create table SalesByCountry
SalesByCountry:
Load * Inline [
Country, Year, Sales
Argentina, 2016, 66295.03
Argentina, 2015, 140037.89
```

```
Austria, 2016, 54166.09
Austria, 2015, 182739.87
Belgium, 2016, 182766.87
Belgium, 2015, 178042.33
Brazil, 2016, 174492.67
Brazil, 2015, 2104.22
Canada, 2016, 101801.33
Canada, 2015, 40288.25
Denmark, 2016, 45273.25
Denmark, 2015, 106938.41
Finland, 2016, 107565.55
Finland, 2015, 30583.44
France, 2016, 115644.26
France, 2015, 30696.98
Germany, 2016, 8775.18
Germany, 2015, 77185.68
];
```

Sintassi per le espressioni set

La sintassi completa (non includendo l'utilizzo opzionale delle parentesi standard per definire la precedenza) viene descritta mediante la metasintassi Backus-Naur Form:

```
set_expression ::= { set_entity { set_operator set_entity } }
set_entity ::= set_identifier [ set_modifier ] | set_modifier
set_identifier ::= 1 | $ | $N | $_N | bookmark_id | bookmark_name
set_operator ::= + | - | * | /
set_modifier ::= < field_selection {, field_selection } >
field_selection ::= field_name [ = | += | -= | *= | /= ] element_set_
expression
element_set_expression ::= [ - ] element_set { set_operator element_set }
element_set ::= [ field_name ] | { element_list } | element_function
element_list ::= element { , element }
element_function ::= ( P | E ) ( [set_expression] [field_name] )
element ::= field_value | " search_mask "
```

3.3 Sintassi generale per le espressioni grafiche

Per le espressioni grafiche, è possibile utilizzare la seguente struttura sintattica generale, con molti parametri opzionali:

```
expression ::= ( constant | expressionname | operator1 expression | expression operator2
expression | function | aggregation function | (expression ) )
```

dove:

constant è una stringa (un testo, una data o un'ora) racchiusa tra virgolette singole diritte o un numero. Le costanti sono scritte senza separatore delle migliaia e con un punto decimale come separatore decimale.

expressionname è il nome (etichetta) di un'altra espressione dello stesso grafico.

operator1 è un operatore unario (che agisce su un'unica espressione, quella a destra).

operator2 è un operatore binario (che agisce su due espressioni, una per ogni lato).

```
function ::= functionname ( parameters )
parameters ::= expression { , expression }
```
Il numero e i tipi dei parametri non sono arbitrari. Dipendono dal tipo di funzione utilizzata.

```
aggregationfunction ::= aggregationfunctionname ( parameters2 )
parameters2 ::= aggrexpression { , aggrexpression }
```
Il numero e i tipi dei parametri non sono arbitrari. Dipendono dal tipo di funzione utilizzata.

3.4 Sintassi generale per le aggregazioni

Per le aggregazioni, è possibile utilizzare la seguente struttura sintattica generale, con molti parametri opzionali:

```
aggrexpression ::= ( fieldref | operator1 aggrexpression | aggrexpression operator2
aggrexpression | functioninaggr | ( aggrexpression ) )
```
fieldref è un nome di campo.

```
functionaggr ::= functionname ( parameters2 )
```
Le espressioni e le funzioni possono essere nidificate liberamente e, purché **fieldref** sia sempre racchiuso esattamente da una sola funzione di aggregazione e l'espressione restituisca un valore interpretabile, Qlik Sense non genererà alcun messaggio di errore.

4 Operatori

In questa sezione vengono descritti gli operatori che è possibile utilizzare in Qlik Sense. Sono disponibili due tipologie di operatori:

- Operatori unari (utilizzano un solo operando)
- Operatori binari (utilizzano due operandi)

La maggior parte degli operatori è di tipo binario.

È possibile definire le seguenti tipologie di operatori:

- Operatori bit a bit
- Operatori logici
- Operatori numerici
- Operatori relazionali
- Operatori su stringa

4.1 Operatori bit a bit

Tutti gli operatori bit a bit convertono (troncano) gli operandi in numeri interi con segno (32 bit) e restituiscono un risultato nello stesso modo. Tutte le operazioni vengono eseguite bit a bit. Se un operando non può essere interpretato come numero, l'operazione restituirà NULL.

Operatori bit a bit

Operatore	Nome completo	Descrizione
bitnot	Bit inverso.	Operatore unario. L'operazione restituisce l'inverso logico dell'operando eseguito bit a bit. **Esempio:** `bitnot 17` restituisce -18
bitand	Bit AND.	L'operazione restituisce l'AND logico degli operandi eseguito bit a bit. **Esempio:** `17 bitand 7` restituisce 1
bitor	Bit OR.	L'operazione restituisce l'OR logico degli operandi eseguito bit a bit. **Esempio:** `17 bitor 7` restituisce 23

Operatore	Nome completo	Descrizione
bitxor	Bit OR esclusivo.	L'operazione restituisce l'OR esclusivo logico degli operandi eseguito bit a bit. **Esempio:** `17 bitxor 7` restituisce 22
>>	Spostamento a destra di bit.	L'operazione restituisce il primo operando spostato a destra. Il numero di passi viene definito nel secondo operando. **Esempio:** `8 >> 2` restituisce 2
<<	Spostamento a sinistra di bit.	L'operazione restituisce il primo operando spostato a sinistra. Il numero di passi viene definito nel secondo operando. **Esempio:** `8 << 2` restituisce 32

4.2 Operatori logici

Tutti gli operatori logici interpretano gli operandi in modo logico e restituiscono True (-1) o False (0) come risultato.

Operatori logici

Operatore	Descrizione
not	Inverso logico. Uno dei pochi operatori unari. L'operazione restituisce l'inverso logico dell'operando.
and	And logico. Restituisce l'operatore logico AND degli operandi.
or	Or logico. Restituisce l'operatore logico OR degli operandi.
Xor	Or esclusivo logico. Restituisce l'esclusivo logico OR degli operandi. È l'equivalente dell'operatore logico OR, ma con la differenza che restituisce un risultato False se entrambi gli operandi hanno un valore True.

4.3 Operatori numerici

Tutti gli operatori numerici usano i valori numerici degli operandi e restituiscono un valore numerico come risultato.

Operatori numerici

Operatore	Descrizione
+	Segno per i numeri positivi (operatore unario) o addizione aritmetica. L'operatore binario restituisce la somma dei due operandi.
-	Segno per i numeri negativi (operatore unario) o sottrazione aritmetica. L'operazione unaria restituisce l'operando moltiplicato per -1, e quella binaria la differenza fra i due operandi.
*	Moltiplicazione aritmetica. L'operazione restituisce il prodotto dei due operandi.
/	Divisione aritmetica. L'operazione restituisce il rapporto fra i due operandi.

4.4 Operatori relazionali

Tutti gli operatori relazionali confrontano i valori degli operandi e restituiscono True (-1) o False (0) come risultato. Tutti gli operatori relazionali sono binari.

Operatori relazionali

Operatore	Descrizione
<	Minore di. Viene effettuato un confronto numerico se tutti e due gli operandi possono essere interpretati numericamente. L'operazione restituisce il valore logico della valutazione del confronto.
<=	Minore di o uguale a. Viene effettuato un confronto numerico se tutti e due gli operandi possono essere interpretati numericamente. L'operazione restituisce il valore logico della valutazione del confronto.
>	Maggiore di. Viene effettuato un confronto numerico se tutti e due gli operandi possono essere interpretati numericamente. L'operazione restituisce il valore logico della valutazione del confronto.
>=	Maggiore di o uguale a. Viene effettuato un confronto numerico se tutti e due gli operandi possono essere interpretati numericamente. L'operazione restituisce il valore logico della valutazione del confronto.
=	Uguale a. Viene effettuato un confronto numerico se tutti e due gli operandi possono essere interpretati numericamente. L'operazione restituisce il valore logico della valutazione del confronto.

Operatore	Descrizione
<>	Diverso da. Viene effettuato un confronto numerico se tutti e due gli operandi possono essere interpretati numericamente. L'operazione restituisce il valore logico della valutazione del confronto.
precedes	A differenza dell'operatore **<**, non vengono eseguiti tentativi di interpretazione numerica dei valori dell'argomento prima del confronto. L'operazione restituisce true se il valore a sinistra dell'operatore ha una rappresentazione testuale che, confrontando le stringhe, precede la rappresentazione testuale del valore di destra. **Esempio:** `'1 ' precedes ' 2'` restituisce FALSE `' 1' precedes ' 2'`restituisce TRUE dato che il valore ASCII di uno spazio (' ') ha un valore inferiore rispetto al valore ASCII di un numero. Mentre: `'1 ' < ' 2'` restituisce TRUE `' 1' < ' 2'`restituisce TRUE
follows	A differenza dell'operatore **>**, non vengono eseguiti tentativi di interpretazione numerica dei valori dell'argomento prima del confronto. L'operazione restituisce true se il valore a sinistra dell'operatore ha una rappresentazione testuale che, confrontando le stringhe, segue la rappresentazione testuale del valore di destra. **Esempio:** `' 2' follows '1'` restituisce FALSE ' 2' follows ' 1'restituisce TRUE dato che il valore ASCII di uno spazio (' ') ha un valore inferiore rispetto al valore ASCII di un numero. Mentre: `' 2' > ' 1'`restituisce TRUE `' 2' > '1 '`restituisce TRUE

4.5 Operatori su stringa

Esistono due operatori su stringa. Uno utilizza i valori delle stringhe degli operandi e restituisce una stringa come risultato. L'altro confronta gli operandi e restituisce un valore booleano per indicare la corrispondenza.

&

Concatenazione di stringhe. L'operazione restituisce una stringa di testo, costituita dall'unione delle due stringhe degli operandi.

Esempio:

`'abc' & 'xyz'` restituisce 'abcxyz'

like

Confronto tra stringhe con caratteri speciali. L'operazione restituisce un valore booleano True (-1) se la stringa prima dell'operatore coincide con la stringa dopo l'operatore. La seconda stringa può contenere i caratteri jolly * (qualsiasi numero di caratteri arbitrari) oppure ? (un carattere arbitrario).

Esempio:

`'abc' like 'a*'` restituisce True (-1)

`'abcd' like 'a?c*'` restituisce True (-1)

`'abc' like 'a??bc'` restituisce False (0)

5 Funzioni per script e grafici

Trasformare e aggregare i dati mediante le funzioni negli script di caricamento dati e nelle espressioni del grafico.

È possibile utilizzare diverse funzioni nello stesso modo sia negli script di caricamento dei dati che nelle espressioni grafiche, anche se vi sono alcune eccezioni:

- Alcune funzioni possono essere utilizzate esclusivamente negli script di caricamento dei dati e si caratterizzano dal fatto che sono funzioni di script.
- Alcune funzioni possono essere utilizzate esclusivamente nelle espressioni grafiche e si caratterizzano dal fatto che sono funzioni grafiche.
- Alcune funzioni possono essere utilizzate sia negli script di caricamento dei dati che nelle espressioni grafiche, ma con alcune differenze nei parametri e nell'applicazione. Tali funzioni vengono descritte in argomenti separati relativi alla funzione di script o alla funzione grafica.

5.1 Connessioni di analisi per estensioni lato server (SSE, Server-Side Extension)

Le funzioni abilitate dalle connessioni di analisi saranno visibili solo una volta configurate le connessioni di analisi e avviato Qlik Sense.

Per configurare le connessioni di analisi nella QMC, vedere l'argomento "Creazione di una connessione di analisi" nella guida Gestione di siti Qlik Sense.

Per configurare le connessioni di analisi in Qlik Sense Desktop è necessario modificare il file *Settings.ini*. Vedere l'argomento “Configurazione di connessioni di analisi in Qlik Sense Desktop” nella guida Qlik Sense Desktop.

5.2 Funzioni di aggregazione

La serie di funzioni conosciute come funzioni di aggregazione è costituita da funzioni che prendono più valori di campo come input e restituiscono un singolo risultato per gruppo, dove l’aggregazione viene definita con una dimensione del grafico o una clausola **group by** nell'istruzione dello script.

Le funzioni di aggregazione comprendono **Sum()**, **Count()**, **Min()**, **Max()** e molte altre ancora.

La maggior parte delle funzioni può essere utilizzata sia nello script di caricamento dei dati che nelle espressioni grafiche, anche se la sintassi sarà diversa.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Quando si denomina un'entità, evitare di assegnare lo stesso nome a più campi, variabili o misure. Esiste un rigoroso ordine di precedenza per risolvere i conflitti tra entità con nomi identici. Questo ordine si riflette in qualsiasi oggetto o contesto in cui queste entità vengono utilizzate. L'ordine di precedenza è il seguente:

- All'interno di un'aggregazione, un campo ha la precedenza su una variabile. Le etichette delle misure non sono rilevanti nelle aggregazioni e non hanno priorità.
- Al di fuori di un'aggregazione, un'etichetta di misura ha la precedenza su una variabile, che a sua volta ha la precedenza su un nome di campo.
- Inoltre, al di fuori di un'aggregazione, una misura può essere riutilizzata facendo riferimento alla sua etichetta, a meno che non si tratti di un'etichetta calcolata. In tale situazione, la misura diminuisce di significato per ridurre il rischio di autoreferenzialità e in questo caso il nome sarà sempre interpretato prima come etichetta di misura, poi come nome di campo e infine come nome di variabile.

Utilizzo delle funzioni di aggregazione in uno script di caricamento dei dati

Le funzioni di aggregazione possono essere utilizzate soltanto all'interno di istruzioni **LOAD** e **SELECT** .

Utilizzo delle funzioni di aggregazione nelle espressioni grafiche

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

La funzione di aggregazione aggrega il set di possibili record definiti dalla selezione. Tuttavia, è possibile definire un set alternativo di record mediante un'espressione di gruppo nell'analisi di gruppo.

Come vengono calcolate le aggregazioni

Un'aggregazione si chiude intorno ai record di una tabella specifica, aggregando i record al suo interno. Ad esempio, **Count**(<Field>) conteggerà il numero di record nella tabella in cui risiede <Field>. Se si desidera aggregare solo i valori di campo distinti, utilizzare la clausola **distinct**, come **Count**(**distinct** <Field>).

Se la funzione di aggregazione contiene campi da tabelle diverse, la funzione di aggregazione si chiude sui record del prodotto incrociato delle tabelle dei campi costituenti. Ciò penalizza le prestazioni e per questo motivo tali aggregazioni devono essere evitate, specialmente in presenza di grandi quantità di dati.

Aggregazione dei campi chiave

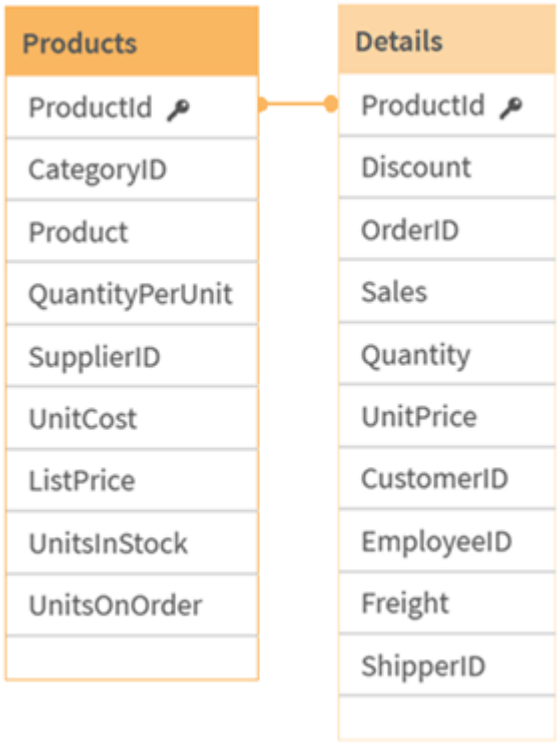
Il modo in cui vengono calcolate le aggregazioni fa sì che non sia possibile aggregare campi chiave, perché non è chiaro quale tabella debba essere utilizzata per l'aggregazione. Ad esempio, se il campo <Key> collega due tabelle, non è chiaro se **Count**(<Key>) debba restituire il numero di record dalla prima o dalla seconda tabella.

Tuttavia, se si utilizza la clausola **distinct**, l'aggregazione viene definita adeguatamente e può essere calcolata.

Pertanto, se si utilizza un campo chiave all'interno di una funzione di aggregazione senza la clausola **distinct**, Qlik Sense restituirà un numero che può essere privo di significato. La soluzione consiste nell'utilizzare la clausola **distinct** o una copia della chiave – una copia che risiede in una sola tabella.

Ad esempio, nelle tabelle seguenti, ProductID rappresenta la chiave tra le tabelle.

Chiave ProductID tra le tabelle Prodotti e Dettagli



Count(ProductID) può essere conteggiato nella tabella Products (che presenta un solo record per prodotto – ProductID è la chiave primaria) oppure può essere conteggiato nella tabella Details (che disporrà molto probabilmente di svariati record per prodotto). Se si desidera conteggiare il numero di prodotti distinti, utilizzare Count(distinct ProductID). Se si desidera conteggiare il numero di righe in una tabella specifica, non utilizzare la chiave.

Funzioni di aggregazione di base

Prospetto delle funzioni di aggregazione di base

Le funzioni di aggregazione di base sono un gruppo delle funzioni di aggregazione più comuni.

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Funzioni di aggregazione di base nello script di caricamento dei dati

FirstSortedValue

FirstSortedValue() restituisce il valore dell'espressione specificata in **value** che corrisponde al risultato della classificazione dell'argomento a **sort_weight**, ad esempio, il nome del prodotto con il prezzo unitario più basso. Il valore n nell'ordine di classificazione può essere specificato in **rank**. Se più valori risultanti condividono lo stesso **sort_weight** per il **rank** specificato, la funzione restituisce NULL. I valori classificati vengono ripetuti su un insieme di record, come definito da una clausola **group by**, o aggregati tra la serie di dati completa, qualora la clausola **group by** non sia stata definita.

```
FirstSortedValue ([ distinct ] expression, sort_weight [, rank ])
```

Max

Max() individua il valore numerico più alto dei dati aggregati nell'espressione, come definito da una clausola **group by**. Specificando un **rank** n, è possibile trovare il valore n-esimo più alto.

```
Max ( expression[, rank])
```

Min

Min() restituisce il valore numerico più basso dei dati aggregati nell'espressione, come definito da una clausola **group by**. Specificando un **rank** n, è possibile trovare il valore n-esimo più basso.

```
Min ( expression[, rank])
```

Mode

Mode() restituisce il valore più comune, il valore mode, dei dati aggregati nell'espressione, come definito da una clausola **group by**. La funzione **Mode()** può restituire valori numerici e valori di testo.

```
Mode (expression )
```

Only

Only() restituisce un valore se esiste esclusivamente un unico risultato possibile dai dati aggregati. Se i record contengono solo un valore verrà restituito tale valore, altrimenti verrà restituito NULL. Utilizzare la clausola **group by** per valutare più record. La funzione **Only()** può restituire valori numerici e valori di testo.

```
Only (expression )
```

Sum

Sum() calcola il totale dei valori aggregati nell'espressione, come definito da una clausola **group by**.

```
Sum ([distinct]expression)
```

Funzioni di aggregazione di base nelle espressioni grafiche

Le funzioni di aggregazione nei grafici possono essere utilizzate solo nei campi delle espressioni grafiche. L'espressione di argomento di una funzione di aggregazione non deve contenere un'altra funzione di aggregazione.

FirstSortedValue

FirstSortedValue() restituisce il valore dell'espressione specificata in **value** che corrisponde al risultato della classificazione dell'argomento a **sort_weight**, ad esempio, il nome del prodotto con il prezzo unitario più basso. Il valore n nell'ordine di classificazione può essere specificato in **rank**. Se più valori risultanti condividono lo stesso **sort_weight** per il **rank** specificato, la funzione restituisce NULL.

```
FirstSortedValue - funzione per grafici([{SetExpression}] [DISTINCT] [TOTAL
[<fld {,fld}>]] value, sort_weight [,rank])
```

Max

Max() trova il valore più alto dei dati aggregati. Specificando un **rank** n, è possibile trovare il valore n-esimo più alto.

```
Max - funzione per graficiMax() trova il valore più alto dei dati aggregati.
Specificando un rank n, è possibile trovare il valore n-esimo più alto. Può
essere utile consultare anche le funzioni FirstSortedValue e rangemax, che
presentano somiglianze con la funzione Max. Max([{SetExpression}] [TOTAL
[<fld {,fld}>]] expr [,rank])
numerico ArgomentiArgomentoDescrizioneexprL'espressione o il campo contenente
i dati da misurare.rankIl valore predefinito di rank è 1, che corrisponde al
valore più elevato. Specificando rank come 2 verrà restituito il secondo
valore più elevato. Se rank è 3, verrà restituito il terzo valore più elevato
e così via.SetExpressionPer impostazione predefinita, la funzione di
aggregazione aggrega la serie di possibili record definiti dalla selezione. È
possibile definire una serie di record alternativa mediante un'espressione
Set Analysis. TOTALSe la parola TOTAL viene riportata prima degli argomenti
della funzione, il calcolo verrà effettuato su tutti i valori possibili dati
dalle selezioni correnti e non solo su quelli relativi al valore dimensionale
attuale, vale a dire che verranno ignorate le dimensioni del grafico.
Utilizzando TOTAL [<fld {.fld}>], dove il qualificatore TOTAL è seguito da un
elenco di uno o più nomi di campo come sottogruppo delle variabili di
dimensione del grafico, si crea un sottogruppo di tutti i valori
possibili. DatiCustomerProductUnitSalesUnitPrice
AstridaAA416AstridaAA1015AstridaBB99BetacabBB510BetacabCC220BetacabDD-
25CanutilityAA815CanutilityCC-19Esempi e risultatiEsempiRisultatiMax
(UnitSales)10, in quanto questo è il valore più elevato in UnitSales.Il
valore di un ordine viene calcolato moltiplicando il numero di unità vendute
(UnitSales) per il prezzo unitario.Max(UnitSales*UnitPrice)150, in quanto
questo è il valore più elevato del risultato del calcolo di tutti i valori
possibili di (UnitSales)*(UnitPrice).Max(UnitSales, 2)9, che è il secondo
valore più elevato.Max(TOTAL UnitSales)10, perché il qualificatore TOTAL sta
a indicare che viene trovato il valore più elevato possibile, ignorando le
dimensioni del grafico. Per un grafico con Customer come dimensione, il
qualificatore TOTAL assicurerà il valore massimo nell'intera serie di dati,
anziché il massimo UnitSales per ciascun cliente.Selezionare Customer B.Max
({1} TOTAL UnitSales)10, indipendentemente dalla selezione effettuata, perché
l'espressione Set Analysis {1} definisce la serie di record da valutare come
ALL, quale che sia la selezione.Dati utilizzati negli esempi:ProductData:LOAD
```

```
* inline
[Customer|Product|UnitSales|UnitPriceAstrida|AA|4|16Astrida|AA|10|15Astrida|B
B|9|9Betacab|BB|5|10Betacab|CC|2|20Betacab|DD||25Canutility|AA|8|15Canutility
|CC||19] (delimiter is '|'); FirstSortedValue RangeMax ([{SetExpression}]
[DISTINCT] [TOTAL [<fld {,fld}>]] expr [,rank])
```

Min

Min() trova il valore più basso dei dati aggregati. Specificando un **rank** n, è possibile trovare il valore n-esimo più basso.

```
Min - funzione per grafici([{SetExpression}] [DISTINCT] [TOTAL [<fld
{,fld}>]] expr [,rank])
```

Mode

Mode() trova il valore più comune, il valore della modalità, nei dati aggregati. La funzione **Mode()** può elaborare valori di testo e valori numerici.

```
Mode - funzione per grafici ({[SetExpression] [TOTAL [<fld {,fld}>]]} expr)
```

Only

Only() restituisce un valore se esiste esclusivamente un unico risultato possibile dai dati aggregati. Ad esempio, la ricerca dell'unico prodotto con prezzo unitario = 9 restituirà NULL se più di un prodotto ha un prezzo unitario di 9.

```
Only - funzione per grafici([{SetExpression}] [DISTINCT] [TOTAL [<fld
{,fld}>]] expr)
```

Sum

Sum() calcola il totale dei valori dati dall'espressione o dal campo nei dati aggregati.

```
Sum - funzione per grafici([{SetExpression}] [DISTINCT] [TOTAL [<fld
{,fld}>]] expr])
```

FirstSortedValue

FirstSortedValue() restituisce il valore dell'espressione specificata in **value** che corrisponde al risultato della classificazione dell'argomento a **sort_weight**, ad esempio, il nome del prodotto con il prezzo unitario più basso. Il valore n nell'ordine di classificazione può essere specificato in **rank**. Se più valori risultanti condividono lo stesso **sort_weight** per il **rank** specificato, la funzione restituisce NULL. I valori classificati vengono ripetuti su un insieme di record, come definito da una clausola **group by**, o aggregati tra la serie di dati completa, qualora la clausola **group by** non sia stata definita.

Sintassi:

```
FirstSortedValue ([ distinct ] value, sort-weight [, rank ])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
value Expression	La funzione trova il valore dell'espressione **value** che corrisponde al risultato della classificazione di **sort_weight**.
sort-weight Expression	L'espressione contenente i dati da ordinare. Viene trovato il primo valore (il più basso) di **sort_weight** dal quale viene determinato il valore corrispondente dell'espressione **value**. Inserendo un segno meno davanti a **sort_weight**, la funzione restituisce invece l'ultimo valore ordinato (il più elevato).
rank Expression	Dichiarando un valore "n" di **rank** maggiore di 1, si otterrà il valore n-esimo nell'ordine.
distinct	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempi di script

Esempio	Risultato

`Temp:`
`LOAD * inline [`
`Customer|Product|OrderNumber|UnitSales|CustomerID`
`Astrida|AA|1|10|1`
`Astrida|AA|7|18|1`
`Astrida|BB|4|9|1`
`Astrida|CC|6|2|1`
`Betacab|AA|5|4|2`
`Betacab|BB|2|5|2`
`Betacab|DD|12|25|2`
`Canutility|AA|3|8|3`
`Canutility|CC|13|19|3`
`Divadip|AA|9|16|4`
`Divadip|AA|10|16|4`
`Divadip|DD|11|10|4`
`] (delimiter is '|');`
 	
`FirstSortedValue:`
`LOAD Customer,FirstSortedValue(Product, UnitSales)`
`as MyProductWithSmallestOrderByCustomer Resident`
`Temp Group By Customer;`
	`Customer`
	`MyProductWithSmallestOrderByCustomer`
	`Astrida CC`
	`Betacab AA`
	`Canutility AA`
	`Divadip DD`
	La funzione classifica UnitSales in ordine crescente, accedendo al Customer con il valore di UnitSales minimo, l'ordine più piccolo.
	
	Perché CC corrisponde all'ordine più piccolo (valore di UnitSales=2) per cliente Astrida. AA corrisponde all'ordine più piccolo (4) valore per il cliente Betacab, AA corrisponde all'ordine più piccolo (8) per il cliente Canutility e DD corrisponde all'ordine più piccolo (10) per il cliente Divadip..

Presupponendo che la tabella **Temp** venga caricata come nell'esempio precedente:
 	
`LOAD Customer,FirstSortedValue(Product, -UnitSales)`
`as MyProductWithLargestOrderByCustomer Resident`
`Temp Group By Customer;`
	`Customer`
	`MyProductWithLargestOrderByCustomer`
	`Astrida AA`
	`Betacab DD`
	`Canutility CC`
	`Divadip -`
	il segno meno precede l'argomento sort_ weight, in modo tale che la funzione classifichi prima i più elevati.
	
	Perché AA corrisponde all'ordine più elevato (valore di UnitSales:18) per il cliente Astrida, DD corrisponde all'ordine più elevato (12) per il cliente Betacab e CC corrisponde all'ordine più elevato (13) per il cliente Canutility. Sono presenti due valori identici per l'ordine più elevato (16) per il cliente Divadip, pertanto viene prodotto un risultato null.

Esempio	Risultato

Presupponendo che la tabella **Temp** venga caricata come nell'esempio precedente:
`LOAD Customer,FirstSortedValue(distinct Product, -UnitSales) as MyProductWithSmallestOrderByCustomer Resident Temp Group By Customer;`
	`Customer MyProductWithLargestOrderByCustomer`
	`Astrida AA`
	`Betacab DD`
	`Canutility CC`
	`Divadip AA`
	si verifica la stessa situazione dell'esempio precedente, tranne per il fatto che viene utilizzato il qualificatore distinct. Ciò comporta un risultato doppio per Divadip che deve non deve essere preso in considerazione, consentendo la presentazione di un valore non null.

FirstSortedValue - funzione per grafici

FirstSortedValue() restituisce il valore dell'espressione specificata in **value** che corrisponde al risultato della classificazione dell'argomento a **sort_weight**, ad esempio, il nome del prodotto con il prezzo unitario più basso. Il valore n nell'ordine di classificazione può essere specificato in **rank**. Se più valori risultanti condividono lo stesso **sort_weight** per il **rank** specificato, la funzione restituisce NULL.

Sintassi:

```
FirstSortedValue([{SetExpression}] [DISTINCT] [TOTAL [<fld {,fld}>]] value, sort_weight [,rank])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
value	Campo di output. La funzione trova il valore dell'espressione **value** che corrisponde al risultato della classificazione di **sort_weight**.
sort_weight	Campo di input. L'espressione contenente i dati da ordinare. Viene trovato il primo valore (il più basso) di **sort_weight** dal quale viene determinato il valore corrispondente dell'espressione **value**. Inserendo un segno meno davanti a **sort_weight**, la funzione restituisce invece l'ultimo valore ordinato (il più elevato).
rank	Dichiarando un valore "n" di **rank** maggiore di 1, si otterrà il valore n-esimo nell'ordine.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.

Argomento	Descrizione
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {,fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Dati

Customer	Product	UnitSales	UnitPrice
Astrida	AA	4	16
Astrida	AA	10	15
Astrida	BB	9	9
Betacab	BB	5	10
Betacab	CC	2	20
Betacab	DD	-	25
Canutility	AA	8	15
Canutility	CC	-	19

Esempi e risultati

Esempio	Risultato
`firstsortedvalue (Product, UnitPrice)`	BB, che corrisponde al `Product` con l'`UnitPrice` più basso (9).
`firstsortedvalue (Product, UnitPrice, 2)`	BB, che corrisponde al `Product` con il secondo `UnitPrice` più basso (10).
`firstsortedvalue (Customer, -UnitPrice, 2)`	Betacab, che corrisponde al `Customer` con il `Product` con il secondo `UnitPrice` più elevato (20).

Esempio	Risultato
`firstsortedvalue (Customer, UnitPrice, 3)`	NULL, perché vi sono due valori di `Customer` (Astrida e Canutility) con lo stesso valore rank (terzo più basso) `UnitPrice` (15). Usare il qualificatore distinct per accertarsi che non insorgano risultati null.
`firstsortedvalue (Customer, -UnitPrice*UnitSales, 2)`	Canutility, che corrisponde all'elemento `Customer` con il secondo valore di ordinamento delle vendite più elevato `UnitPrice` moltiplicato per `UnitSales` (120).

Dati utilizzati negli esempi:

```
ProductData:
LOAD * inline [
Customer|Product|UnitSales|UnitPrice
Astrida|AA|4|16
Astrida|AA|10|15
Astrida|BB|9|9
Betacab|BB|5|10
Betacab|CC|2|20
Betacab|DD||25
Canutility|AA|8|15
Canutility|CC||19
] (delimiter is '|');
```

Max

Max() individua il valore numerico più alto dei dati aggregati nell'espressione, come definito da una clausola **group by**. Specificando un **rank** n, è possibile trovare il valore n-esimo più alto.

Sintassi:

```
Max ( expr [, rank])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr Expression	L'espressione o il campo contenente i dati da misurare.
rank Expression	Il valore predefinito di **rank** è 1, che corrisponde al valore più elevato. Specificando **rank** come 2 verrà restituito il secondo valore più elevato. Se **rank** è 3, verrà restituito il terzo valore più elevato e così via.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempio:

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|CustomerID
Astrida|AA|1|10|1
Astrida|AA|7|18|1
Astrida|BB|4|9|1
Astrida|CC|6|2|1
Betacab|AA|5|4|2
Betacab|BB|2|5|2
Betacab|DD
Canutility|DD|3|8
Canutility|CC
] (delimiter is '|');

Max:
LOAD Customer, Max(UnitSales) as MyMax Resident Temp Group By Customer;
```

Tabella risultante

Customer	MyMax
Astrida	18
Betacab	5
Canutility	8

Esempio:

Presupponendo che la tabella **Temp** venga caricata come nell'esempio precedente:

```
LOAD Customer, Max(UnitSales,2) as MyMaxRank2 Resident Temp Group By Customer;
```

Tabella risultante

Customer	MyMaxRank2
Astrida	10
Betacab	4
Canutility	-

Max - funzione per grafici

Max() trova il valore più alto dei dati aggregati. Specificando un **rank** n, è possibile trovare il valore n-esimo

più alto.

*Può essere utile consultare anche le funzioni **FirstSortedValue** e **rangemax**, che presentano somiglianze con la funzione **Max**.*

Sintassi:

```
Max([{SetExpression}] [TOTAL [<fld {,fld}>]] expr [,rank])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
rank	Il valore predefinito di **rank** è 1, che corrisponde al valore più elevato. Specificando **rank** come 2 verrà restituito il secondo valore più elevato. Se **rank** è 3, verrà restituito il terzo valore più elevato e così via.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Dati

Customer	Product	UnitSales	UnitPrice
Astrida	AA	4	16
Astrida	AA	10	15
Astrida	BB	9	9
Betacab	BB	5	10
Betacab	CC	2	20

Customer	Product	UnitSales	UnitPrice
Betacab	DD	-	25
Canutility	AA	8	15
Canutility	CC	-	19

Esempi e risultati

Esempi	Risultati
`Max(UnitSales)`	10, in quanto questo è il valore più elevato in `UnitSales`.
Il valore di un ordine viene calcolato moltiplicando il numero di unità vendute (`UnitSales`) per il prezzo unitario. `Max (UnitSales*UnitPrice)`	150, in quanto questo è il valore più elevato del risultato del calcolo di tutti i valori possibili di (`UnitSales`)*(`UnitPrice`).
`Max(UnitSales, 2)`	9, che è il secondo valore più elevato.
`Max(TOTAL UnitSales)`	10, perché il qualificatore TOTAL sta a indicare che viene trovato il valore più elevato possibile, ignorando le dimensioni del grafico. Per un grafico con Customer come dimensione, il qualificatore TOTAL assicurerà il valore massimo nell'intera serie di dati, anziché il massimo UnitSales per ciascun cliente.
Selezionare `Customer B`. `Max({1} TOTAL UnitSales)`	10, indipendentemente dalla selezione effettuata, perché l'espressione Set Analysis {1} definisce la serie di record da valutare come ALL, quale che sia la selezione.

Dati utilizzati negli esempi:

```
ProductData:
LOAD * inline [
Customer|Product|UnitSales|UnitPrice
Astrida|AA|4|16
Astrida|AA|10|15
Astrida|BB|9|9
Betacab|BB|5|10
Betacab|CC|2|20
Betacab|DD||25
Canutility|AA|8|15
Canutility|CC||19
] (delimiter is '|');
```

Vedere anche:

p *FirstSortedValue - funzione per grafici (page 335)*

p *RangeMax (page 1334)*

Min

Min() restituisce il valore numerico più basso dei dati aggregati nell'espressione, come definito da una clausola **group by**. Specificando un **rank** n, è possibile trovare il valore n-esimo più basso.

Sintassi:

```
Min ( expr [, rank])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr Expression	L'espressione o il campo contenente i dati da misurare.
rank Expression	Il valore predefinito di **rank** è 1, che corrisponde al valore più basso. Specificando 2 come **rank**, verrà restituito il secondo valore più basso. Se **rank** è 3, verrà restituito il terzo valore più basso e così via.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempio:

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|CustomerID
Astrida|AA|1|10|1
Astrida|AA|7|18|1
Astrida|BB|4|9|1
Astrida|CC|6|2|1
Betacab|AA|5|4|2
Betacab|BB|2|5|2
Betacab|DD
Canutility|DD|3|8
Canutility|CC
] (delimiter is '|');
Min:
LOAD Customer, Min(UnitSales) as MyMin Resident Temp Group By Customer;
```

Tabella risultante

Customer	MyMin
Astrida	2
Betacab	4
Canutility	8

Esempio:

Presupponendo che la tabella **Temp** venga caricata come nell'esempio precedente:

```
LOAD Customer, Min(UnitSales,2) as MyMinRank2 Resident Temp Group By Customer;
```

Tabella risultante

Customer	MyMinRank2
Astrida	9
Betacab	5
Canutility	-

Min - funzione per grafici

Min() trova il valore più basso dei dati aggregati. Specificando un **rank** n, è possibile trovare il valore n-esimo più basso.

*Può essere utile consultare anche le funzioni **FirstSortedValue** e **rangemin**, che presentano somiglianze con la funzione **Min**.*

Sintassi:

```
Min({[SetExpression] [TOTAL [<fld {,fld}>]]} expr [,rank])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
rank	Il valore predefinito di **rank** è 1, che corrisponde al valore più basso. Specificando 2 come **rank**, verrà restituito il secondo valore più basso. Se **rank** è 3, verrà restituito il terzo valore più basso e così via.

Argomento	Descrizione
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Dati

Customer	Product	UnitSales	UnitPrice
Astrida	AA	4	16
Astrida	AA	10	15
Astrida	BB	9	9
Betacab	BB	5	10
Betacab	CC	2	20
Betacab	DD	-	25
Canutility	AA	8	15
Canutility	CC	-	19

La funzione Min() deve restituire un valore non NULL (se ne esiste uno) dalla matrice di valori restituita dall'espressione. Pertanto negli esempi, poiché sono presenti valori NULL tra i dati, la funzione restituisce il primo valore non NULL valutato dall'espressione.

Esempi e risultati

Esempi	Risultati
`Min(UnitSales)`	2, in quanto è il valore non NULL più basso in `UnitSales`.

Esempi	Risultati
Il valore di un ordine viene calcolato moltiplicando il numero di unità vendute (`UnitSales`) per il prezzo unitario. `Min (UnitSales*UnitPrice)`	40, in quanto è il risultato con il valore non NULL più basso ottenuto calcolando tutti i valori possibili di (`UnitSales`)*(`UnitPrice`).
`Min(UnitSales, 2)`	4, che è il secondo valore più basso (dopo i valori NULL).
`Min(TOTAL UnitSales)`	2, perché il qualificatore TOTAL sta a indicare che viene trovato il valore più basso possibile, ignorando le dimensioni del grafico. Per un grafico con Customer come dimensione, il qualificatore TOTAL assicurerà che venga restituito il valore di UnitSales minimo nell'intero set di dati, anziché il valore minimo per ciascun cliente.
Selezionare Customer B. `Min({1} TOTAL UnitSales)`	2, che non dipende dalla selezione di Customer B. L'espressione Set Analysis {1} definisce l'insieme di record da valutare come ALL, indipendentemente dalla selezione.

Dati utilizzati negli esempi:

```
ProductData:
LOAD * inline [
Customer|Product|UnitSales|UnitPrice
Astrida|AA|4|16
Astrida|AA|10|15
Astrida|BB|9|9
Betacab|BB|5|10
Betacab|CC|2|20
Betacab|DD||25
Canutility|AA|8|15
Canutility|CC||19
] (delimiter is '|');
```

Vedere anche:

p *FirstSortedValue - funzione per grafici (page 335)*
p *RangeMin (page 1338)*

Mode

Mode() restituisce il valore più comune, il valore mode, dei dati aggregati nell'espressione, come definito da una clausola **group by**. La funzione **Mode()** può restituire valori numerici e valori di testo.

Sintassi:

```
Mode ( expr)
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
expr Expression	L'espressione o il campo contenente i dati da misurare.

Limiti:

Se più di un valore comune ricorre lo stesso numero di volte, verrà restituito NULL.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempi di script

Esempio	Risultato

`Temp:`
`LOAD * inline [`
`Customer|Product|OrderNumber|UnitSales|CustomerID`
`Astrida|AA|1|10|1`
`Astrida|AA|7|18|1`
`Astrida|BB|4|9|1`
`Astrida|CC|6|2|1`
`Betacab|AA|5|4|2`
`Betacab|BB|2|5|2`
`Betacab|DD`
`Canutility|DD|3|8`
`Canutility|CC`
`] (delimiter is '|');`
 	
`Mode:`
`LOAD Customer, Mode(Product) as MyMostOftenSoldProduct`
`Resident Temp Group By Customer;`
	MyMostOftenSoldProduct
	
	AA
	
	perché AA è l'unico prodotto venduto più volte.

Mode - funzione per grafici

Mode() trova il valore più comune, il valore della modalità, nei dati aggregati. La funzione **Mode()** può elaborare valori di testo e valori numerici.

Sintassi:

```
Mode({[SetExpression] [TOTAL [<fld {,fld}>]]} expr)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Dati

Customer	Product	UnitSales	UnitPrice
Astrida	AA	4	16
Astrida	AA	10	15
Astrida	BB	9	9
Betacab	BB	5	10
Betacab	CC	2	20
Betacab	DD	-	25
Canutility	AA	8	15
Canutility	CC	-	19

Esempi e risultati

Esempi	Risultati
`Mode(UnitPrice)` Selezionare `Customer A`.	15, in quanto è il valore che ricorre con più frequenza in `UnitSales`. Restituisce NULL (-). Nessun singolo valore ricorre più frequentemente di un altro.
`Mode(Product)` Eseguire la selezione `Customer A`.	AA, in quanto è il valore che ricorre con più frequenza in `Product`. Restituisce NULL (-). Nessun singolo valore ricorre più frequentemente di un altro.
`Mode (TOTAL UnitPrice)`	15, perché il qualificatore TOTAL sta a indicare che il valore più comunemente ricorrente è 15, anche se le dimensioni del grafico vengono ignorate.
Selezionare `Customer B`. `Mode({1} TOTAL UnitPrice)`	15, indipendentemente dalla selezione effettuata, perché l'espressione Set Analysis {1} definisce la serie di record da valutare come ALL, quale che sia la selezione.

Dati utilizzati negli esempi:

```
ProductData:
LOAD * inline [
Customer|Product|UnitSales|UnitPrice
Astrida|AA|4|16
Astrida|AA|10|15
Astrida|BB|9|9
Betacab|BB|5|10
Betacab|CC|2|20
Betacab|DD||25
Canutility|AA|8|15
Canutility|CC||19
] (delimiter is '|');
```

Vedere anche:

p *Avg - funzione per grafici (page 400)*
p *Median - funzione per grafici (page 439)*

Only

Only() restituisce un valore se esiste esclusivamente un unico risultato possibile dai dati aggregati. Se i record contengono solo un valore verrà restituito tale valore, altrimenti verrà restituito NULL. Utilizzare la clausola **group by** per valutare più record. La funzione **Only()** può restituire valori numerici e valori di testo.

Sintassi:

```
Only ( expr )
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
expr Expression	L'espressione o il campo contenente i dati da misurare.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|CustomerID
Astrida|AA|1|10|1
Astrida|AA|7|18|1
Astrida|BB|4|9|1
Astrida|CC|6|2|1
Betacab|AA|5|4|2
Betacab|BB|2|5|2
Betacab|DD
Canutility|DD|3|8
Canutility|CC
] (delimiter is '|');
Only:
LOAD Customer, Only(CustomerID) as MyUniqIDCheck Resident Temp Group By Customer;
```

Tabella risultante

Customer	MyUniqIDCheck
Astrida	1 perché solo il cliente Astrida presenta record completi che includono CustomerID.

Only - funzione per grafici

Only() restituisce un valore se esiste esclusivamente un unico risultato possibile dai dati aggregati. Ad esempio, la ricerca dell'unico prodotto con prezzo unitario = 9 restituirà NULL se più di un prodotto ha un prezzo unitario di 9.

Sintassi:

```
Only([{SetExpression}] [TOTAL [<fld {,fld}>]] expr)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Utilizzare Only() quando si desidera un risultato NULL se nei dati campione sono presenti più valori possibili.

Esempi e risultati:

Dati

Customer	Product	UnitSales	UnitPrice
Astrida	AA	4	16
Astrida	AA	10	15
Astrida	BB	9	9
Betacab	BB	5	10
Betacab	CC	2	20
Betacab	DD	-	25
Canutility	AA	8	15
Canutility	CC	-	19

Esempi e risultati

Esempi	Risultati
`Only({<UnitPrice={9}>} Product)`	BB, perché è l'unico `Product` che ha un `UnitPrice` di '9'.
`Only({<Product={DD}>} Customer)`	Betacab, perché si tratta del solo `Customer` a vendere un `Product` denominato 'DD'.
`Only({<UnitPrice={20}>} UnitSales)`	Il numero di `UnitSales` in cui `UnitPrice` è 20 è 2, perché esiste un solo valore di `UnitSales` dove `UnitPrice` =20.
`Only({<UnitPrice={15}>} UnitSales)`	NULL, perché sono presenti due valori di `UnitSales` dove `UnitPrice` =15.

Dati utilizzati negli esempi:

```
ProductData:
LOAD * inline [
Customer|Product|UnitSales|UnitPrice
Astrida|AA|4|16
Astrida|AA|10|15
Astrida|BB|9|9
Betacab|BB|5|10
Betacab|CC|2|20
Betacab|DD||25
Canutility|AA|8|15
Canutility|CC||19
] (delimiter is '|');
```

Sum

Sum() calcola il totale dei valori aggregati nell'espressione, come definito da una clausola **group by**.

Sintassi:

```
sum ( [ distinct] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.
expr Expression	L'espressione o il campo contenente i dati da misurare.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|CustomerID
Astrida|AA|1|10|1
Astrida|AA|7|18|1
Astrida|BB|4|9|1
Astrida|CC|6|2|1
Betacab|AA|5|4|2
Betacab|BB|2|5|2
Betacab|DD
Canutility|DD|3|8
Canutility|CC
] (delimiter is '|');
Sum:
LOAD Customer, Sum(UnitSales) as MySum Resident Temp Group By Customer;
```

Tabella risultante

Customer	MySum
Astrida	39
Betacab	9
Canutility	8

Sum - funzione per grafici

Sum() calcola il totale dei valori dati dall'espressione o dal campo nei dati aggregati.

Sintassi:

```
Sum([{SetExpression}] [DISTINCT] [TOTAL [<fld {,fld}>]] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.

Argomento	Descrizione
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.  *Sebbene il qualificatore DISTINCT sia supportato, si consiglia di utilizzarlo con estrema cautela perché può fuorviare l'utente inducendolo a credere che è visualizzato un valore totale quando invece alcuni dati sono stati omessi.*
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Dati

Customer	Product	UnitSales	UnitPrice
Astrida	AA	4	16
Astrida	AA	10	15
Astrida	BB	9	9
Betacab	BB	5	10
Betacab	CC	2	20
Betacab	DD	-	25
Canutility	AA	8	15
Canutility	CC	-	19

Esempi e risultati

Esempi	Risultati
`Sum(UnitSales)`	38. Il totale dei valori in `UnitSales`.

Esempi	Risultati
`Sum(UnitSales*UnitPrice)`	505. Il totale di `UnitPrice` moltiplicato per i risultati aggregati di `UnitSales`.
`Sum (TOTAL UnitSales*UnitPrice)`	505 per tutte le righe della tabella e il totale perché il qualificatore TOTAL significa che la somma è ancora 505, indipendentemente dalle dimensioni del grafico.
Selezionare `Customer B`. `Sum({1} TOTAL UnitSales*UnitPrice)`	505, indipendentemente dalla selezione effettuata, perché l'espressione Set Analysis {1} definisce la serie di record da valutare come ALL, quale che sia la selezione.

Dati utilizzati negli esempi:

```
ProductData:
LOAD * inline [
Customer|Product|UnitSales|UnitPrice
Astrida|AA|4|16
Astrida|AA|10|15
Astrida|BB|9|9
Betacab|BB|5|10
Betacab|CC|2|20
Betacab|DD||25
Canutility|AA|8|15
Canutility|CC||19
] (delimiter is '|');
```

Funzioni di aggregazione contatore

Le funzioni di aggregazione contatore restituiscono vari tipi di conteggi di un'espressione su un insieme di record in uno script di caricamento dei dati oppure un numero di valori in una dimensione del grafico.

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Funzioni di aggregazione contatore nello script di caricamento dei dati

Count

Count() restituisce il numero dei valori aggregati nell'espressione, come definito da una clausola **group by**.

```
Count ([distinct ] expression | * )
```

MissingCount

MissingCount() restituisce il numero dei valori mancanti aggregati nell'espressione, come definito da una clausola **group by**.

```
MissingCount ([ distinct ] expression)
```

NullCount

NullCount() restituisce il numero dei valori NULL aggregati nell'espressione, come definito da una clausola **group by**.

```
NullCount ([ distinct ] expression)
```

NumericCount

NumericCount() restituisce il numero dei valori numerici presenti nell'espressione, come definito da una clausola **group by**.

```
NumericCount ([ distinct ] expression)
```

TextCount

TextCount() restituisce il numero dei valori di campo non numerici aggregati nell'espressione, come definito da una clausola **group by**.

```
TextCount ([ distinct ] expression)
```

Funzioni di aggregazione contatore nelle espressioni grafiche

Le seguenti funzioni di aggregazione contatore possono essere utilizzate nei grafici:

Count

Count() viene utilizzato per aggregare il numero di valori, di testo e numerici, in ciascuna dimensione del grafico.

```
Count - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld
{,fld}>]]} expr)
```

MissingCount

MissingCount() viene utilizzata per aggregare il numero di valori mancanti in ciascuna dimensione del grafico. I valori mancanti sono tutti valori non numerici.

```
MissingCount - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld
{,fld}>]] expr)
```

NullCount

NullCount() viene utilizzata per aggregare il numero di valori NULL in ciascuna dimensione del grafico.

```
NullCount - funzione per grafici({[SetExpression][DISTINCT] [TOTAL [<fld
{,fld}>]]} expr)
```

NumericCount

NumericCount() aggrega il numero di valori numerici in ciascuna dimensione del grafico.

```
NumericCount - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld
{,fld}>]]} expr)
```

TextCount

TextCount() viene utilizzata per aggregare il numero di valori di campo che non sono numerici in ciascuna dimensione del grafico.

```
TextCount - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld
{,fld}>]]} expr)
```

Count

Count() restituisce il numero dei valori aggregati nell'espressione, come definito da una clausola **group by**.

Sintassi:

```
Count( [distinct ] expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempi di script

Esempio	Risultato
Temp: LOAD * inline [Customer|Product|OrderNumber|UnitSales|UnitPrice Astrida|AA|1|4|16 Astrida|AA|7|10|15 Astrida|BB|4|9|9 Betacab|CC|6|5|10 Betacab|AA|5|2|20 Betacab|BB|1|25| 25 Canutility|AA|3|8|15 Canutility|CC|||19 Divadip|CC|2|4|16 Divadip|DD|3|1|25] (delimiter is '|'); Count1: LOAD Customer,Count(OrderNumber) as OrdersByCustomer Resident Temp Group By Customer;	Customer OrdersByCustomer Astrida 3 Betacab 3 Canutility 2 Divadip 2 Finché la dimensione Customer risulta inclusa nella tabella sul foglio, altrimenti il risultato di OrdersByCustomer è 3, 2.

Esempio	Risultato
Presupponendo che la tabella **Temp** venga caricata come nell'esempio precedente: `LOAD Count(OrderNumber) as TotalOrderNumber Resident Temp;`	`TotalOrderNumber` `10`

Presupponendo che la tabella **Temp** venga caricata come nel primo esempio:
`LOAD Count(distinct OrderNumber) as TotalOrderNumber Resident Temp;`
	`TotalOrderNumber`
	`8`
	Poiché vi sono due valori di OrderNumber con lo stesso valore, 1, e un solo valore null.

Count - funzione per grafici

Count() viene utilizzato per aggregare il numero di valori, di testo e numerici, in ciascuna dimensione del grafico.

Sintassi:

```
Count({[SetExpression] [DISTINCT] [TOTAL [<fld {,fld}>]]} expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Data

Customer	Product	OrderNumber	UnitSales	Unit Price
Astrida	AA	1	4	16
Astrida	AA	7	10	15
Astrida	BB	4	9	9
Betacab	BB	6	5	10
Betacab	CC	5	2	20
Betacab	DD	1	25	25
Canutility	AA	3	8	15
Canutility	CC			19
Divadip	AA	2	4	16
Divadip	DD	3		25

Negli esempi riportati di seguito si presuppone che siano selezionati tutti i clienti, tranne nei casi in cui è diversamente specificato.

Esempi e risultati

Esempio	Risultato
`Count(OrderNumber)`	10, perché sono presenti 10 campi che possono avere un valore per OrderNumber e vengono contati tutti i record, inclusi quelli vuoti. *"0" viene conteggiato come valore e non una cella vuota. Tuttavia, se una misura viene aggregata a 0 per una dimensione, tale dimensione non verrà inclusa nei grafici.*
`Count(Customer)`	10, perché Count valuta il numero di occorrenze in tutti i campi.
`Count(DISTINCT [Customer])`	4, perché, con l'utilizzo del qualificatore Distinct, Count valuta solo le occorrenze univoche.

Esempio	Risultato
Con il cliente Canutility selezionato `Count (OrderNumber)/Count ({1} TOTAL OrderNumber)`	0,2 perché l'espressione restituisce il numero di ordini del cliente selezionato come percentuale degli ordini di tutti i clienti. In questo caso 2/10.
Dato che sono stati selezionati i clienti Astrida e Canutility `Count(TOTAL <Product> OrderNumber)`	5 perché si tratta del numero di ordini effettuati per i prodotti solo per i clienti selezionati e vengono conteggiate le celle vuote.

Dati utilizzati negli esempi:

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|UnitPrice
Astrida|AA|1|4|16
Astrida|AA|7|10|15
Astrida|BB|4|9|9
Betacab|CC|6|5|10
Betacab|AA|5|2|20
Betacab|BB|1|25| 25
Canutility|AA|3|8|15
Canutility|CC|||19
Divadip|CC|2|4|16
Divadip|DD|3|1|25
] (delimiter is '|');
```

MissingCount

MissingCount() restituisce il numero dei valori mancanti aggregati nell'espressione, come definito da una clausola **group by**.

Sintassi:

```
MissingCount ( [ distinct ] expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr Expression	L'espressione o il campo contenente i dati da misurare.

Argomento	Descrizione
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempi di script

Esempio	Risultato

`Temp:`
`LOAD * inline [`
`Customer|Product|OrderNumber|UnitSales|UnitPrice`
`Astrida|AA|1|4|16`
`Astrida|AA|7|10|15`
`Astrida|BB|4|9|9`
`Betacab|CC|6|5|10`
`Betacab|AA|5|2|20`
`Betacab|BB||| 25`
`Canutility|AA|||15`
`Canutility|CC| ||19`
`Divadip|CC|2|4|16`
`Divadip|DD|3|1|25`
`] (delimiter is '|');`
`MissCount1:`
`LOAD Customer,MissingCount(OrderNumber) as MissingOrdersByCustomer Resident Temp Group By Customer;`
 	
`Load MissingCount(OrderNumber) as TotalMissingCount Resident Temp;`
	`Customer MissingOrdersByCustomer`
	`Astrida 0`
	`Betacab 1`
	`Canutility 2`
	`Divadip 0`
	
	La seconda istruzione restituisce:
	
	`TotalMissingCount`
	`3`
	in una tabella con tale dimensione.

Presupponendo che la tabella **Temp** venga caricata come nell'esempio precedente:
 	
`LOAD MissingCount(distinct OrderNumber) as TotalMissingCountDistinct Resident Temp;`
	`TotalMissingCountDistinct`
	`1`
	Perché è presente un unico valore OrderNumber mancante.

MissingCount - funzione per grafici

MissingCount() viene utilizzata per aggregare il numero di valori mancanti in ciascuna dimensione del grafico. I valori mancanti sono tutti valori non numerici.

Sintassi:

MissingCount({[SetExpression] [**DISTINCT**] [**TOTAL** [<fld {,fld}>]]} expr**)**

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Data

Customer	Product	OrderNumber	UnitSales	Unit Price
Astrida	AA	1	4	16
Astrida	AA	7	10	15
Astrida	BB	4	9	9
Betacab	BB	6	5	10
Betacab	CC	5	2	20
Betacab	DD			25
Canutility	AA			15
Canutility	CC			19
Divadip	AA	2	4	16
Divadip	DD	3		25

Esempi e risultati

Esempio	Risultato
`MissingCount([OrderNumber])`	3 perché 3 dei 10 campi OrderNumber sono vuoti *"0" viene conteggiato come valore e non una cella vuota. Tuttavia, se una misura viene aggregata a 0 per una dimensione, tale dimensione non verrà inclusa nei grafici.*
`MissingCount ([OrderNumber])/MissingCount ({1} Total [OrderNumber])`	L'espressione restituisce il numero di ordini incompleti del cliente selezionato come frazione degli ordini incompleti di tutti i clienti. Esiste un totale di 3 valori mancanti per OrderNumber per tutti i clienti. Pertanto, per ciascun Customer che presenta un valore mancante per Product il risultato è 1/3.

Dati utilizzati nell'esempio:

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|UnitPrice
Astrida|AA|1|4|16
Astrida|AA|7|10|15
Astrida|BB|4|9|9
Betacab|CC|6|5|10
Betacab|AA|5|2|20
Betacab|BB||| 25
Canutility|AA|||15
Canutility|CC| ||19
Divadip|CC|2|4|16
Divadip|DD|3|1|25
] (delimiter is '|');
```

NullCount

NullCount() restituisce il numero dei valori NULL aggregati nell'espressione, come definito da una clausola **group by**.

Sintassi:

```
NullCount ( [ distinct ] expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr Expression	L'espressione o il campo contenente i dati da misurare.
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempi di script

Esempio	Risultato

`Set NULLINTERPRET = NULL;`
`Temp:`
`LOAD * inline [`
`Customer|Product|OrderNumber|UnitSales|CustomerID`
`Astrida|AA|1|10|1`
`Astrida|AA|7|18|1`
`Astrida|BB|4|9|1`
`Astrida|CC|6|2|1`
`Betacab|AA|5|4|2`
`Betacab|BB|2|5|2`
`Betacab|DD||`
`Canutility|AA|3|8|`
`Canutility|CC|NULL||`
`] (delimiter is '|');`
`Set NULLINTERPRET=;`
`NullCount1:`
`LOAD Customer,NullCount(OrderNumber) as NullOrdersByCustomer Resident Temp Group By Customer;`
 	
`LOAD NullCount(OrderNumber) as TotalNullCount Resident Temp;`
	`Customer NullOrdersByCustomer`
	`Astrida 0`
	`Betacab 0`
	`Canutility 1`
	
	La seconda istruzione restituisce:
	`TotalNullCount`
	`1`
	in una tabella con tale dimensione, poiché solo un record contiene un valore null.

NullCount - funzione per grafici

NullCount() viene utilizzata per aggregare il numero di valori NULL in ciascuna dimensione del grafico.

Sintassi:

```
NullCount({[SetExpression][DISTINCT] [TOTAL [<fld {,fld}>]]} expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
set_ expression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Esempi e risultati

Esempio	Risultato
NullCount ([OrderNumber])	1 perché è stato introdotto un valore null utilizzando una stringa NullInterpret nell'istruzione **LOAD** inline.

Dati utilizzati nell'esempio:

```
Set NULLINTERPRET = NULL;
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|CustomerID
Astrida|AA|1|10|1
Astrida|AA|7|18|1
Astrida|BB|4|9|1
Astrida|CC|6|2|1
Betacab|AA|5|4|2
Betacab|BB|2|5|2
Betacab|DD|||
Canutility|AA|3|8|
Canutility|CC|NULL||
] (delimiter is '|');
```

```
Set NULLINTERPRET=;
```

NumericCount

NumericCount() restituisce il numero dei valori numerici presenti nell'espressione, come definito da una clausola **group by**.

Sintassi:

```
NumericCount ( [ distinct ] expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr Expression	L'espressione o il campo contenente i dati da misurare.
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempio di script

Esempio	Risultato
`LOAD NumericCount(OrderNumber) as TotalNumericCount Resident Temp;`	La seconda istruzione restituisce: TotalNumericCount 7 in una tabella con tale dimensione.
Presupponendo che la tabella **Temp** venga caricata come nell'esempio precedente: `LOAD NumericCount(distinct OrderNumber) as TotalNumericCountDistinct Resident Temp;`	TotalNumericCountDistinct 6 Poiché è presente un valore OrderNumber che ne duplica un altro, il risultato è 6 valori non duplicati.

Esempio:

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|UnitPrice
Astrida|AA|1|4|16
```

```
Astrida|AA|7|10|15
Astrida|BB|4|9|9
Betacab|CC|6|5|10
Betacab|AA|5|2|20
Betacab|BB||| 25
Canutility|AA|||15
Canutility|CC| ||19
Divadip|CC|2|4|16
Divadip|DD|7|1|25
] (delimiter is '|');
NumCount1:
LOAD Customer,NumericCount(OrderNumber) as NumericCountByCustomer Resident Temp Group By
Customer;
```

Tabella risultante

Customer	NumericCountByCustomer
Astrida	3
Betacab	2
Canutility	0
Divadip	2

NumericCount - funzione per grafici

NumericCount() aggrega il numero di valori numerici in ciascuna dimensione del grafico.

Sintassi:

```
NumericCount({[SetExpression] [DISTINCT] [TOTAL [<fld {,fld}>]]} expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
set_ expression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.

Argomento	Descrizione
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Data

Customer	Product	OrderNumber	UnitSales	Unit Price
Astrida	AA	1	4	16
Astrida	AA	7	10	15
Astrida	BB	4	9	1
Betacab	BB	6	5	10
Betacab	CC	5	2	20
Betacab	DD			25
Canutility	AA			15
Canutility	CC			19
Divadip	AA	2	4	16
Divadip	DD	3		25

Negli esempi riportati di seguito si presuppone che siano selezionati tutti i clienti, tranne nei casi in cui è diversamente specificato.

Esempi e risultati

Esempio	Risultato
`NumericCount ([OrderNumber])`	7 perché 3 dei 10 campi in OrderNumber sono vuoti. *"0" viene conteggiato come valore e non una cella vuota. Tuttavia, se una misura viene aggregata a 0 per una dimensione, tale dimensione non verrà inclusa nei grafici.*

Esempio	Risultato
`NumericCount ([Product])`	0 perché tutti i nomi di prodotto sono riportati con il testo. In genere questa funzione può essere utilizzata per verificare che a nessun campo di testo sia stato assegnato contenuto numerico.
`NumericCount (DISTINCT [OrderNumber])/Count (DISTINCT [OrderNumber)]`	Conta tutto il numero di numeri distinti di ordine numerico e lo divide per il numero dei numeri di ordine numerico e non numerico. Questo sarà 1 se tutti i valori di campo sono numerici. È in genere possibile utilizzare questa operazione per verificare che tutti i valori di campo siano numerici. Nell'esempio sono presenti 7 valori numerici distinti per OrderNumber di 8 valori distinti numerici e non numerici, pertanto l'espressione restituisce 0.875.

Dati utilizzati nell'esempio:

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|UnitPrice
Astrida|AA|1|4|16
Astrida|AA|7|10|15
Astrida|BB|4|9|9
Betacab|CC|6|5|10
Betacab|AA|5|2|20
Betacab|BB||| 25
Canutility|AA|||15
Canutility|CC| ||19
Divadip|CC|2|4|16
Divadip|DD|3|1|25
] (delimiter is '|');
```

TextCount

TextCount() restituisce il numero dei valori di campo non numerici aggregati nell'espressione, come definito da una clausola **group by**.

Sintassi:

```
TextCount ( [ distinct ] expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr Expression	L'espressione o il campo contenente i dati da misurare.
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempio:

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|UnitPrice
Astrida|AA|1|4|16
Astrida|AA|7|10|15
Astrida|BB|4|9|9
Betacab|CC|6|5|10
Betacab|AA|5|2|20
Betacab|BB||| 25
Canutility|AA|||15
Canutility|CC| ||19
Divadip|CC|2|4|16
Divadip|DD|3|1|25
] (delimiter is '|');
TextCount1:
LOAD Customer,TextCount(Product) as ProductTextCount Resident Temp Group By Customer;
```

Tabella risultante

Customer	ProductTextCount
Astrida	3
Betacab	3
Canutility	2
Divadip	2

Esempio:

```
LOAD Customer,TextCount(OrderNumber) as OrderNumberTextCount Resident Temp Group By Customer;
```

Tabella risultante

Customer	OrderNumberTextCount
Astrida	0
Betacab	1
Canutility	2
Divadip	0

TextCount - funzione per grafici

TextCount() viene utilizzata per aggregare il numero di valori di campo che non sono numerici in ciascuna dimensione del grafico.

Sintassi:

```
TextCount({[SetExpression] [DISTINCT] [TOTAL [<fld {,fld}>]]} expr)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Data

Customer	Product	OrderNumber	UnitSales	Unit Price
Astrida	AA	1	4	16
Astrida	AA	7	10	15
Astrida	BB	4	9	1
Betacab	BB	6	5	10
Betacab	CC	5	2	20
Betacab	DD			25

Customer	Product	OrderNumber	UnitSales	Unit Price
Canutility	AA			15
Canutility	CC			19
Divadip	AA	2	4	16
Divadip	DD	3		25

Esempi e risultati

Esempio	Risultato
`TextCount ([Product])`	10 perché tutti i 10 campi in Product sono campi di testo. *"0" viene conteggiato come valore e non una cella vuota. Tuttavia, se una misura viene aggregata a 0 per una dimensione, tale dimensione non verrà inclusa nei grafici.L e celle vuote vengono considerate come non testo e non vengono calcolate da TextCount.*
`TextCount ([OrderNumber])`	3 perché le celle vuote vengono conteggiate. In genere, può essere utilizzato per verificare che nessun campo numerico presenti valori testuali o valori diversi da zero.
`TextCount (DISTINCT [Product])/Count ([Product)]`	Calcola l'intero numero di valori testuali distinti di Product (4) e lo divide per il numero totale di valori in Product (10). Il risultato è 0.4.

Dati utilizzati nell'esempio:

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|UnitPrice
Astrida|AA|1|4|16
Astrida|AA|7|1|15
Astrida|BB|4|9|9
Betacab|CC|6|5|10
Betacab|AA|5|2|20
Betacab|BB|||| 25
Canutility|AA|||15
Canutility|CC|||19
Divadip|CC|2|4|16
Divadip|DD|3|1|25
] (delimiter is '|');
```

Funzioni di aggregazione finanziaria

In questa sezione vengono descritte le funzioni di aggregazione per le operazioni finanziarie relative ai pagamenti e al flusso di cassa.

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Funzioni di aggregazione finanziaria nello script di caricamento dei dati

IRR

IRR() restituisce il tasso di rendimento interno aggregato per una serie di flussi di cassa rappresentati dai numeri nell'espressione ripetuti su un numero di record, come definito da una clausola group by.

```
IRR (expression)
```

XIRR

XIRR() restituisce il tasso di rendimento interno aggregato per una programmazione di flussi di cassa (non necessariamente periodici) rappresentati da coppie di numeri in **pmt** e **date** ripetuti su un insieme di record, come definito da una clausola group by. Tutti i pagamenti sono scontati in base ad un anno composto da 365 giorni.

```
XIRR (valueexpression, dateexpression )
```

NPV

La funzione script **NPV()** prende un tasso di sconto e più valori ordinati per periodo. I flussi in entrata (redditi) sono positivi e i flussi in uscita (pagamenti futuri) sono considerati valori negativi per questi calcoli. Si verificano alla fine di ogni periodo.

```
VAN (rate, expression)
```

XNPV

La funzione di script **XNPV()** prende le date specifiche corrispondenti a ciascun flusso di cassa da attualizzare, oltre al tasso di sconto. È diversa dalla funzione **NPV()**, poiché **NPV()** assume che tutti i periodi di tempo siano uguali. Per questo motivo, **XNPV()** è più preciso di **NPV()**.

```
XNPV (rate, valueexpression, dateexpression)
```

Funzioni di aggregazione finanziaria nelle espressioni grafiche

Queste funzioni di aggregazione possono essere utilizzate nei grafici.

IRR

IRR() restituisce il tasso di rendimento interno aggregato di una serie di flussi di cassa rappresentati dai numeri dell'espressione data da **value** ripetuti sulle dimensioni del grafico.

```
IRR - funzione per grafici[TOTAL [<fld {,fld}>]] value)
```

NPV

NPV() restituisce il valore attuale netto aggregato di un investimento basato su un **discount_rate** per periodo e una serie di pagamenti futuri (valori negativi) ed entrate (valori positivi) rappresentati dai numeri in **value** ripetuti sulle dimensioni del grafico. Si presuppone che i pagamenti e le entrate avvengano alla fine di ciascun periodo.

```
NPV - funzione per grafici([TOTAL [<fld {,fld}>]] discount_rate, value)
```

XIRR

XIRR() restituisce il tasso di rendimento interno aggregato per una programmazione di flussi di cassa (non necessariamente periodica) rappresentati da coppie di numeri nelle espressioni date da **pmt** e **date** ripetuti sulle dimensioni del grafico. Tutti i pagamenti sono scontati in base ad un anno composto da 365 giorni.

```
XIRR - funzione per grafici (page 384)([TOTAL [<fld {,fld}>]] pmt, date)
```

XNPV

XNPV() restituisce il valore netto attuale aggregato per una programmazione di flussi di cassa (non necessariamente periodica) rappresentati da coppie di numeri nelle espressioni date da **pmt** e **date** ripetuti sulle dimensioni del grafico. Tutti i pagamenti sono scontati in base ad un anno composto da 365 giorni.

```
XNPV - funzione per grafici([TOTAL [<fld{,fld}>]] discount_rate, pmt, date)
```

IRR

IRR() restituisce il tasso di rendimento interno aggregato per una serie di flussi di cassa rappresentati dai numeri nell'espressione ripetuti su un numero di record, come definito da una clausola group by.

Questi flussi di cassa non devono essere necessariamente pari, come dovrebbero essere per una annualità. Tuttavia, i flussi di cassa devono ricorrere a intervalli regolari, ad esempio ogni mese o ogni anno. Il tasso di rendimento interno è il tasso di interesse ricevuto per un investimento che consiste in pagamenti (valori negativi) e in entrate (valori positivi) che ricorrono ad intervalli regolari. La funzione necessita di almeno un valore positivo e di uno negativo per essere calcolata.

Sintassi:

```
IRR(value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	L'espressione o il campo contenente i dati da misurare.

Limiti:

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Esempi e risultati:

Esempi e risultati

Esempio	Anno	IRR2013
Cashflow: LOAD 2013 as Year, * inline [Date|Discount|Payments 2013-01-01|0.1|-10000 2013-03-01|0.1|3000 2013-10-30|0.1|4200 2014-02-01|0.2|6800] (delimiter is '|'); Cashflow1: LOAD Year,IRR(Payments) as IRR2013 Resident Cashflow Group By Year;	2013	0.1634

IRR - funzione per grafici

IRR() restituisce il tasso di rendimento interno aggregato di una serie di flussi di cassa rappresentati dai numeri dell'espressione data da **value** ripetuti sulle dimensioni del grafico.

Questi flussi di cassa non devono essere necessariamente pari, come dovrebbero essere per una annualità. Tuttavia, i flussi di cassa devono ricorrere a intervalli regolari, ad esempio ogni mese o ogni anno. Il tasso di rendimento interno è il tasso di interesse ricevuto per un investimento che consiste in pagamenti (valori negativi) ed entrate (valori positivi) che ricorrono ad intervalli regolari. La funzione necessita di almeno un valore positivo e un valore negativo da calcolare.

Sintassi:

```
IRR([TOTAL [<fld {,fld}>]] value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	L'espressione o il campo contenente i dati da misurare.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {,fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Esempi e risultati:

Esempi e risultati

Esempio	Risultato
`IRR (Payments)`	0.1634 Si presuppone che i pagamenti siano di natura periodica, ad esempio mensili. *Il campo Date viene utilizzato nell'esempio XIRR dove i pagamenti possono essere non periodici purché vengano fornite le date in cui sono effettuati.*

Dati utilizzati negli esempi:

```
Cashflow:
LOAD 2013 as Year, * inline [
Date|Discount|Payments
2013-01-01|0.1|-10000
2013-03-01|0.1|3000
2013-10-30|0.1|4200
2014-02-01|0.2|6800
] (delimiter is '|');
```

Vedere anche:

p *XIRR - funzione per grafici (page 384)*
p *Aggr - funzione per grafici (page 541)*

VAN

La funzione script **NPV()** prende un tasso di sconto e più valori ordinati per periodo. I flussi in entrata (redditi) sono positivi e i flussi in uscita (pagamenti futuri) sono considerati valori negativi per questi calcoli. Si verificano alla fine di ogni periodo.

Il Valore Attuale Netto, o VAN, viene utilizzato per calcolare il valore totale attuale di un flusso di cassa futuro. Per calcolare il VAN, dobbiamo stimare i flussi di cassa futuri per ogni periodo e determinare il tasso di sconto corretto. La funzione script **NPV()** prende un tasso di sconto e più valori ordinati per periodo. I flussi in entrata (redditi) sono positivi e i flussi in uscita (pagamenti futuri) sono considerati valori negativi per questi calcoli. Si verificano alla fine di ogni periodo.

Sintassi:

```
NPV(discount_rate, value)
```

Tipo di dati restituiti: numerico. Per impostazione predefinita, il risultato sarà formattato come valuta.

La formula per calcolare il valore attuale netto è:

$$NPV = \sum_{t=1}^{n} \frac{R_t}{(1+i)^t}$$

dove:

- R_t = Flussi di cassa netti in entrata e in uscita durante un singolo periodo `t`
- `i` = Tasso di sconto o rendimento che si potrebbe ottenere con investimenti alternativi
- `t` = Numero di periodi del timer

Argomenti

Argomento	Descrizione
discount_rate	**discount_rate** è il tasso percentuale di sconto applicato. Un valore di 0,1 indica un tasso di sconto del 10%.
value	Questo campo contiene i valori di più periodi ordinati per periodo. Il primo valore viene assunto come flusso di cassa alla fine del periodo 1, e così via.

Limiti:

La funzione `NPV()` presenta le seguenti limitazioni:

- I valori di testo, i valori NULL e i valori mancanti vengono ignorati.
- I valori del flusso di cassa devono essere in ordine di periodo crescente.

Casi di utilizzo

`NPV()` è una funzione finanziaria utilizzata per verificare la redditività del progetto e per ricavare altre misure. Questa funzione è utile quando i flussi di cassa sono disponibili come dati non elaborati.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Pagamento singolo (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di un progetto e del suo flusso di cassa per un periodo, che viene caricato in una tabella denominata `CashFlow`.
- Un caricamento residente dalla tabella `CashFlow`, utilizzato per calcolare il campo VAN del progetto in una tabella denominata `NPV`.
- Un tasso di sconto fisso del 10%, utilizzato per il calcolo del VAN.
- Un'istruzione `Group By`, utilizzata per raggruppare tutti i pagamenti del progetto.

Script di caricamento

```
CashFlow:
Load
*
Inline
[
PrjId,PeriodId,Values
1,1,1000
];

NPV:
Load
       PrjId,
       NPV(0.1,Values) as NPV //Discount Rate of 10%
Resident CashFlow
Group By PrjId;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `PrjId`
- `NPV`

Tabella dei risultati

PrjId	VAN
1	$909.09

Per un singolo pagamento di $1000 da ricevere alla fine di un periodo, con un tasso di sconto del 10% per periodo, il VAN è pari a $1000 diviso per (1 + tasso di sconto). Il VAN effettivo è pari a $909,09.

Esempio 2 - Pagamenti multipli (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di un progetto e del suo flusso di cassa per più periodi, che viene caricato in una tabella denominata `CashFlow`.
- Un caricamento residente dalla tabella `CashFlow`, utilizzato per calcolare il campo VAN del progetto in una tabella denominata `NPV`.
- Nel calcolo del VAN viene utilizzato un tasso di sconto fisso del 10% (0,1).
- Un'istruzione `Group By`, utilizzata per raggruppare tutti i pagamenti del progetto.

Script di caricamento

```
CashFlow:
Load
*
Inline
[
PrjId,PeriodId,Values
1,1,1000
1,2,1000
];

NPV:
Load
       PrjId,
       NPV(0.1,Values) as NPV //Discount Rate of 10%
Resident CashFlow
Group By PrjId;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `PrjId`
- `NPV`

Tabella dei risultati

PrjId	VAN
1	$1735.54

Per pagamenti di $1000 dollari da ricevere alla fine di due periodi, con un tasso di sconto del 10% per periodo, il VAN effettivo è pari a $1735,54.

Esempio 3 - Pagamenti multipli (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Tassi di sconto per due progetti, che vengono caricati in una tabella denominata `Project`.
- Flussi di cassa per più periodi per ogni progetto per ID progetto e ID periodo. Questo ID periodo può essere utilizzato per ordinare i record nel caso in cui i dati non siano ordinati.
- La combinazione di `NoConcatenate`, caricamenti residenti e la funzione `Left Join` per creare una tabella temporanea, `tmpNPV`. La tabella combina i record di `Project` e delle tabelle `CashFlow` in una tabella piatta. In questa tabella i tassi di sconto saranno ripetuti per ogni periodo.
- Un caricamento residente dalla tabella `tmpNPV`, utilizzato per calcolare il campo VAN per ciascun progetto in una tabella denominata `NPV`.
- Il tasso di sconto a valore singolo associato a ciascun progetto. Questo dato viene recuperato con la funzione `only()` e viene utilizzato nel calcolo del VAN per ogni progetto.
- Un'istruzione `Group By`, utilizzata per raggruppare tutti i pagamenti per ciascun progetto in base all'ID progetto.

Per evitare che vengano caricati dati sintetici o ridondanti nel modello di dati, la tabella `tmpNPV` viene eliminata alla fine dello script.

Script di caricamento

```
Project:
Load * inline [
PrjId,Discount_Rate
1,0.1
2,0.15
];

CashFlow:
Load
*
Inline
[
PrjId,PeriodId,Values
1,1,1000
```

```
1,2,1000
1,3,1000
2,1,500
2,2,500
2,3,1000
2,4,1000
];

tmpNPV:
NoConcatenate Load *
Resident Project;
Left Join
Load *
Resident CashFlow;

NPV:
Load
       PrjId,
       NPV(Only(Discount_Rate),Values) as NPV //Discount Rate will be 10% for Project 1 and 15% for
Project 2
Resident tmpNPV
Group By PrjId;

Drop table tmpNPV;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- PrjId
- NPV

Tabella dei risultati

PrjId	VAN
1	$2486.85
2	$2042.12

L'ID progetto 1 prevede di ricevere pagamenti di $1000 alla fine di tre periodi, con un tasso di sconto del 10% per periodo. Pertanto, il VAN effettivo è pari a 2486,85 dollari.

L'ID progetto 2 prevede due pagamenti di $500 e altri due pagamenti di $1000 in quattro periodi con un tasso di sconto del 15%. Pertanto, il VAN effettivo è pari a $2042.12.

Esempio 4 - Esempio di redditività del progetto (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Tassi di sconto e investimenti iniziali (periodo 0) per due progetti, caricati in una tabella denominata `Project`.
- Flussi di cassa per più periodi per ogni progetto per ID progetto e ID periodo. Questo ID periodo può essere utilizzato per ordinare i record nel caso in cui i dati non siano ordinati.
- La combinazione di `NoConcatenate`, caricamenti residenti e la funzione `Left Join` per creare una tabella temporanea, `tmpNPV`. La tabella combina i record di `Project` e delle tabelle `CashFlow` in una tabella piatta. In questa tabella i tassi di sconto saranno ripetuti per ogni periodo.
- Il tasso di sconto a valore singolo associato a ciascun progetto, che viene recuperato con la funzione `only()` e utilizzato nel calcolo del VAN per ciascun progetto.
- Un caricamento residente dalla tabella tmpNPV viene utilizzato per calcolare il campo VAN per ciascun progetto in una tabella denominata NPV.
- Per calcolare l'indice di redditività del progetto viene creato un campo aggiuntivo che divide il VAN per l'investimento iniziale di ogni progetto.
- Una dichiarazione di raggruppamento per ID progetto viene utilizzata per raggruppare tutti i pagamenti per ogni progetto.

Per evitare che vengano caricati dati sintetici o ridondanti nel modello di dati, la tabella `tmpNPV` viene eliminata alla fine dello script.

Script di caricamento

```
Project:
Load * inline [
PrjId,Discount_Rate, Initial_Investment
1,0.1,100000
2,0.15,100000
];

CashFlow:
Load
*
Inline
[
PrjId,PeriodId,Values,
1,1,35000
1,2,35000
1,3,35000
2,1,30000
2,2,40000
2,3,50000
2,4,60000
];

tmpNPV:
NoConcatenate Load *
Resident Project;
Left Join
Load *
Resident CashFlow;
```

```
NPV:
Load
        PrjId,
        NPV(Only(Discount_Rate),Values) as NPV, //Discount Rate will be 10% for Project 1 and
15% for Project 2
    NPV(Only(Discount_Rate),Values)/ Only(Initial_Investment) as Profitability_Index
Resident tmpNPV
Group By PrjId;

Drop table tmpNPV;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- PrjId
- NPV

Creare la seguente misura:

```
=only(Profitability_Index)
```

Tabella dei risultati

PrjId	VAN	=only(Profitability_Index)
1	$87039.82	0.87
2	$123513.71	1.24

Il progetto ID 1 ha un VAN effettivo di $87039,82 e un investimento iniziale di $100.000. Pertanto, l'indice di redditività è pari a 0,87. Poiché è inferiore a 1, il progetto non è redditizio.

Il progetto ID 2 ha un VAN effettivo di $123513,71 e un investimento iniziale di $100.000. Pertanto, l'indice di redditività è pari a 1.24. Poiché è superiore a 1, il progetto è redditizio.

NPV - funzione per grafici

NPV() restituisce il valore attuale netto aggregato di un investimento basato su un **discount_rate** per periodo e una serie di pagamenti futuri (valori negativi) ed entrate (valori positivi) rappresentati dai numeri in **value** ripetuti sulle dimensioni del grafico. Si presuppone che i pagamenti e le entrate avvengano alla fine di ciascun periodo.

Sintassi:

```
NPV([TOTAL [<fld {,fld}>]] discount_rate, value)
```

Tipo di dati restituiti: numerico Per impostazione predefinita, il risultato sarà formattato come valuta.

Argomenti:

Argomenti

Argomento	Descrizione
discount_rate	**discount_rate** is the rate of discount over the length of the period.**discount_rate** è il tasso percentuale di sconto applicato.
value	L'espressione o il campo contenente i dati da misurare.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili. Il qualificatore **TOTAL** può essere seguito da un elenco di uno o più nomi di campo tra parentesi acute. Questi nomi di campo devono essere un sottogruppo delle variabili di dimensione del grafico. In questo caso, il calcolo verrà effettuato ignorando tutte le variabili di dimensione del grafico eccetto quelle elencate, ad esempio un valore verrà restituito per ogni combinazione di valori di campo nei campi delle dimensioni elencati. Anche i campi che non sono attualmente una dimensione in un grafico possono essere inclusi nell'elenco. Questo può essere utile in caso di dimensioni di gruppo, dove i campi di dimensione non sono fissi. Elencando tutte le variabili nel gruppo viene attivata la funzione in corrispondenza delle modifiche del livello di drill-down.

Limiti:

discount_rate e **value** non devono contenere funzioni di aggregazione, a meno che queste aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Esempi e risultati:

Esempi e risultati

Esempio	Risultato
`NPV(Discount, Payments)`	-$540.12

Dati utilizzati negli esempi:

```
Cashflow:
LOAD 2013 as Year, * inline [
```

```
Date|Discount|Payments
2013-01-01|0.1|-10000
2013-03-01|0.1|3000
2013-10-30|0.1|4200
2014-02-01|0.2|6800
] (delimiter is '|');
```

Vedere anche:

p *XNPV - funzione per grafici (page 391)*
p *Aggr - funzione per grafici (page 541)*

XIRR

XIRR() restituisce il tasso di rendimento interno aggregato per una programmazione di flussi di cassa (non necessariamente periodici) rappresentati da coppie di numeri in **pmt** e **date** ripetuti su un insieme di record, come definito da una clausola group by. Tutti i pagamenti sono scontati in base ad un anno composto da 365 giorni.

Sintassi:

```
XIRR(pmt, date )
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
pmt	Pagamenti. L'espressione o il campo contenente i flussi di cassa corrispondenti alla programmazione di pagamento fornita in **date**.
date	L'espressione o il campo contenente la programmazione di date corrispondente ai pagamenti con flusso di cassa forniti in **pmt**.

Limiti:

Se una o entrambe le parti di una coppia di dati include o includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Esempi e risultati

Esempio	Anno	XIRR2013
Cashflow: LOAD 2013 as Year, * inline [Date|Discount|Payments 2013-01-01|0.1|-10000 2013-03-01|0.1|3000 2013-10-30|0.1|4200 2014-02-01|0.2|6800] (delimiter is '|'); Cashflow1: LOAD Year,XIRR(Payments, Date) as XIRR2013 Resident Cashflow Group By Year;	2013	0.5385

XIRR - funzione per grafici

XIRR() restituisce il tasso di rendimento interno aggregato per una programmazione di flussi di cassa (non necessariamente periodica) rappresentati da coppie di numeri nelle espressioni date da **pmt** e **date** ripetuti sulle dimensioni del grafico. Tutti i pagamenti sono scontati in base ad un anno composto da 365 giorni.

Sintassi:

```
XIRR([TOTAL [<fld {,fld}>]] pmt, date)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
pmt	Pagamenti. L'espressione o il campo contenente i flussi di cassa corrispondenti alla programmazione di pagamento fornita in **date**.
date	L'espressione o il campo contenente la programmazione di date corrispondente ai pagamenti con flusso di cassa forniti in **pmt**.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

pmt e **date** non devono contenere funzioni di aggregazione, a meno che queste aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Esempi e risultati:

Esempi e risultati

Esempio	Risultato
`XIRR(Payments, Date)`	0.5385

Dati utilizzati negli esempi:

```
Cashflow:
LOAD 2013 as Year, * inline [
Date|Discount|Payments
2013-01-01|0.1|-10000
2013-03-01|0.1|3000
2013-10-30|0.1|4200
2014-02-01|0.2|6800
] (delimiter is '|');
```

Vedere anche:

p *IRR - funzione per grafici (page 373)*
p *Aggr - funzione per grafici (page 541)*

XNPV

La funzione di script **XNPV()** prende le date specifiche corrispondenti a ciascun flusso di cassa da attualizzare, oltre al tasso di sconto. È diversa dalla funzione **NPV()**, poiché **NPV()** assume che tutti i periodi di tempo siano uguali. Per questo motivo, **XNPV()** è più preciso di **NPV()**.

Sintassi:

```
XNPV(discount_rate, pmt, date)
```

Tipo di dati restituiti: numerico. Per impostazione predefinita, il risultato sarà formattato come valuta.

La formula per calcolare l'XNPV è:

$$XNPV = \sum_{i=1}^{n} \frac{P_i}{(1+rate)^{(di-d1)/365}}$$

dove:

- P_i = Flussi di cassa netti in entrata e in uscita durante un singolo periodo `i`
- d_1 = la data del primo pagamento
- d_i = la data del i^o pagamento
- `rate` = tasso di sconto

Il valore attuale netto, o VAN, viene utilizzato per calcolare il valore totale attuale di un flusso di cassa futuro. Per calcolare il VAN, dobbiamo stimare i flussi di cassa futuri per ogni periodo e determinare il tasso di sconto corretto.

`XNPV()` prende un tasso di sconto e più valori ordinati per periodo. I flussi in entrata (redditi) sono positivi e i flussi in uscita (pagamenti futuri) sono considerati valori negativi. Si verificano alla fine di ogni periodo.

Argomenti

Argomento	Descrizione
discount_rate	**discount_rate** è il tasso percentuale di sconto applicato. Un valore di 0,1 indica un tasso di sconto del 10%.
value	Questo campo contiene i valori del flusso di cassa. Il primo valore viene assunto come flusso di cassa all'inizio e la data corrispondente viene utilizzata come riferimento per il calcolo del valore attuale di tutti i flussi di cassa futuri. *__XNPV()__ non sconta il flusso di cassa iniziale. I pagamenti successivi sono scontati in base ad un anno composto da 365 giorni. Questo è diverso da __NPV()__, dove ogni pagamento viene scontato.*
date	Questo campo contiene la data in cui si è verificato il flusso di cassa (**value**, il secondo parametro). Il primo valore viene utilizzato come data di inizio per il calcolo delle compensazioni per i flussi di cassa futuri.

Limiti:

Se sono presenti valori di testo, valori NULL e valori mancanti in una o entrambe le parti di una coppia di dati, la coppia di dati verrà ignorata.

Casi di utilizzo

- `XNPV()` è utilizzato nella modellistica finanziaria per calcolare il valore attuale netto (VAN) di un'opportunità di investimento.
- Grazie alla sua maggiore precisione, l'XNPV è preferito al VAN per tutti i tipi di modelli finanziari.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Pagamento singolo (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di un progetto e del suo flusso di cassa per un anno, in una tabella denominata `CashFlow`. La data iniziale per il calcolo è fissata al 1° luglio 2022, con un flusso di cassa netto pari a 0. Dopo un anno, si verifica un flusso di cassa di 1000 dollari.
- Un caricamento residente dalla tabella `CashFlow`, utilizzato per calcolare il campo `XNPV` del progetto in una tabella denominata `XNPV`.
- Nel calcolo del XNPV viene utilizzato un tasso di sconto fisso del 10% (0,1).
- Un'istruzione `Group By` viene utilizzata per raggruppare tutti i pagamenti del progetto.

Script di caricamento

```
CashFlow:
Load
*
Inline
[
PrjId,Dates,Values
1,'07/01/2022',0
1,'07/01/2023',1000
];

XNPV:
Load
       PrjId,
       XNPV(0.1,Values,Dates) as XNPV //Discount Rate of 10%
```

```
Resident CashFlow
Group By PrjId;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `PrjId`
- `XNPV`

Tabella dei risultati

PrjId	XNPV
1	$909.09

Secondo la formula, il valore XNPV per il primo record è 0, mentre per il secondo record il valore XNPV è di $909,09. Pertanto, il valore XNPV totale è $909.09.

Esempio 2 - Pagamenti multipli (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di un progetto e del suo flusso di cassa per un anno, in una tabella denominata `CashFlow`.
- Un caricamento residente dalla tabella `CashFlow`, utilizzato per calcolare il campo `XNPV` del progetto in una tabella denominata `XNPV`.
- Nel calcolo del XNPV viene utilizzato un tasso di sconto fisso del 10% (0,1).
- Un'istruzione `Group By` viene utilizzata per raggruppare tutti i pagamenti del progetto.

Script di caricamento

```
CashFlow:
Load
*
Inline
[
PrjId,Dates,Values
1,'07/01/2022',0
1,'07/01/2024',500
1,'07/01/2023',1000
];

XNPV:
Load
       PrjId,
```

```
      XNPV(0.1,Values,Dates) as XNPV //Discount Rate of 10%
Resident CashFlow
Group By PrjId;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- PrjId
- XNPV

Tabella dei risultati

PrjId	XNPV
1	$1322.21

In questo esempio, alla fine del primo anno si riceve un pagamento di $1000 dollari e alla fine del secondo anno se ne riceve uno di $500. Con un tasso di sconto del 10% per periodo, l'XNPV effettivo è pari a $1322,21.

Notare che solo la prima riga di dati deve fare riferimento alla data base per i calcoli. Per il resto delle righe, l'ordine non è importante, poiché il parametro della data viene utilizzato per calcolare il periodo trascorso.

Esempio 3 - Pagamenti multipli e flussi di cassa irregolari (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Tassi di sconto per due progetti in una tabella denominata `Project`.
- Flussi di cassa per più periodi per ogni progetto per ID progetto e Date. Il campo `Dates` viene utilizzato per calcolare la durata per la quale il tasso di sconto viene applicato al flusso di cassa. A parte il primo record (flusso di cassa iniziale e data), l'ordine dei record non è importante e la sua modifica non dovrebbe influire sui calcoli.
- Utilizzando una combinazione di `NoConcatenate`, caricamenti residenti e la funzione `Left Join`, viene creata una tabella temporanea, `tmpNPV`, che combina i record delle tabelle `Project` e `CashFlow` in una tabella piatta. In questa tabella i tassi di sconto saranno ripetuti per ogni flusso di cassa.
- Un caricamento residente dalla tabella `tmpNPV`, utilizzato per calcolare il campo `XNPV` per ciascun progetto in una tabella denominata `XNPV`.
- Il tasso di sconto a valore singolo associato a ciascun progetto viene recuperato tramite la funzione `only()` e utilizzato nel calcolo `XNPV` per ciascun progetto.
- Un'istruzione `Group By`, raggruppata per ID progetto, viene utilizzata per raggruppare tutti i pagamenti e le date corrispondenti per ogni progetto.

- Per evitare che vengano caricati dati sintetici o ridondanti nel modello di dati, la tabella `tmpXNPV` viene eliminata alla fine dello script.

Script di caricamento

```
Project:
Load * inline [
PrjId,Discount_Rate
1,0.1
2,0.15
];

CashFlow:
Load
*
Inline
[
PrjId,Dates,Values
1,'07/01/2021',0
1,'07/01/2022',1000
1,'07/01/2023',1000
2,'07/01/2020',0
2,'07/01/2023',500
2,'07/01/2024',1000
2,'07/01/2022',500
];

tmpXNPV:
NoConcatenate Load *
Resident Project;
Left Join
Load *
Resident CashFlow;

XNPV:
Load
       PrjId,
       XNPV(Only(Discount_Rate),Values,Dates) as XNPV //Discount Rate will be 10% for Project 1 and
15% for Project 2
Resident tmpXNPV
Group By PrjId;

Drop table tmpXNPV;
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `PrjId`
- `XNPV`

Tabella dei risultati

PrjId	XNPV
1	$1735.54
2	$278.36

Il progetto ID 1 ha un flusso di cassa iniziale di 0 dollari il 1° luglio 2021. Sono previsti due pagamenti di 1.000 dollari da ricevere alla fine di due anni successivi, a un tasso di sconto del 10% per periodo. Pertanto, il valore XNPV effettivo è pari a $1735.54.

Il progetto ID 2 ha un'uscita iniziale di $1.000 (quindi il segno negativo) il 1° luglio 2020. Dopo due anni è previsto un pagamento di $500. Dopo 3 anni è previsto un altro pagamento di $500. Infine, il 1° luglio 2024 è previsto un pagamento di $1.000. Con un tasso di sconto del 15%, l'XNPV effettivo è pari a $278,36.

XNPV - funzione per grafici

XNPV() restituisce il valore netto attuale aggregato per una programmazione di flussi di cassa (non necessariamente periodica) rappresentati da coppie di numeri nelle espressioni date da **pmt** e **date** ripetuti sulle dimensioni del grafico. Tutti i pagamenti sono scontati in base ad un anno composto da 365 giorni.

Sintassi:

```
XNPV([TOTAL [<fld{,fld}>]] discount_rate, pmt, date)
```

Tipo di dati restituiti: numerico Per impostazione predefinita, il risultato sarà formattato come valuta.

Argomenti:

Argomenti

Argomento	Descrizione
discount_rate	**discount_rate** is the rate of discount over the length of the period.**discount_rate** è il tasso percentuale di sconto applicato.
pmt	Pagamenti. L'espressione o il campo contenente i flussi di cassa corrispondenti alla programmazione di pagamento fornita in **date**.
date	L'espressione o il campo contenente la programmazione di date corrispondente ai pagamenti con flusso di cassa forniti in **pmt**.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

discount_rate, **pmt** e **date** non devono contenere funzioni di aggregazione, a meno che queste aggregazioni interne non contengano il qualificatore **TOTAL** o **ALL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Esempi e risultati:

Esempi e risultati

Esempio	Risultato
`XNPV(Discount, Payments, Date)`	-$3164.35

Dati utilizzati negli esempi:

```
Cashflow:
LOAD 2013 as Year, * inline [
Date|Discount|Payments
2013-01-01|0.1|-10000
2013-03-01|0.1|3000
2013-10-30|0.1|4200
2014-02-01|0.2|6800
] (delimiter is '|');
```

Vedere anche:

p *NPV - funzione per grafici (page 381)*
p *Aggr - funzione per grafici (page 541)*

Funzioni di aggregazione statistica

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Funzioni di aggregazione statistica nello script di caricamento dei dati

Le funzioni di aggregazione statistica seguenti possono essere utilizzate negli script.

Avg
Avg() restituisce il valore medio dei dati aggregati nell'espressione su un insieme di record, come definito da una clausola **group by**.

```
Avg  ([distinct] expression)
```

Correl

Correl() restituisce il coefficiente di correlazione aggregato per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record definito da una clausola **group by**.

```
Correl (x-expression, y-expression)
```

Fractile

Fractile() restituisce il valore corrispondente al frattale inclusivo (quantile) dei dati aggregati nell'espressione su un insieme di record, come definito da una clausola **group by**.

```
Fractile (expression, fractile)
```

FractileExc

FractileExc() restituisce il valore corrispondente al frattale esclusivo (quantile) dei dati aggregati nell'espressione su un insieme di record, come definito da una clausola **group by**.

```
FractileExc (expression, fractile)
```

Kurtosis

Kurtosis() restituisce il kurtosis dei dati nell'espressione su un insieme di record, come definito da una clausola **group by**.

```
Kurtosis  ([distinct ] expression )
```

LINEST_B

LINEST_B() restituisce il valore b aggregato (intercettazione sull'asse y) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_B (y-expression, x-expression [, y0 [, x0 ]])
```

LINEST_df

LINEST_DF() restituisce i gradi di libertà aggregati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_DF  (y-expression, x-expression [, y0 [, x0 ]])
```

LINEST_f

Questa funzione di script restituisce la statistica F aggregata ($r^2/(1-r^2)$) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record definito da una clausola **group by**.

```
LINEST_F (y-expression, x-expression [, y0 [, x0 ]])
```

LINEST_m

LINEST_M() restituisce il valore m aggregato (pendenza) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_M  (y-expression, x-expression [, y0 [, x0 ]])
```

LINEST_r2

LINEST_R2() restituisce il valore r^2 aggregato (coefficiente di determinazione) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_R2 (y-expression, x-expression [, y0 [, x0 ]])
```

LINEST_seb

LINEST_SEB() restituisce l'errore standard aggregato del valore b di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_SEB  (y-expression, x-expression [, y0 [, x0 ]])
```

LINEST_sem

LINEST_SEM() restituisce l'errore standard aggregato del valore m di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_SEM  (y-expression, x-expression [, y0 [, x0 ]])
```

LINEST_sey

LINEST_SEY() restituisce l'errore standard aggregato della stima del valore y di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_SEY  (y-expression, x-expression [, y0 [, x0 ]])
```

LINEST_ssreg

LINEST_SSREG() restituisce la somma di una regressione aggregata dei quadrati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_SSREG  (y-expression, x-expression [, y0 [, x0 ]])
```

Linest_ssresid

LINEST_SSRESID() restituisce la somma residua aggregata dei quadrati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
LINEST_SSRESID  (y-expression, x-expression [, y0 [, x0 ]])
```

Median

Median() restituisce la mediana aggregata dei valori nell'espressione su un insieme di record, come definito da una clausola **group by**.

```
Median (expression)
```

Skew

Skew() restituisce l'asimmetria dell'espressione su un insieme di record, come definito da una clausola **group by**.

```
Skew ([ distinct] expression)
```

Stdev

Stdev() restituisce la deviazione standard dei valori dati dall'espressione su un insieme di record, come definito da una clausola **group by**.

```
Stdev ([distinct] expression)
```

Sterr

Sterr() restituisce l'errore standard aggregato (stdev/sqrt(n)) per una serie di valori rappresentata da un'espressione ripetuta su un insieme di record, come definito da una clausola **group by**.

```
Sterr  ([distinct] expression)
```

STEYX

STEYX() restituisce l'errore standard aggregato del valore y previsto per ogni valore x nella regressione per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

```
STEYX  (y-expression, x-expression)
```

Funzioni di aggregazione statistica nelle espressioni grafiche

Le seguenti funzioni di aggregazione statistica possono essere utilizzate nei grafici.

Avg

Avg() restituisce la media aggregata dell'espressione o del campo ripetuto sulle dimensioni del grafico.

```
Avg - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld{,
fld}>]]} expr)
```

Correl

Correl() restituisce il coefficiente di correlazione aggregato per due serie di dati. La funzione di correlazione è una misura della relazione tra le serie di dati e viene aggregata per coppie di valori (x,y) ripetute sulle dimensioni del grafico.

```
Correl - funzione per grafici({[SetExpression] [TOTAL [<fld {, fld}>]]}
value1, value2 )
```

Fractile

Fractile() trova il valore che corrisponde al frattale inclusivo (quantile) dei dati aggregati nella scala data dall'espressione ripetuta sulle dimensioni del grafico.

```
Fractile - funzione per grafici({[SetExpression] [TOTAL [<fld {, fld}>]]}
expr, fraction)
```

FractileExc

FractileExc() trova il valore che corrisponde al frattale esclusivo (quantile) dei dati aggregati nella scala data dall'espressione ripetuta sulle dimensioni del grafico.

```
FractileExc - funzione per grafici({[SetExpression] [TOTAL [<fld {, fld}>]]}
expr, fraction)
```

Kurtosis

Kurtosis() trova il kurtosis della scala di dati aggregati nell'espressione o nel campo ripetuto sulle dimensioni del grafico.

```
Kurtosis - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld{,
fld}>]]} expr)
```

LINEST_b

LINEST_B() restituisce il valore b aggregato (intersezione con l'asse y) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri nelle espressioni date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_R2 - funzione per grafici({[SetExpression] [TOTAL [<fld{ ,fld}>]] }y_
value, x_value[, y0_const[, x0_const]])
```

LINEST_df

LINEST_DF() restituisce i gradi di libertà aggregato di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri nelle espressioni date da **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_DF - funzione per grafici({[SetExpression] [TOTAL [<fld{, fld}>]]} y_
value, x_value [, y0_const [, x0_const]])
```

LINEST_f

LINEST_F() restituisce la statistica F aggregata (r2/(1-r2)) di una regressione lineare definita dall'equazione y=mx+b di una serie di coordinate rappresentate da coppie di numeri nell'espressione data da **x_value** e da **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_F - funzione per grafici({[SetExpression] [TOTAL[<fld{, fld}>]]} y_
value, x_value [, y0_const [, x0_const]])
```

LINEST_m

LINEST_M() restituisce il valore m aggregato (pendenza) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_M - funzione per grafici({[SetExpression] [TOTAL[<fld{, fld}>]]} y_
value, x_value [, y0_const [, x0_const]])
```

LINEST_r2

LINEST_R2() restituisce il valore r2 aggregato (coefficiente di determinazione) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_R2 - funzione per grafici({[SetExpression] [TOTAL [<fld{ ,fld}>]] }y_
value, x_value[, y0_const[, x0_const]])
```

LINEST_seb

LINEST_SEB() restituisce l'errore standard aggregato del valore b di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri fornite dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_SEB - funzione per grafici({[SetExpression] [TOTAL [<fld{ ,fld}>]] }y_
value, x_value[, y0_const[, x0_const]])
```

LINEST_sem

LINEST_SEM() restituisce l'errore standard aggregato del valore m di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri fornite dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_SEM - funzione per grafici([{set_expression}][ distinct ] [total [<fld
{,fld}>] ] y-expression, x-expression [, y0 [, x0 ]] )
```

LINEST_sey

LINEST_SEY() restituisce l'errore standard aggregato della stima y di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_SEY - funzione per grafici({[SetExpression] [TOTAL [<fld{ ,fld}>]] }y_
value, x_value[, y0_const[, x0_const]])
```

LINEST_ssreg

LINEST_SSREG() restituisce la somma di regressione aggregata dei quadrati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_SSREG - funzione per grafici({[SetExpression] [TOTAL [<fld{ ,fld}>]]
}y_value, x_value[, y0_const[, x0_const]])
```

LINEST_ssresid

LINEST_SSRESID() restituisce la somma residua aggregata dei quadrati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri nelle espressioni fornite da **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

```
LINEST_SSRESID - funzione per graficiLINEST_SSRESID() restituisce la somma
residua aggregata dei quadrati di una regressione lineare definita
dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di
numeri nelle espressioni fornite da x_value e y_value, ripetute sulle
dimensioni del grafico.  LINEST_SSRESID([{SetExpression}] [DISTINCT] [TOTAL
```

[<fld{, fld}>]] y_value, x_value[, y0_const[, x0_const]])
numerico ArgomentiArgomentoDescrizioney_valueL'espressione o il campo contenente la scala di valori y da misurare.x_valueL'espressione o il campo contenente la scala di valori x da misurare.y0, x0È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati. SetExpressionPer impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis. DISTINCTSe la parola DISTINCT è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati. TOTALSe la parola TOTAL viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando TOTAL [<fld {.fld}>], dove il qualificatore TOTAL è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore TOTAL. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata Aggr in combinazione con una dimensione specificata. Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata. An example of how to use linest functionsavg({[SetExpression] [**TOTAL** [<fld{ ,fld}>]] }y_value, x_value[, y0_const[, x0_const]])**

Median

Median() restituisce il valore mediano della scala di valori aggregati nell'espressione ripetuti sulle dimensioni del grafico.

Median - funzione per grafici({[SetExpression] [**TOTAL** [<fld{, fld}>]]} expr**)**

MutualInfo

MutualInfo calcola le informazioni reciproche (MI, Mutual Information) tra due campi o tra valori aggregati in **Aggr()**.

MutualInfo - funzione per grafici (page 440){[SetExpression] [**DISTINCT**] [**TOTAL** target, driver [, datatype [, breakdownbyvalue [, samplesize]]]**)**

Skew

Skew() restituisce l'asimmetria aggregata dell'espressione o del campo ripetuta sulle dimensioni del grafico.

```
Skew - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld{
,fld}>]]} expr)
```

Stdev

Stdev() trova la deviazione standard della scala di dati aggregati nell'espressione o nel campo ripetuta sulle dimensioni del grafico.

```
Stdev - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld{,
fld}>]]} expr)
```

Sterr

Sterr() trova il valore dell'errore standard della media, (stdev/sqrt(n)), per la serie di valori aggregati nell'espressione ripetuta sulle dimensioni del grafico.

```
Sterr - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL[<fld{,
fld}>]]} expr)
```

STEYX

STEYX() restituisce l'errore standard aggregato quando si prevedono i valori y per ciascun valore x in una regressione lineare data da una serie di coordinate rappresentate da coppie di numeri nelle espressioni date da **y_value** e **x_value**.

```
STEYX - funzione per grafici{[SetExpression] [TOTAL [<fld{, fld}>]]} y_value,
x_value)
```

Avg

Avg() restituisce il valore medio dei dati aggregati nell'espressione su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
Avg([DISTINCT] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
DISTINCT	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato

`Temp:`
`crosstable (Month, Sales) load * inline [`
`Customer|Jan|Feb|Mar||Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec`
`Astrida|46|60|70|13|78|20|45|65|78|12|78|22`
`Betacab|65|56|22|79|12|56|45|24|32|78|55|15`
`Canutility|77|68|34|91|24|68|57|36|44|90|67|27`
`Divadip|36|44|90|67|27|57|68|47|90|80|94`
`] (delimiter is '|');`
 	
`Avg1:`
`LOAD Customer, Avg(Sales) as MyAverageSalesByCustomer`
`Resident Temp Group By Customer;`
	`Customer`
	`MyAverageSalesByCustomer`
	`Astrida 48.916667`
	`Betacab 44.916667`
	`Canutility 56.916667`
	`Divadip 63.083333`
	Questo può essere controllato nel foglio creando una tabella che includa la misura:
	`Sum(Sales)/12`

Presupponendo che la tabella **Temp** venga caricata come nell'esempio precedente:
`LOAD Customer,Avg(DISTINCT Sales) as MyAvgSalesDistinct`
`Resident Temp Group By Customer;`
	`Customer`
	`MyAverageSalesByCustomer`
	`Astrida 43.1`
	`Betacab 43.909091`
	`Canutility 55.909091`
	`Divadip 61`
	Vengono contati solo i valori distinti. Dividere il totale per il numero di valori non duplicati.

Avg - funzione per grafici

Avg() restituisce la media aggregata dell'espressione o del campo ripetuto sulle dimensioni del grafico.

Sintassi:

```
Avg([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.

Argomento	Descrizione
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Esempi e risultati:

Example table

Customer	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Astrida	46	60	70	13	78	20	45	65	78	12	78	22
Betacab	65	56	22	79	12	56	45	24	32	78	55	15
Canutility	77	68	34	91	24	68	57	36	44	90	67	27
Divadip	57	36	44	90	67	27	57	68	47	90	80	94

Esempi di funzioni

Esempio	Risultato
`Avg(Sales)`	Per una tabella che include la dimensione `Customer` e la misura `Avg([Sales])`, se i **Totali** sono visualizzati, il risultato è 2566.
`Avg([TOTAL (Sales))`	53,458333 per tutti i valori di `Customer`, perché il qualificatore TOTAL sta a indicare che le dimensioni vengono ignorate.
`Avg (DISTINCT (Sales))`	51,862069 per il totale perché l'utilizzo del qualificatore Distinct sta a indicare che vengono valutati solo i valori univoci in `Sales` per ogni `Customer`.

Dati utilizzati negli esempi:

```
Monthnames:
LOAD *, Dual(MonthText,MonthNumber) as Month INLINE [
MonthText, MonthNumber
Jan, 1
Feb, 2
Mar, 3
Apr, 4
May, 5
Jun, 6
Jul, 7
Aug, 8
Sep, 9
Oct, 10
Nov, 11
Dec, 12
];

Sales2013:
Crosstable (MonthText, Sales) LOAD * inline [
Customer|Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec
Astrida|46|60|70|13|78|20|45|65|78|12|78|22
Betacab|65|56|22|79|12|56|45|24|32|78|55|15
Canutility|77|68|34|91|24|68|57|36|44|90|67|27
Divadip|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');
```

Vedere anche:

p *Aggr - funzione per grafici (page 541)*

Correl

Correl() restituisce il coefficiente di correlazione aggregato per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record definito da una clausola **group by**.

Sintassi:

```
Correl(value1, value2)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value1, value2	Le espressioni o i campi contenenti i due gruppi campione per i quali deve essere misurato il coefficiente di correlazione.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato

`Salary:`
`Load *, 1 as Grp;`
`LOAD * inline [`
`"Employee`
`name"|Gender|Age|Salary`
`Aiden Charles|Male|20|25000`
`Brenda Davies|Male|25|32000`
`Charlotte`
`Edberg|Female|45|56000`
`Daroush Ferrara|Male|31|29000`
`Eunice Goldblum|Female|31|32000`
`Freddy Halvorsen|Male|25|26000`
`Gauri Indu|Female|36|46000`
`Harry Jones|Male|38|40000`
`Ian Underwood|Male|40|45000`
`Jackie Kingsley|Female|23|28000`
`] (delimiter is '|');`
 	
`Correl1:`
`LOAD Grp,`
`Correl(Age,Salary) as Correl_`
`Salary Resident Salary Group By`
`Grp;`
	In una tabella con dimensione `Correl_Salary`, verrà mostrato il risultato del calcolo Correl() nello script di caricamento dei dati: 0,9270611

Correl - funzione per grafici

Correl() restituisce il coefficiente di correlazione aggregato per due serie di dati. La funzione di correlazione è una misura della relazione tra le serie di dati e viene aggregata per coppie di valori (x,y) ripetute sulle dimensioni del grafico.

Sintassi:

```
Correl([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] value1, value2 )
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value1, value2	Le espressioni o i campi contenenti i due gruppi campione per i quali deve essere misurato il coefficiente di correlazione.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {,fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Esempi e risultati:

Esempi di funzioni

Esempio	Risultato
`Correl (Age, Salary)`	Per una tabella che include la dimensione `Employee name` e la misura `Correl(Age, Salary)`, il risultato è 0,9270611. Il risultato viene visualizzato solo per la cella dei totali.

Esempio	Risultato
`Correl (TOTAL Age, Salary))`	0.927. Questo risultato e quelli successivi sono visualizzati con tre posizioni decimali per favorire la leggibilità. Se si crea una casella di filtro con la dimensione Gender e si eseguono selezioni da questa, si otterrà il risultato 0,951 quando viene selezionato il valore Female e 0,939 quando viene selezionato il valore Male. Ciò si verifica perché la selezione esclude tutti i risultati che non appartengono all'altro valore di Gender.
`Correl({1} TOTAL Age, Salary))`	0.927. Indipendente dalle selezioni. Ciò si verifica perché l'espressione di gruppo {1} ignora tutte le selezioni e le dimensioni.
`Correl (TOTAL <Gender> Age, Salary))`	0,927 nella cella del totale, 0,939 per tutti i valori di Male e 0,951 per tutti i valori di Female. Questo corrisponde ai risultati ottenuti eseguendo selezioni in una casella di filtro in base a Gender.

Dati utilizzati negli esempi:

```
Salary:
LOAD * inline [
"Employee name"|Gender|Age|Salary
Aiden Charles|Male|20|25000
Brenda Davies|Male|25|32000
Charlotte Edberg|Female|45|56000
Daroush Ferrara|Male|31|29000
Eunice Goldblum|Female|31|32000
Freddy Halvorsen|Male|25|26000
Gauri Indu|Female|36|46000
Harry Jones|Male|38|40000
Ian Underwood|Male|40|45000
Jackie Kingsley|Female|23|28000
] (delimiter is '|');
```

Vedere anche:

p *Aggr - funzione per grafici (page 541)*
p *Avg - funzione per grafici (page 400)*
p *RangeCorrel (page 1326)*

Fractile

Fractile() restituisce il valore corrispondente al frattale inclusivo (quantile) dei dati aggregati nell'espressione su un insieme di record, come definito da una clausola **group by**.

È possibile utilizzare FractileExc (page 409) per calcolare il frattale esclusivo.

Sintassi:

```
Fractile(expr, fraction)
```

Tipo di dati restituiti: numerico

La funzione restituisce il valore corrispondente alla classificazione come definito da `classificazione= frazione * (N-1) + 1` in cui `N` è il numero dei valori in `expr`. Se `classificazione` è un numero non intero, viene effettuata un'interpolazione tra i due valori più vicini.

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o campo contenente i dati da utilizzare durante il calcolo del frattale.
fraction	Un numero compreso tra 0 e 1 corrispondente al frattale (quantile espresso come frazione) da calcolare.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato
Table1: crosstable LOAD recno() as ID, * inline [Observation|Comparison 35|2 40|27 12|38 15|31 21|1 14|19 46|1 10|34 28|3 48|1 16|2 30|3 32|2 48|1 31|2 22|1 12|3 39|29 19|37 25|2] (delimiter is '|'); Fractile1: LOAD Type, Fractile(Value,0.75) as MyFractile Resident Table1 Group By Type;	In una tabella con le dimensioni `Type` e `MyFractile`, i risultati dei calcoli Fractile() nello script di caricamento dei dati sono: Type MyFractile Comparison 27.5 Observation 36

Fractile - funzione per grafici

Fractile() trova il valore che corrisponde al frattale inclusivo (quantile) dei dati aggregati nella scala data dall'espressione ripetuta sulle dimensioni del grafico.

È possibile utilizzare FractileExc - funzione per grafici (page 411) per calcolare il frattale esclusivo.

Sintassi:

```
Fractile([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] expr, fraction)
```

Tipo di dati restituiti: numerico

La funzione restituisce il valore corrispondente alla classificazione come definito da `classificazione= frazione * (N-1) + 1` in cui `N` è il numero dei valori in `expr`. Se `classificazione` è un numero non intero, viene effettuata un'interpolazione tra i due valori più vicini.

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o campo contenente i dati da utilizzare durante il calcolo del frattale.
fraction	Un numero compreso tra 0 e 1 corrispondente al frattale (quantile espresso come frazione) da calcolare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Esempi e risultati:

Example table

Customer	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Astrida	46	60	70	13	78	20	45	65	78	12	78	22
Betacab	65	56	22	79	12	56	45	24	32	78	55	15
Canutility	77	68	34	91	24	68	57	36	44	90	67	27
Divadip	57	36	44	90	67	27	57	68	47	90	80	94

Esempi di funzioni

Esempio	Risultato
`Fractile (Sales, 0.75)`	Per una tabella che include la dimensione `Customer` e la misura `Fractile([Sales])`, se i **Totali** sono visualizzati, il risultato è 71,75. Questo è il punto nella distribuzione dei valori di `Sales` sotto al quale ricade il 75% dei valori.
`Fractile (TOTAL Sales, 0.75))`	71,75 per tutti i valori di `Customer`, perché il qualificatore TOTAL sta a indicare che le dimensioni vengono ignorate.
`Fractile (DISTINCT Sales, 0.75)`	70 per il totale perché l'utilizzo del qualificatore DISTINCT sta a indicare che vengono valutati solo i valori univoci in `Sales` per ogni `Customer`.

Dati utilizzati negli esempi:

```
Monthnames:
LOAD *, Dual(MonthText,MonthNumber) as Month INLINE [
MonthText, MonthNumber
Jan, 1
Feb, 2
Mar, 3
Apr, 4
May, 5
Jun, 6
Jul, 7
Aug, 8
Sep, 9
Oct, 10
Nov, 11
Dec, 12
];

Sales2013:
Crosstable (MonthText, Sales) LOAD * inline [
Customer|Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec
Astrida|46|60|70|13|78|20|45|65|78|12|78|22
Betacab|65|56|22|79|12|56|45|24|32|78|55|15
Canutility|77|68|34|91|24|68|57|36|44|90|67|27
Divadip|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');
```

Vedere anche:

p *Aggr - funzione per grafici (page 541)*

FractileExc

FractileExc() restituisce il valore corrispondente al frattale esclusivo (quantile) dei dati aggregati nell'espressione su un insieme di record, come definito da una clausola **group by**.

È possibile utilizzare Fractile (page 405) per calcolare il frattale inclusivo.

Sintassi:

```
FractileExc(expr, fraction)
```

Tipo di dati restituiti: numerico

La funzione restituisce il valore corrispondente alla classificazione come definito da `classificazione = frazione * (N+1)` in cui `N` è il numero dei valori in `expr`. Se `classificazione` è un numero non intero, viene effettuata un'interpolazione tra i due valori più vicini.

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o campo contenente i dati da utilizzare durante il calcolo del frattale.
fraction	Un numero compreso tra 0 e 1 corrispondente al frattale (quantile espresso come frazione) da calcolare.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato
Table1: crosstable LOAD recno() as ID, * inline [Observation|Comparison 35|2 40|27 12|38 15|31 21|1 14|19 46|1 10|34 28|3 48|1 16|2 30|3 32|2 48|1 31|2 22|1 12|3 39|29 19|37 25|2] (delimiter is '|'); Fractile1: LOAD Type, FractileExc(Value,0.75) as MyFractile Resident Table1 Group By Type;	In una tabella con le dimensioni `Type` e `MyFractile`, i risultati dei calcoli FractileExc() nello script di caricamento dei dati sono: Type MyFractile Comparison 28.5 Observation 38

FractileExc - funzione per grafici

FractileExc() trova il valore che corrisponde al frattale esclusivo (quantile) dei dati aggregati nella scala data dall'espressione ripetuta sulle dimensioni del grafico.

È possibile utilizzare Fractile - funzione per grafici (page 407) per calcolare il frattale inclusivo.

Sintassi:

```
FractileExc([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] expr,
fraction)
```

Tipo di dati restituiti: numerico

La funzione restituisce il valore corrispondente alla classificazione come definito da `classificazione = frazione * (N+1)` in cui `N` è il numero dei valori in `expr`. Se `classificazione` è un numero non intero, viene effettuata un'interpolazione tra i due valori più vicini.

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o campo contenente i dati da utilizzare durante il calcolo del frattale.
fraction	Un numero compreso tra 0 e 1 corrispondente al frattale (quantile espresso come frazione) da calcolare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Esempi e risultati:

Example table

Customer	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Astrida	46	60	70	13	78	20	45	65	78	12	78	22
Betacab	65	56	22	79	12	56	45	24	32	78	55	15
Canutility	77	68	34	91	24	68	57	36	44	90	67	27
Divadip	57	36	44	90	67	27	57	68	47	90	80	94

Esempi di funzioni

Esempio	Risultato
`FractileExc (Sales, 0.75)`	Per una tabella che include la dimensione `Customer` e la misura `FractileExc ([Sales])`, se i **Totali** sono visualizzati, il risultato è 75,25. Questo è il punto nella distribuzione dei valori di `Sales` sotto al quale ricade il 75% dei valori.
`FractileExc (TOTAL Sales, 0.75))`	75,25 per tutti i valori di `Customer`, perché il qualificatore TOTAL sta a indicare che le dimensioni vengono ignorate.
`FractileExc (DISTINCT Sales, 0.75)`	73,50 per il totale perché l'utilizzo del qualificatore DISTINCT sta a indicare che vengono valutati solo i valori univoci in `Sales` per ogni `Customer`.

Dati utilizzati negli esempi:

```
Monthnames:
LOAD *, Dual(MonthText,MonthNumber) as Month INLINE [
MonthText, MonthNumber
Jan, 1
Feb, 2
Mar, 3
Apr, 4
May, 5
Jun, 6
Jul, 7
Aug, 8
Sep, 9
Oct, 10
Nov, 11
Dec, 12
];

Sales2013:
Crosstable (MonthText, Sales) LOAD * inline [
Customer|Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec
Astrida|46|60|70|13|78|20|45|65|78|12|78|22
Betacab|65|56|22|79|12|56|45|24|32|78|55|15
Canutility|77|68|34|91|24|68|57|36|44|90|67|27
Divadip|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');
```

Vedere anche:

p *Aggr - funzione per grafici (page 541)*

Kurtosis

Kurtosis() restituisce il kurtosis dei dati nell'espressione su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
Kurtosis([distinct ] expr )
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato
Table1: crosstable LOAD recno() as ID, * inline [Observation|Comparison 35|2 40|27 12|38 15|31 21|1 14|19 46|1 10|34 28|3 48|1 16|2 30|3 32|2 48|1 31|2 22|1 12|3 39|29 19|37 25|2] (delimiter is '|'); Kurtosis1: LOAD Type, Kurtosis(Value) as MyKurtosis1, Kurtosis(DISTINCT Value) as MyKurtosis2 Resident Table1 Group By Type;	In una tabella con dimensioni Type, MyKurtosis1 e MyKurtosis2, i risultati dei calcoli Kurtosis() nello script di caricamento dei dati sono: Type MyKurtosis1 MyKurtosis2 Comparison -1.1612957 -1.4982366 Observation -1.1148768 -0.93540144

Kurtosis - funzione per grafici

Kurtosis() trova il kurtosis della scala di dati aggregati nell'espressione o nel campo ripetuto sulle dimensioni del grafico.

Sintassi:

```
Kurtosis([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Esempi e risultati:

Example table

Type	Value																			
Comparison	2	27	38	31	1	19	1	34	3	1	2	3	2	1	2	1	3	29	37	2
Observation	35	40	12	15	21	14	46	10	28	48	16	30	32	48	31	22	12	39	19	25

Esempi di funzioni

Esempio	Risultato
`Kurtosis (Value)`	Per una tabella che include la dimensione `Type` e la misura `Kurtosis(Value)`, se vengono visualizzati i **Totali** per la tabella e la formattazione del numero è impostata su 3 cifre significative, il risultato è 1,252. Per `Comparison` è 1,161 e per `Observation` è 1,115.
`Kurtosis (TOTAL Value))`	1,252 per tutti i valori di `Type`, perché il qualificatore TOTAL sta a indicare che le dimensioni vengono ignorate.

Dati utilizzati negli esempi:

```
Table1:
crosstable LOAD recno() as ID, * inline [
Observation|Comparison
35|2
40|27
12|38
15|31
21|1
14|19
46|1
10|34
28|3
48|1
16|2
30|3
32|2
48|1
31|2
22|1
12|3
39|29
19|37
25|2 ] (delimiter is '|');
```

Vedere anche:

p *Avg - funzione per grafici (page 400)*

LINEST_B

LINEST_B() restituisce il valore b aggregato (intercettazione sull'asse y) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
LINEST_B (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_B - funzione per grafici

LINEST_B() restituisce il valore b aggregato (intersezione con l'asse y) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri nelle espressioni date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_B([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_value
[, y0_const [ , x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.

Argomento	Descrizione
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0_const, x0_const	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*
p *Avg - funzione per grafici (page 400)*

LINEST_DF

LINEST_DF() restituisce i gradi di libertà aggregati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola

group by.

Sintassi:

```
LINEST_DF (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_DF - funzione per grafici

LINEST_DF() restituisce i gradi di libertà aggregato di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri nelle espressioni date da **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_DF([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_
value [, y0_const [, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

p *Avg - funzione per grafici (page 400)*

LINEST_F

Questa funzione di script restituisce la statistica F aggregata ($r^2/(1-r^2)$) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record definito da una clausola **group by**.

Sintassi:

```
LINEST_F (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_F - funzione per grafici

LINEST_F() restituisce la statistica F aggregata (r2/(1-r2)) di una regressione lineare definita dall'equazione y=mx+b di una serie di coordinate rappresentate da coppie di numeri nell'espressione data da **x_value** e da **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_F([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_value
[, y0_const [, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

p *Avg - funzione per grafici (page 400)*

LINEST_M

LINEST_M() restituisce il valore m aggregato (pendenza) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
LINEST_M (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_M - funzione per grafici

LINEST_M() restituisce il valore m aggregato (pendenza) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_M([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_value
[, y0_const [, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

p *Avg - funzione per grafici (page 400)*

LINEST_R2

LINEST_R2() restituisce il valore r^2 aggregato (coefficiente di determinazione) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
LINEST_R2 (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_R2 - funzione per grafici

LINEST_R2() restituisce il valore r2 aggregato (coefficiente di determinazione) di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_R2([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_
value[, y0_const[, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

p *Avg - funzione per grafici (page 400)*

LINEST_SEB

LINEST_SEB() restituisce l'errore standard aggregato del valore b di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
LINEST_SEB (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_SEB - funzione per grafici

LINEST_SEB() restituisce l'errore standard aggregato del valore b di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri fornite dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_SEB([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_
value[, y0_const[, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

p *Avg - funzione per grafici (page 400)*

LINEST_SEM

LINEST_SEM() restituisce l'errore standard aggregato del valore m di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
LINEST_SEM (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_SEM - funzione per grafici

LINEST_SEM() restituisce l'errore standard aggregato del valore m di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri fornite dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_SEM([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_
value[, y0_const[, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

p *Avg - funzione per grafici (page 400)*

LINEST_SEY

LINEST_SEY() restituisce l'errore standard aggregato della stima del valore y di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
LINEST_SEY (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_SEY - funzione per grafici

LINEST_SEY() restituisce l'errore standard aggregato della stima y di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_SEY([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_
value[, y0_const[, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {,fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

p *Avg - funzione per grafici (page 400)*

LINEST_SSREG

LINEST_SSREG() restituisce la somma di una regressione aggregata dei quadrati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
LINEST_SSREG (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_SSREG - funzione per grafici

LINEST_SSREG() restituisce la somma di regressione aggregata dei quadrati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri date dalle espressioni **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_SSREG([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_
value[, y0_const[, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

p *Avg - funzione per grafici (page 400)*

LINEST_SSRESID

LINEST_SSRESID() restituisce la somma residua aggregata dei quadrati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
LINEST_SSRESID (y_value, x_value[, y0 [, x0 ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y(0), x(0)	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa. A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*

LINEST_SSRESID - funzione per grafici

LINEST_SSRESID() restituisce la somma residua aggregata dei quadrati di una regressione lineare definita dall'equazione y=mx+b per una serie di coordinate rappresentate da coppie di numeri nelle espressioni fornite da **x_value** e **y_value**, ripetute sulle dimensioni del grafico.

Sintassi:

```
LINEST_SSRESID([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value,
x_value[, y0_const[, x0_const]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.
y0, x0	È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.  *A meno che non vengano dichiarati sia y0 che x0, la funzione richiede almeno due coppie di dati valide per il calcolo. Se vengono dichiarati i valori y0 e x0, sarà sufficiente una singola coppia di dati.*
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {,fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

È possibile dichiarare un valore opzionale y0 forzando il passaggio della linea di regressione attraverso l'asse delle y in un determinato punto. Dichiarando sia y0 che x0, è possibile forzare il passaggio della linea di regressione attraverso una coordinata singola fissa.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Vedere anche:

p *Esempi di utilizzo delle funzioni linest (page 456)*
p *Avg - funzione per grafici (page 400)*

Median

Median() restituisce la mediana aggregata dei valori nell'espressione su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
Median (expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.

Esempio: Espressione script mediante Median

Esempio - espressione script

Script di caricamento

Caricare i seguenti dati inline e l'espressione di script nell'editor caricamento dati per questo esempio.

```
Table 1: Load RecNo() as RowNo, Letter, Number Inline [Letter, Number A,1 A,3 A,4 A,9 B,2 B,8
B,9];                                  Median: LOAD Letter, Median(Number) as MyMedian Resident Table1 Grou
```

Creazione di una visualizzazione

Creare una visualizzazione tabella in un foglio Qlik Sense con **Lettera** e **MyMedian** come dimensioni.

Risultato

Letter	MyMedian
A	3.5
B	8

Spiegazione

La mediana è considerata il numero "medio" quando i numeri sono stati ordinati dal più piccolo al più grande. Se la serie di dati presenta un numero pari di valori, la funzione restituisce la media dei due valori centrali. In questo esempio, la mediana è calcolata per ogni serie di valori di **A** e **B**, che è 3,5 e 8, rispettivamente.

Median - funzione per grafici

Median() restituisce il valore mediano della scala di valori aggregati nell'espressione ripetuti sulle dimensioni del grafico.

Sintassi:

```
Median([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Esempio: Espressione del grafico mediante la mediana

Esempio - Espressione del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare l'esempio di espressione del grafico in basso.

```
Load RecNo() as RowNo, Letter, Number Inline [Letter, Number A,1 A,3 A,4 A,9 B,2 B,8 B,9];
```

Creazione di una visualizzazione

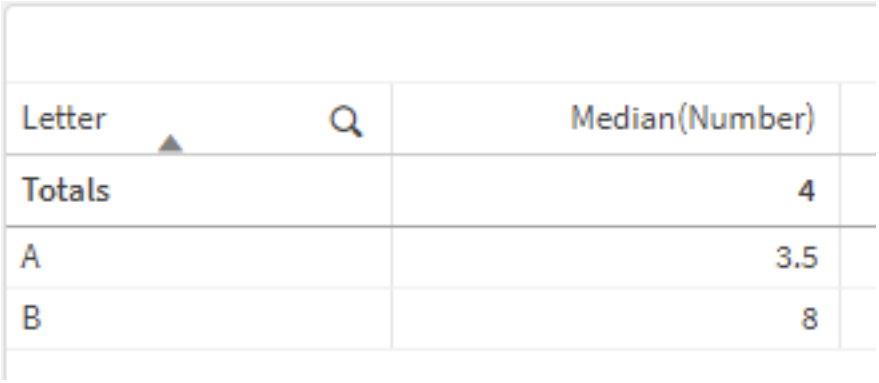
Creare una visualizzazione tabella in un foglio Qlik Sense con **Lettera** come dimensione.

Espressione del grafico

Aggiungere la seguente espressione alla tabella come misura:

```
Median(Number)
```

Risultato



Letter	Median(Number)
Totals	4
A	3.5
B	8

Spiegazione

La mediana è considerata il numero "medio" quando i numeri sono stati ordinati dal più piccolo al più grande. Se la serie di dati presenta un numero pari di valori, la funzione restituisce la media dei due valori centrali. In questo esempio, la mediana è calcolata per ogni serie di valori di **A** e **B**, che è 3,5 e 8, rispettivamente.

La mediana per i **Totali** è calcolata da tutti i valori, che equivale a 4.

Vedere anche:

p *Avg - funzione per grafici (page 400)*

MutualInfo - funzione per grafici

MutualInfo calcola le informazioni reciproche (MI, Mutual Information) tra due campi o tra valori aggregati in **Aggr()**.

MutualInfo restituisce le informazioni reciproche aggregate per due set di dati. Ciò consente un'analisi dei driver chiave tra un campo e un driver potenziale. Le informazioni reciproche misurano il rapporto tra i set di dati e vengono aggregate per i valori di coppia (x,y) ripetuti sulle dimensioni dei grafici. Le informazioni reciproche vengono misurate tra 0 e 1 e possono essere formattate come valore percentile. **MutualInfo** viene definito mediante selezioni o un'espressione set.

MutualInfo consente tipi diversi di analisi MI:

- MI pairwise: Calcola il valore MI tra un campo driver e un campo target.
- Scomposizione driver per valore: il valore MI viene calcolato tra i singoli valori di campo nei campi driver e target.
- Selezione funzionalità: Utilizzare **MutualInfo** in un grafico a griglia per creare una matrice in cui tutti i campi vengono confrontati l'uno con l'altro in base alle informazioni reciproche (MI).

MutualInfo non indica necessariamente casualità tra i campi che condividono informazioni reciproche. Due campi possono condividere informazioni reciproche, ma potrebbero non essere driver reciprocamente uguali. Ad esempio, al momento di confrontare le vendite di gelati e la temperatura esterna, **MutualInfo** mostrerà le informazioni reciproche tra i due. Non indicherà se è la temperatura esterna a spingere le vendite di gelati, che rappresenta una motivazione probabile, o se sono le vendite dei gelati a spingere la temperatura esterna, ovvero una motivazione piuttosto improbabile.

Quando si calcolano le informazioni reciproche, le associazioni influiscono sulla corrispondenza tra e la frequenza dei valori dai campi che provengono da tabelle diverse.

I valori restituiti per gli stessi campi o selezioni possono variare leggermente. Ciò è dovuto al fatto che ciascun richiamo **MutualInfo** opera in base a un campione selezionato in modo casuale e alla casualità intrinseca dell'algoritmo **MutualInfo**.

MutualInfo può essere applicato alla funzione **Aggr()**.

Sintassi:

```
MutualInfo({SetExpression}] [DISTINCT] [TOTAL] field1, field2 , datatype [,
breakdownbyvalue [, samplesize ]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
field1, field2	Le espressioni o i campi contenenti i due set di campioni per i quali vengono misurate le informazioni mutue.
datatype	I tipi di dati contenuti nel target e nel driver, 1 o 'dd' per discrete:discrete 2 o 'cc' per continuous:continuous 3 o 'cd' per continuous:discrete 4 o 'dc' per discrete:continuous I tipi di dati non fanno distinzioni tra maiuscole e minuscole.
breakdownbyvalue	Un valore statico corrispondente a un valore nel driver. Se fornito, il calcolo calcolerà il contributo MI per tale valore. È possibile utilizzare **ValueList()** o **ValueLoop()**. Se viene aggiunto **Null()**, il calcolo calcolerà il valore MI complessivo per tutti i valori nel driver. La scomposizione per valore richiede che il driver contenga dati discreti.

Argomento	Descrizione
samplesize	Il numero di valori da campionare dal target e dal driver. Il campionamento è casuale. **MutualInfo** richiede una dimensione di campionamento minima di 80. Per impostazione predefinita, **MutualInfo** campiona solo fino a 10.000 coppie di dati, dato che **MutualInfo** può presentare un grosso impatto sulle risorse. È possibile specificare un numero maggiore di coppie dati nelle dimensioni del campione. In caso di esaurimento del tempo a disposizione per **MutualInfo**, ridurre le dimensioni del campione.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Esempi di funzioni

Esempio	Risultato
`mutualinfo (Age, Salary, 1)`	Per un tabella che include la dimensione `Employee name` e la misura `mutualinfo(Age, Salary, 1)`, il risultato è 0.99820986. Il risultato viene visualizzato solo per la cella dei totali.
`mutualinfo (TOTAL Age, Salary, 1, null(), 81)`	Se si crea una casella di filtro con la dimensione Gender e si eseguono selezioni da questa, si otterrà il risultato 0.99805677 quando viene selezionato il valore Female e 0.99847373 quando viene selezionato il valore Male. Ciò si verifica perché la selezione esclude tutti i risultati che non appartengono all'altro valore di Gender.

Esempio	Risultato
`mutualinfo (TOTAL Age, Gender, 1, ValueLoop (25,35))`	0.68196996. La selezione di qualsiasi valore da Gender porterà il risultato a 0.
`mutualinfo ({1} TOTAL Age, Salary, 1, null())`	0.99820986. Ciò risulta indipendente dalle selezioni. L'espressione set {1} ignora tutte le selezioni e le dimensioni.

Dati utilizzati negli esempi:

```
Salary:
LOAD * inline [
"Employee name"|Age|Gender|Salary
Aiden Charles|20|Male|25000
Ann Lindquist|69|Female|58000
Anna Johansen|37|Female|36000
Anna Karlsson|42|Female|23000
Antonio Garcia|20|Male|61000
Benjamin Smith|42|Male|27000
Bill Yang|49|Male|50000
Binh Protzmann|69|Male|21000
Bob Park|51|Male|54000
Brenda Davies|25|Male|32000
Celine Gagnon|48|Female|38000
Cezar Sandu|50|Male|46000
Charles Ingvar Jönsson|27|Male|58000
Charlotte Edberg|45|Female|56000
Cindy Lynn|69|Female|28000
Clark Wayne|63|Male|31000
Daroush Ferrara|31|Male|29000
David Cooper|37|Male|64000
David Leg|58|Male|57000
Eunice Goldblum|31|Female|32000
Freddy Halvorsen|25|Male|26000
Gauri Indu|36|Female|46000
George van Zaant|59|Male|47000
Glenn Brown|58|Male|40000
Harry Jones|38|Male|40000
Helen Brolin|52|Female|66000
Hiroshi Ito|24|Male|42000
Ian Underwood|40|Male|45000
Ingrid Hendrix|63|Female|27000
Ira Baumel|39|Female|39000
Jackie Kingsley|23|Female|28000
Jennica Williams|36|Female|48000
Jerry Tessel|31|Male|57000
Jim Bond|50|Male|58000
Joan Callins|60|Female|65000
Joan Cleaves|25|Female|61000
Joe Cheng|61|Male|41000
John Doe|36|Male|59000
John Lemon|43|Male|21000
```

```
Karen Helmkey|54|Female|25000
Karl Berger|38|Male|68000
Karl Straubaum|30|Male|40000
Kaya Alpan|32|Female|60000
Kenneth Finley|21|Male|25000
Leif Shine|63|Male|70000
Lennart Skoglund|63|Male|24000
Leona Korhonen|46|Female|50000
Lina André|50|Female|65000
Louis Presley|29|Male|36000
Luke Langston|50|Male|63000
Marcus Salvatori|31|Male|46000
Marie Simon|57|Female|23000
Mario Rossi|39|Male|62000
Markus Danzig|26|Male|48000
Michael Carlen|21|Male|45000
Michelle Tyson|44|Female|69000
Mike Ashkenaz|45|Male|68000
Miro Ito|40|Male|39000
Nina Mihn|62|Female|57000
Olivia Nguyen|35|Female|51000
Olivier Simenon|44|Male|31000
Östen Ärlig|68|Male|57000
Pamala Garcia|69|Female|29000
Paolo Romano|34|Male|45000
Pat Taylor|67|Female|69000
Paul Dupont|34|Male|38000
Peter Smith|56|Male|53000
Pierre Clouseau|21|Male|37000
Preben Jørgensen|35|Male|38000
Rey Jones|65|Female|20000
Ricardo Gucci|55|Male|65000
Richard Ranieri|30|Male|64000
Rob Carsson|46|Male|54000
Rolf Wesenlund|25|Male|51000
Ronaldo Costa|64|Male|39000
Sabrina Richards|57|Female|40000
Sato Hiromu|35|Male|21000
Sehoon Daw|57|Male|24000
Stefan Lind|67|Male|35000
Steve Cioazzi|58|Male|23000
Sunil Gupta|45|Male|40000
Sven Svensson|45|Male|55000
Tom Lindwall|46|Male|24000
Tomas Nilsson|27|Male|22000
Trinity Rizzo|52|Female|48000
Vanessa Lambert|54|Female|27000
] (delimiter is '|');
```

Skew

Skew() restituisce l'asimmetria dell'espressione su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
Skew([ distinct] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
DISTINCT	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Creare quindi una tabella lineare utilizzando `Type` e *MySkew* come dimensioni.

Dati risultanti

Esempio	Risultato

`Table1:`
`crosstable LOAD recno() as ID, * inline [`
`Observation|Comparison`
`35|2`
`40|27`
`12|38`
`15|31`
`21|1`
`14|19`
`46|1`
`10|34`
`28|3`
`48|1`
`16|2`
`30|3`
`32|2`
`48|1`
`31|2`
`22|1`
`12|3`
`39|29`
`19|37`
`25|2 ] (delimiter is '|');`
 	
`Skew1:`
`LOAD Type,`
`Skew(Value) as MySkew`
`Resident Table1 Group By Type;`
	I risultati del calcolo di Skew() sono:
	• `Type` è `MySkew`
	• `Comparison` è 0.86414768
	• `Observation` è 0.32625351

Skew - funzione per grafici

Skew() restituisce l'asimmetria aggregata dell'espressione o del campo ripetuta sulle dimensioni del grafico.

Sintassi:

```
Skew([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Creare quindi una tabella lineare utilizzando `Type` come dimensione e `Skew(Value)` come misura.

Si consiglia di abilitare `Totals` nelle proprietà della tabella.

Esempio	Risultato
Table1: crosstable LOAD recno() as ID, * inline [Observation|Comparison 35|2 40|27 12|38 15|31 21|1 14|19 46|1 10|34 28|3 48|1 16|2 30|3 32|2 48|1 31|2 22|1 12|3 39|29 19|37 25|2] (delimiter is '|');	I risultati del calcolo di Skew(Value) sono: • Total è 0.23522195 • Comparison è 0.86414768 • Observation è 0.32625351

Vedere anche:

p *Avg - funzione per grafici (page 400)*

Stdev

Stdev() restituisce la deviazione standard dei valori dati dall'espressione su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
Stdev([distinct] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Creare quindi una tabella lineare utilizzando `Type` e `MyStdev` come dimensioni.

Dati risultanti

Esempio	Risultato

`Table1:`
`crosstable LOAD recno() as ID, * inline [`
`Observation|Comparison`
`35|2`
`40|27`
`12|38`
`15|31`
`21|1`
`14|19`
`46|1`
`10|34`
`28|3`
`48|1`
`16|2`
`30|3`
`32|2`
`48|1`
`31|2`
`22|1`
`12|3`
`39|29`
`19|37`
`25|2 ] (delimiter is '|');`
 	
`Stdev1:`
`LOAD Type,`
`Stdev(Value) as MyStdev`
`Resident Table1 Group By Type;`
	I risultati del calcolo di Stdev() sono:
	• `Type` è `MyStdev`
	• `Comparison` è 14.61245
	• `Observation` è 12.507997

Stdev - funzione per grafici

Stdev() trova la deviazione standard della scala di dati aggregati nell'espressione o nel campo ripetuta sulle dimensioni del grafico.

Sintassi:

```
Stdev([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Creare quindi una tabella lineare utilizzando `Type` come dimensione e `Stdev(Value)` come misura.

Si consiglia di abilitare `Totals` nelle proprietà della tabella.

Esempio	Risultato

`Stdev(Value)`
`Table1:`
`crosstable LOAD recno()`
`as ID, * inline [`
`Observation|Comparison`
`35|2`
`40|27`
`12|38`
`15|31`
`21|1`
`14|19`
`46|1`
`10|34`
`28|3`
`48|1`
`16|2`
`30|3`
`32|2`
`48|1`
`31|2`
`22|1`
`12|3`
`39|29`
`19|37`
`25|2 ] (delimiter is`
`'|');`
	I risultati del calcolo di Stdev(Value) sono:
	• `Total` è 15.47529
	• `Comparison` è 14.61245
	• `Observation` è 12.507997

Vedere anche:

p *Avg - funzione per grafici (page 400)*
p *STEYX - funzione per grafici (page 454)*

Sterr

Sterr() restituisce l'errore standard aggregato (stdev/sqrt(n)) per una serie di valori rappresentata da un'espressione ripetuta su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
Sterr ([distinct] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Limiti:

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato

`Table1:`
`crosstable LOAD recno()`
`as ID, * inline [`
`Observation|Comparison`
`35|2`
`40|27`
`12|38`
`15|31`
`21|1`
`14|19`
`46|1`
`10|34`
`28|3`
`48|1`
`16|2`
`30|3`
`32|2`
`48|1`
`31|2`
`22|1`
`12|3`
`39|29`
`19|37`
`25|2 ] (delimiter is`
`'|');`
 	
`Sterr1:`
`LOAD Type,`
`Sterr(Value) as MySterr`
`Resident Table1 Group By`
`Type;`
	In una tabella con le dimensioni `Type` e `MySterr`, i risultati del calcolo di Sterr() nello script di caricamento dei dati sono:
	
	`Type MySterr`
	`Comparison 3.2674431`
	`Observation 2.7968733`

Sterr - funzione per grafici

Sterr() trova il valore dell'errore standard della media, (stdev/sqrt(n)), per la serie di valori aggregati nell'espressione ripetuta sulle dimensioni del grafico.

Sintassi:

```
Sterr([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] expr)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Creare quindi una tabella lineare utilizzando `Type` come dimensione e `Sterr(Value)` come misura.

Si consiglia di abilitare `Totals` nelle proprietà della tabella.

Esempio	Risultato
Table1: crosstable LOAD recno() as ID, * inline [Observation|Comparison 35|2 40|27 12|38 15|31 21|1 14|19 46|1 10|34 28|3 48|1 16|2 30|3 32|2 48|1 31|2 22|1 12|3 39|29 19|37 25|2] (delimiter is '|');	I risultati del calcolo di Sterr(Value) sono: • Total è 2.4468583 • Comparison è 3.2674431 • Observation è 2.7968733

Vedere anche:

p *Avg - funzione per grafici (page 400)*
p *STEYX - funzione per grafici (page 454)*

STEYX

STEYX() restituisce l'errore standard aggregato del valore y previsto per ogni valore x nella regressione per una serie di coordinate rappresentata da coppie di numeri in x-expression e y-expression ripetute su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
STEYX (y_value, x_value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y da misurare.
x_value	L'espressione o il campo contenente la scala di valori x da misurare.

Limiti:

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato

`Trend:`
`Load *, 1 as Grp;`
`LOAD * inline [`
`Month|KnownY|KnownX`
`Jan|2|6`
`Feb|3|5`
`Mar|9|11`
`Apr|6|7`
`May|8|5`
`Jun|7|4`
`Jul|5|5`
`Aug|10|8`
`Sep|9|10`
`Oct|12|14`
`Nov|15|17`
`Dec|14|16`
`] (delimiter is '|');`
 	
`STEYX1:`
`LOAD Grp,`
`STEYX(KnownY, KnownX)`
`as MySTEYX`
`Resident Trend Group`
`By Grp;`
	In una tabella con la dimensione `MySTEYX`, il risultato del calcolo di STEYX() nello script di caricamento dei dati è 2,0714764.

STEYX - funzione per grafici

STEYX() restituisce l'errore standard aggregato quando si prevedono i valori y per ciascun valore x in una regressione lineare data da una serie di coordinate rappresentate da coppie di numeri nelle espressioni date da **y_value** e **x_value**.

Sintassi:

```
STEYX([{SetExpression}] [DISTINCT] [TOTAL [<fld{, fld}>]] y_value, x_value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
y_value	L'espressione o il campo contenente la scala di valori y conosciuti da misurare.
x_value	L'espressione o il campo contenente la scala di valori x conosciuti da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {,fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Il parametro della funzione di aggregazione non deve contenere altre funzioni di aggregazione, a meno che tali aggregazioni interne non contengano il qualificatore **TOTAL**. Nel caso di aggregazioni nidificate più complesse, utilizzare la funzione avanzata **Aggr** in combinazione con una dimensione specificata.

Se una o entrambe le parti di una coppia di dati includono valori di testo, valori NULL e valori mancanti, l'intera coppia di dati verrà ignorata.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Creare quindi una tabella lineare utilizzando `KnownY` e `KnownX` come dimensione e `Steyx(KnownY,KnownX)` come misura.

Si consiglia di abilitare `Totals` nelle proprietà della tabella.

Esempio	Risultato
Trend: LOAD * inline [Month|KnownY|KnownX Jan|2|6 Feb|3|5 Mar|9|11 Apr|6|7 May|8|5 Jun|7|4 Jul|5|5 Aug|10|8 Sep|9|10 Oct|12|14 Nov|15|17 Dec|14|16] (delimiter is '|');	Il risultato del calcolo di STEYX(KnownY,KnownX) è 2,071 (se la formattazione del numero è impostata su 3 cifre decimali).

Vedere anche:

p *Avg - funzione per grafici (page 400)*
p *Sterr - funzione per grafici (page 451)*

Esempi di utilizzo delle funzioni linest

Le funzioni linest vengono utilizzate per trovare valori associati con l'analisi della regressione lineare. In questa sezione viene descritta la procedura di creazione delle visualizzazioni mediante dati campione per trovare i valori delle funzioni linest disponibili in Qlik Sense. Le funzioni linest possono essere utilizzate nello script di caricamento dei dati e nelle espressioni grafiche.

Per le descrizioni della sintassi e degli argomenti, fare riferimento ai singoli argomenti delle funzioni grafiche e di script linest.

Espressioni dati e script utilizzati negli esempi

Caricare i seguenti dati inline ed espressioni script nell'editor caricamento dati per gli esempi linest() in basso.

```
T1: LOAD *, 1 as Grp; LOAD * inline [ X|Y 1|0 2|1 3|3 4|8 5|14 6|20 7|0 8|50 9|25 10|60 11|38
12|19 13|26 14|143 15|98 16|27 17|59 18|78 19|158 20|279 ] (delimiter is '|');                    R1: LOAD
Grp, linest_B(Y,X) as Linest_B, linest_DF(Y,X) as Linest_DF, linest_F(Y,X) as Linest_F,
linest_M(Y,X) as Linest_M, linest_R2(Y,X) as Linest_R2, linest_SEB(Y,X,1,1) as Linest_SEB,
linest_SEM(Y,X) as Linest_SEM, linest_SEY(Y,X) as Linest_SEY, linest_SSREG(Y,X) as Linest_
SSREG, linest_SSRESID(Y,X) as Linest_SSRESID resident T1 group by Grp;
```

Esempio 1: Espressioni di script usando linest

Esempio: Espressioni nello script

Creare una visualizzazione dai calcoli script di caricamento dei dati

Creare una visualizzazione tabella in un foglio Qlik Sense con i seguenti campi come colonne:

- Linest_B
- Linest_DF
- Linest_F
- Linest_M
- Linest_R2
- Linest_SEB
- Linest_SEM
- Linest_SEY
- Linest_SSREG
- Linest_SSRESID

Risultato

La tabella contenente i risultati dei calcoli linest effettuati nello script di caricamento dei dati dovrebbe presentare l'aspetto seguente:

Tabella dei risultati

Linest_B	Linest_DF	Linest_F	Linest_M	Linest_R2	Linest_SEB
-35.047	18	20.788	8.605	0.536	22.607

Tabella dei risultati

Linest_SEM	Linest_SEY	Linest_SSREG	Linest_SSRESID
1.887	48.666	49235.014	42631.186

Esempio 2: Espressioni chart usando linest

Esempio: Espressioni del grafico

Creare una visualizzazione tabella in un foglio Qlik Sense con i seguenti campi come dimensioni:

```
ValueList('Linest_b', 'Linest_df','Linest_f', 'Linest_m','Linest_r2','Linest_SEB','Linest_
SEM','Linest_SEY','Linest_SSREG','Linest_SSRESID')
```

Questa espressione utilizza la funzione delle dimensioni sintetiche per creare etichette con i nomi delle funzioni linest. È possibile modificare l'etichetta in **Linest functions** per risparmiare spazio.

Aggiungere la seguente espressione alla tabella come misura:

```
Pick(Match(ValueList('Linest_b', 'Linest_df','Linest_f', 'Linest_m','Linest_r2','Linest_
SEB','Linest_SEM','Linest_SEY','Linest_SSREG','Linest_SSRESID'),'Linest_b', 'Linest_
df','Linest_f', 'Linest_m','Linest_r2','Linest_SEB','Linest_SEM','Linest_SEY','Linest_
SSREG','Linest_SSRESID'),Linest_b(Y,X),Linest_df(Y,X),Linest_f(Y,X),Linest_m(Y,X),Linest_r2
(Y,X),Linest_SEB(Y,X,1,1),Linest_SEM(Y,X),Linest_SEY(Y,X),Linest_SSREG(Y,X),Linest_SSRESID
(Y,X) )
```

Questa espressione consente di visualizzare il risultato di ciascuna funzione linest in relazione al nome corrispondente nella dimensione sintetica. Il risultato di `Linest_b(Y,X)` viene visualizzato accanto a **linest_b** e così via.

Risultato

Tabella dei risultati

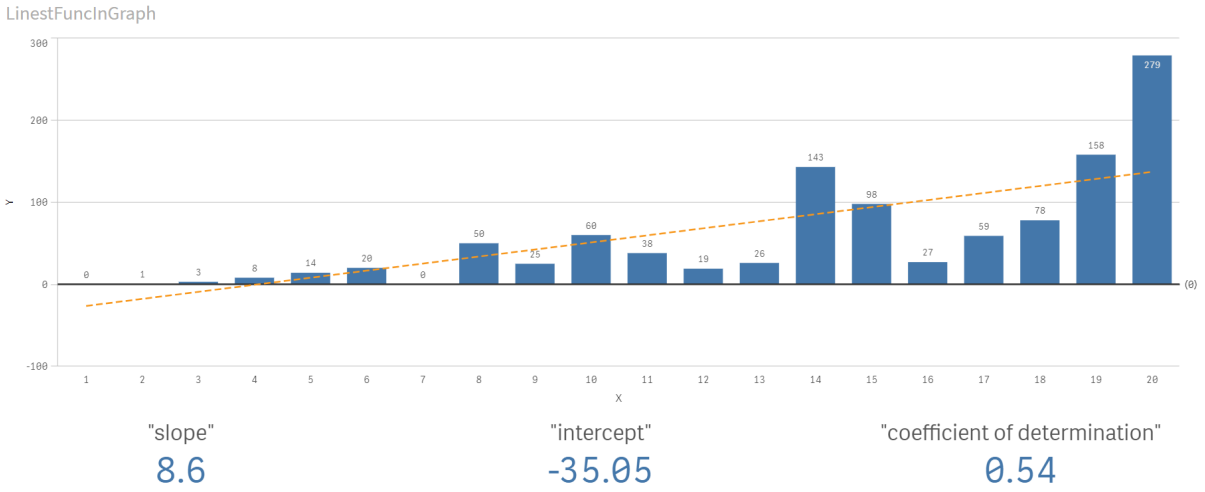
Linest functions	Linest function results
Linest_b	-35.047
Linest_df	18
Linest_f	20.788
Linest_m	8.605
Linest_r2	0.536
Linest_SEB	22.607
Linest_SEM	1.887
Linest_SEY	48.666
Linest_SSREG	49235.014
Linest_SSRESID	42631.186

Esempio 3: Espressioni chart usando linest

Esempio: Espressioni del grafico

1. Creare una visualizzazione grafico a barre in un foglio Qlik Sense con **x** come dimensione e **Y** come misura.
2. Aggiungere una linea di tendenza lineare alla misura Y.
3. Aggiungere una visualizzazione KPI al foglio.
    1. Aggiungere *pendenza* come etichetta per il KPI.
    2. Aggiungere `sum(Linest_M)` come espressione per il KPI.
4. Aggiungere una seconda visualizzazione KPI al foglio.
    1. Aggiungere *Intercetta* come etichetta per il KPI.
    2. Aggiungere `Sum(Linest_B)` come espressione per il KPI.
5. Aggiungere una terza visualizzazione KPI al foglio.
    1. Aggiungere *coefficiente di determinazione* come etichetta per il KPI.
    2. Aggiungere `Sum(Linest_R2)` come espressione per il KPI.

Risultato



"slope"
8.6

"intercept"
-35.05

"coefficient of determination"
0.54

Spiegazione

Il grafico a barre mostra il tracciato dei dati X e Y. Le funzioni linest() pertinenti forniscono valori per l'equazione di regressione lineare su cui si basa la linea di tendenza, cioè y = m * x + b. L'equazione usa il metodo dei "minimi quadrati" per calcolare una linea retta (linea di tendenza) restituendo un array che descrive una linea che meglio si adatta ai dati.

I KPI mostrano i risultati delle funzioni linest() **sum(Linest_M)** per la pendenza e **sum(Linest_B)** per l'intercetta Y, che sono variabili nell'equazione di regressione lineare, e il corrispondente valore aggregato R2 per il coefficiente di determinazione.

Funzioni di test statistici

Le funzioni di test statistici possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche, anche se la sintassi sarà diversa.

Funzioni di test del chi quadrato

Di solito vengono utilizzate per lo studio delle variabili qualitative. È possibile confrontare le frequenze osservate in una tabella di frequenze a senso unico con le frequenze previste oppure studiare il collegamento tra due variabili in una tabella di contingenza.

Funzioni di t-test

Le funzioni t-test vengono utilizzate per un'analisi statistica di due popolazioni medie. Un t-test a due campioni analizza se i due campioni sono diversi e viene generalmente utilizzato quando due distribuzioni standard presentano varianze sconosciute e quando un esperimento utilizza dimensioni ridotte per il campione.

Funzioni di z-test

Un'analisi statistica di due popolazioni medie. Uno z-test con due campioni analizza se i due campioni sono differenti e viene comunemente utilizzato quando due distribuzioni standard presentano varianze conosciute e quando un esperimento utilizza dimensioni notevoli per il campione.

Funzioni chi2-test

Di solito vengono utilizzate per lo studio delle variabili qualitative. È possibile confrontare le frequenze osservate in una tabella di frequenze a senso unico con le frequenze previste oppure studiare il collegamento tra due variabili in una tabella di contingenza.Chi-squared test functions are used to determine whether there is a statistically significant difference between the expected frequencies and the observed frequencies in one or more groups. Often a histogram is used, and the different bins are compared to an expected distribution.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Chi2Test_chi2
Chi2Test_chi2() restituisce il valore aggregato di chi^2-test per una o due serie di valori..

```
Chi2Test_chi2() restituisce il valore aggregato di chi2-test per una o due serie di valori..(col, row, actual_value[, expected_value])
```

Chi2Test_df
Chi2Test_df() restituisce il valore df (gradi di libertà) aggregato di chi^2-test per una o due serie di valori.

```
Chi2Test_df() restituisce il valore df (gradi di libertà) aggregato di chi2-test per una o due serie di valori.(col, row, actual_value[, expected_value])
```

Chi2Test_p
Chi2Test_p() restituisce il valore p (significatività) aggregato di chi^2-test per una o due serie di valori.

```
Chi2Test_p - funzione per grafici(col, row, actual_value[, expected_value])
```

Vedere anche:

p *Funzioni di t-test (page 463)*
p *Funzioni di z-test (page 499)*

Chi2Test_chi2

Chi2Test_chi2() restituisce il valore aggregato di chi^2-test per una o due serie di valori..

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Tutte le funzioni Qlik Sense del chi^2-test presentano gli stessi argomenti.

Sintassi:

```
Chi2Test_chi2(col, row, actual_value[, expected_value])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
col, row	La riga e la colonna specificate nella matrice di valori testati.
actual_value	Il valore osservato dei dati in corrispondenza di **col** e **row** specificati.
expected_value	Il valore previsto per la distribuzione in corrispondenza di **col** e **row** specificati.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
Chi2Test_chi2( Grp, Grade, Count )
Chi2Test_chi2( Gender, Description, Observed, Expected )
```

Vedere anche:

p *Esempi di utilizzo delle funzioni chi2-test nei grafici (page 515)*
p *Esempi di utilizzo delle funzioni chi2-test negli script di caricamento dei dati (page 518)*

Chi2Test_df

Chi2Test_df() restituisce il valore df (gradi di libertà) aggregato di chi^2-test per una o due serie di valori.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Tutte le funzioni Qlik Sense del chi^2-test presentano gli stessi argomenti.

Sintassi:

```
Chi2Test_df(col, row, actual_value[, expected_value])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
col, row	La riga e la colonna specificate nella matrice di valori testati.
actual_value	Il valore osservato dei dati in corrispondenza di **col** e **row** specificati.
expected_value	Il valore previsto per la distribuzione in corrispondenza di **col** e **row** specificati.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
Chi2Test_df( Grp, Grade, Count )
Chi2Test_df( Gender, Description, Observed, Expected )
```

Vedere anche:

p *Esempi di utilizzo delle funzioni chi2-test nei grafici (page 515)*
p *Esempi di utilizzo delle funzioni chi2-test negli script di caricamento dei dati (page 518)*

Chi2Test_p - funzione per grafici

Chi2Test_p() restituisce il valore p (significatività) aggregato di chi^2-test per una o due serie di valori.Il test può essere eseguito o sui valori in **actual_value**, per testare le variazioni all'interno della matrice **col** e **row** specificata o confrontando i valori in **actual_value** con quelli corrispondenti in **expected_value**, se specificato.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Tutte le funzioni Qlik Sense del chi^2-test presentano gli stessi argomenti.

Sintassi:

```
Chi2Test_p(col, row, actual_value[, expected_value])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
col, row	La riga e la colonna specificate nella matrice di valori testati.
actual_value	Il valore osservato dei dati in corrispondenza di **col** e **row** specificati.
expected_value	Il valore previsto per la distribuzione in corrispondenza di **col** e **row** specificati.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
Chi2Test_p( Grp, Grade, Count )
Chi2Test_p( Gender, Description, Observed, Expected )
```

Vedere anche:

p *Esempi di utilizzo delle funzioni chi2-test nei grafici (page 515)*
p *Esempi di utilizzo delle funzioni chi2-test negli script di caricamento dei dati (page 518)*

Funzioni di t-test

Le funzioni t-test vengono utilizzate per un'analisi statistica di due popolazioni medie. Un t-test a due campioni analizza se i due campioni sono diversi e viene generalmente utilizzato quando due distribuzioni standard presentano varianze sconosciute e quando un esperimento utilizza dimensioni ridotte per il campione.

Nelle sezioni seguenti le funzioni di test statistici t-test vengono raggruppate in base al test per studenti con campioni adatto a ogni tipo di funzione.

Creazione di un report t-test tipico (page 520)

T-test con due campioni indipendenti

Le seguenti funzioni possono essere utilizzate per due t-test per studenti con campioni indipendenti.

ttest_conf
TTest_conf restituisce il valore aggregato dell'intervallo di confidenza di t-test per due campioni indipendenti.

TTest_conf restituisce il valore aggregato dell'intervallo di confidenza di t-test per due campioni indipendenti. (grp, value [, sig[, eq_var]]**)**

ttest_df

TTest_df() restituisce il valore aggregato (gradi di libertà) del t-test di Student per due serie indipendenti di valori.

TTest_df() restituisce il valore aggregato (gradi di libertà) del t-test di Student per due serie indipendenti di valori. (grp, value [, eq_var**)**

ttest_dif

TTest_dif() è una funzione numerica che restituisce la differenza media aggregata del t-test di Student per due serie indipendenti di valori.

TTest_dif() è una funzione numerica che restituisce la differenza media aggregata del t-test di Student per due serie indipendenti di valori. (grp, value**)**

ttest_lower

TTest_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori.

TTest_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori. (grp, value [, sig[, eq_var]]**)**

ttest_sig

TTest_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per due serie indipendenti di valori.

TTest_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per due serie indipendenti di valori. (grp, value [, eq_var]**)**

ttest_sterr

TTest_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per due serie indipendenti di valori.

TTest_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per due serie indipendenti di valori. (grp, value [, eq_var]**)**

ttest_t

TTest_t() restituisce il valore t aggregato per due serie indipendenti di valori.

TTest_t() restituisce il valore t aggregato per due serie indipendenti di valori. (grp, value [, eq_var]**)**

ttest_upper

TTest_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori.

```
TTest_upper() restituisce il valore aggregato per il limite superiore
dell'intervallo di confidenza per due serie indipendenti di valori. (grp,
value [, sig [, eq_var]])
```

Due t-test con campioni indipendenti pesati

Le seguenti funzioni si applicano a due t-test per studenti con campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

ttestw_conf

TTestw_conf() restituisce il valore t aggregato per due serie indipendenti di valori.

```
TTestw_conf() restituisce il valore t aggregato per due serie indipendenti di
valori. (weight, grp, value [, sig[, eq_var]])
```

ttestw_df

TTestw_df() restituisce il valore df (gradi di libertà) aggregato del t-test di Student per due serie indipendenti di valori.

```
TTestw_df() restituisce il valore df (gradi di libertà) aggregato del t-test
di Student per due serie indipendenti di valori. (weight, grp, value [, eq_
var])
```

ttestw_dif

TTestw_dif() restituisce la differenza media aggregata del t-test di Student per due serie indipendenti di valori.

```
TTestw_dif() restituisce la differenza media aggregata del t-test di Student
per due serie indipendenti di valori. ( weight, grp, value)
```

ttestw_lower

TTestw_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori.

```
TTestw_lower() restituisce il valore aggregato per il limite inferiore
dell'intervallo di confidenza per due serie indipendenti di valori. (weight,
grp, value [, sig[, eq_var]])
```

ttestw_sig

TTestw_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per due serie indipendenti di valori.

```
TTestw_sig() restituisce il livello di significatività a due code aggregato
del t-test di Student per due serie indipendenti di valori. ( weight, grp,
value [, eq_var])
```

ttestw_sterr

TTestw_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per due serie indipendenti di valori.

```
TTestw_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per due serie indipendenti di valori. (weight, grp, value [, eq_var])
```

ttestw_t

TTestw_t() restituisce il valore t aggregato per due serie indipendenti di valori.

```
TTestw_t() restituisce il valore t aggregato per due serie indipendenti di valori. (weight, grp, value [, eq_var])
```

ttestw_upper

TTestw_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori.

```
TTestw_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori. (weight, grp, value [, sig [, eq_var]])
```

T-test con un unico campione

Le seguenti funzioni possono essere utilizzate per t-test per studenti con un unico campione.

ttest1_conf

TTest1_conf() restituisce il valore aggregato dell'intervallo di confidenza per una serie di valori.

```
TTest1_conf() restituisce il valore aggregato dell'intervallo di confidenza per una serie di valori. (value [, sig])
```

ttest1_df

TTest1_df() restituisce il valore df (gradi di libertà) aggregato del t-test di Student per una serie di valori.

```
TTest1_df() restituisce il valore df (gradi di libertà) aggregato del t-test di Student per una serie di valori. (value)
```

ttest1_dif

TTest1_dif() restituisce la differenza media aggregata del t-test di Student per una serie di valori.

```
TTest1_dif() restituisce la differenza media aggregata del t-test di Student per una serie di valori. (value)
```

ttest1_lower

TTest1_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per una serie di valori.

```
TTest1_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per una serie di valori. (value [, sig])
```

ttest1_sig

TTest1_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per una serie di valori.

```
TTest1_sig() restituisce il livello di significatività a due code aggregato
del t-test di Student per una serie di valori. (value)
```

ttest1_sterr

TTest1_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per una serie di valori.

```
TTest1_sterr() restituisce l'errore standard aggregato del t-test di Student
della differenza media per una serie di valori. (value)
```

ttest1_t

TTest1_t() restituisce il valore t aggregato per una serie di valori.

```
TTest1_t() restituisce il valore t aggregato per una serie di valori. (value)
```

ttest1_upper

TTest1_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per una serie di valori.

```
TTest1_upper() restituisce il valore aggregato per il limite superiore
dell'intervallo di confidenza per una serie di valori. (value [, sig])
```

T-test con un unico campione pesato

Le seguenti funzioni possono essere utilizzate per t-test per studenti con un unico campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

ttest1w_conf

TTest1w_conf() è una funzione **numerica** che restituisce il valore aggregato dell'intervallo di attendibilità per una serie di valori.

```
TTest1w_conf() è una funzione numerica che restituisce il valore aggregato
dell'intervallo di attendibilità per una serie di valori. (weight, value [,
sig])
```

ttest1w_df

TTest1w_df() restituisce il valore df (gradi di libertà) aggregato del t-test di Student per una serie di valori.

```
TTest1w_df() restituisce il valore df (gradi di libertà) aggregato del t-test
di Student per una serie di valori. (weight, value)
```

ttest1w_dif

TTest1w_dif() restituisce la differenza media aggregata del t-test di Student per una serie di valori.

```
TTest1w_dif() restituisce la differenza media aggregata del t-test di Student
per una serie di valori. (weight, value)
```

ttest1w_lower

TTest1w_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per una serie di valori.

```
TTest1w_lower() restituisce il valore aggregato per il limite inferiore
dell'intervallo di confidenza per una serie di valori. (weight, value [,
sig])
```

ttest1w_sig

TTest1w_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per una serie di valori.

```
TTest1w_sig() restituisce il livello di significatività a due code aggregato
del t-test di Student per una serie di valori. (weight, value)
```

ttest1w_sterr

TTest1w_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per una serie di valori.

```
TTest1w_sterr() restituisce l'errore standard aggregato del t-test di Student
della differenza media per una serie di valori. (weight, value)
```

ttest1w_t

TTest1w_t() restituisce il valore t aggregato per una serie di valori.

```
TTest1w_t() restituisce il valore t aggregato per una serie di valori. (
weight, value)
```

ttest1w_upper

TTest1w_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per una serie di valori.

```
TTest1w_upper() restituisce il valore aggregato per il limite superiore
dell'intervallo di confidenza per una serie di valori. (weight, value [,
sig])
```

TTest_conf

TTest_conf restituisce il valore aggregato dell'intervallo di confidenza di t-test per due campioni indipendenti.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest_conf ( grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest_conf( Group, Value )
TTest_conf( Group, Value, Sig, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest_df

TTest_df() restituisce il valore aggregato (gradi di libertà) del t-test di Student per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest_df (grp, value [, eq_var])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest_df( Group, Value )
TTest_df( Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest_dif

TTest_dif() è una funzione numerica che restituisce la differenza media aggregata del t-test di Student per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest_dif (grp, value [, eq_var] )
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest_dif( Group, Value )
TTest_dif( Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest_lower

TTest_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest_lower (grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest_lower( Group, Value )
TTest_lower( Group, Value, Sig, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest_sig

TTest_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest_sig (grp, value [, eq_var])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest_sig( Group, Value )
TTest_sig( Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest_sterr

TTest_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest_sterr (grp, value [, eq_var])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest_sterr( Group, Value )
TTest_sterr( Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest_t

TTest_t() restituisce il valore t aggregato per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest_t(grp, value[, eq_var])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest_t( Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest_upper

TTest_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest_upper (grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest_upper( Group, Value )
TTest_upper( Group, Value, sig, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTestw_conf

TTestw_conf() restituisce il valore t aggregato per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con due campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTestw_conf (weight, grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTestw_conf( Weight, Group, Value )
TTestw_conf( Weight, Group, Value, sig, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTestw_df

TTestw_df() restituisce il valore df (gradi di libertà) aggregato del t-test di Student per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con due campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTestw_df (weight, grp, value [, eq_var])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTestw_df( Weight, Group, Value )
TTestw_df( Weight, Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTestw_dif

TTestw_dif() restituisce la differenza media aggregata del t-test di Student per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con due campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTestw_dif (weight, grp, value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTestw_dif( Weight, Group, Value )
TTestw_dif( Weight, Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTestw_lower

TTestw_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con due campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTestw_lower (weight, grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTestw_lower( Weight, Group, Value )
TTestw_lower( Weight, Group, Value, sig, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTestw_sig

TTestw_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con due campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTestw_sig ( weight, grp, value [, eq_var])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTestw_sig( Weight, Group, Value )
TTestw_sig( Weight, Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTestw_sterr

TTestw_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con due campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTestw_sterr (weight, grp, value [, eq_var])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTestw_sterr( Weight, Group, Value )
TTestw_sterr( Weight, Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTestw_t

TTestw_t() restituisce il valore t aggregato per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con due campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ttestw_t (weight, grp, value [, eq_var])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.

Argomento	Descrizione
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTestw_t( Weight, Group, Value )
TTestw_t( Weight, Group, Value, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTestw_upper

TTestw_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con due campioni indipendenti in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTestw_upper (weight, grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.

Argomento	Descrizione
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTestw_upper( Weight, Group, Value )
TTestw_upper( Weight, Group, Value, sig, false )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1_conf

TTest1_conf() restituisce il valore aggregato dell'intervallo di confidenza per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1_conf (value [, sig ])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest1_conf( value )
TTest1_conf( value, 0.005 )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1_df

TTest1_df() restituisce il valore df (gradi di libertà) aggregato del t-test di Student per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1_df (value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1_df( Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1_dif

TTest1_dif() restituisce la differenza media aggregata del t-test di Student per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1_dif (value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1_dif( Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1_lower

TTest1_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1_lower (value [, sig])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest1_lower( Value )
TTest1_lower( Value, 0.005 )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1_sig

TTest1_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1_sig (value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1_sig( Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1_sterr

TTest1_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1_sterr (value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1_sterr( Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1_t

TTest1_t() restituisce il valore t aggregato per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1_t (value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1_t( Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1_upper

TTest1_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1_upper (value [, sig])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.

Argomento	Descrizione
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest1_upper( Value )
TTest1_upper( Value, 0.005 )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1w_conf

TTest1w_conf() è una funzione **numerica** che restituisce il valore aggregato dell'intervallo di attendibilità per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1w_conf (weight, value [, sig ])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest1w_conf( Weight, Value )
TTest1w_conf( Weight, Value, 0.005 )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1w_df

TTest1w_df() restituisce il valore df (gradi di libertà) aggregato del t-test di Student per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1w_df (weight, value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1w_df( Weight, Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1w_dif

TTest1w_dif() restituisce la differenza media aggregata del t-test di Student per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1w_dif (weight, value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1w_dif( Weight, Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1w_lower

TTest1w_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1w_lower (weight, value [, sig ])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest1w_lower( Weight, Value )
TTest1w_lower( Weight, Value, 0.005 )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1w_sig

TTest1w_sig() restituisce il livello di significatività a due code aggregato del t-test di Student per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1w_sig (weight, value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1w_sig( Weight, Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1w_sterr

TTest1w_sterr() restituisce l'errore standard aggregato del t-test di Student della differenza media per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1w_sterr (weight, value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1w_sterr( Weight, Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1w_t

TTest1w_t() restituisce il valore t aggregato per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1w_t ( weight, value)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
TTest1w_t( Weight, Value )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

TTest1w_upper

TTest1w_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per una serie di valori.

Questa funzione viene applicata ai t-test di student con un campione in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
TTest1w_upper (weight, value [, sig])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I campioni da valutare. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
weight	Ogni valore in **value** può essere contato una o più volte in base al corrispondente valore del peso in **weight**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
TTest1w_upper( Weight, Value )
TTest1w_upper( Weight, Value, 0.005 )
```

Vedere anche:

p *Creazione di un report t-test tipico (page 520)*

Funzioni di z-test

Un'analisi statistica di due popolazioni medie. Uno z-test con due campioni analizza se i due campioni sono differenti e viene comunemente utilizzato quando due distribuzioni standard presentano varianze conosciute e quando un esperimento utilizza dimensioni notevoli per il campione.

Le funzioni di test statistici z-test vengono raggruppate in base al tipo di serie di dati di input adatto alla funzione.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Esempi di utilizzo delle funzioni z-test (page 523)

Funzioni per il formato a colonna singola

Le funzioni seguenti vengono utilizzate per z-test con serie di dati di input semplici.

ztest_conf

ZTest_conf() restituisce il valore z aggregato per una serie di valori.

```
ZTest_conf() restituisce il valore z aggregato per una serie di valori.
(value [, sigma [, sig ])
```

ztest_dif

ZTest_dif() restituisce la differenza media aggregata di z-test per una serie di valori.

```
ZTest_dif() restituisce la differenza media aggregata di z-test per una serie
di valori. (value [, sigma])
```

ztest_sig

ZTest_sig() restituisce il livello di significatività a due code aggregato di z-test per una serie di valori.

```
ZTest_sig() restituisce il livello di significatività a due code aggregato di
z-test per una serie di valori. (value [, sigma])
```

ztest_sterr

ZTest_sterr() restituisce l'errore standard aggregato di z-test della differenza media per una serie di valori.

```
ZTest_sterr() restituisce l'errore standard aggregato di z-test della
differenza media per una serie di valori. (value [, sigma])
```

ztest_z

ZTest_z() restituisce il valore z aggregato per una serie di valori.

```
ZTest_z() restituisce il valore z aggregato per una serie di valori. (value
[, sigma])
```

ztest_lower

ZTest_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori.

```
ZTest_lower() restituisce il valore aggregato per il limite inferiore
dell'intervallo di confidenza per due serie indipendenti di valori. (grp,
value [, sig [, eq_var]])
```

ztest_upper

ZTest_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori.

```
ZTest_upper() restituisce il valore aggregato per il limite superiore
dell'intervallo di confidenza per due serie indipendenti di valori. (grp,
value [, sig [, eq_var]])
```

Funzioni per il formato a due colonne pesate

Le funzioni seguenti vengono utilizzate per z-test in cui la serie di dati di input viene fornita in un formato a due colonne pesate.

ztestw_conf

ZTestw_conf() restituisce il valore z aggregato dell'intervallo di confidenza per una serie di valori.

```
ZTestw_conf() restituisce il valore z aggregato dell'intervallo di confidenza per una serie di valori. (weight, value [, sigma [, sig]])
```

ztestw_dif

ZTestw_dif() restituisce la differenza media aggregata di z-test per una serie di valori.

```
ZTestw_dif() restituisce la differenza media aggregata di z-test per una serie di valori.  (weight, value [, sigma])
```

ztestw_lower

ZTestw_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori.

```
ZTestw_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori. (weight, value [, sigma])
```

ztestw_sig

ZTestw_sig() restituisce il livello di significatività a due code aggregato di z-test per una serie di valori.

```
ZTestw_sig() restituisce il livello di significatività a due code aggregato di z-test per una serie di valori. (weight, value [, sigma])
```

ztestw_sterr

ZTestw_sterr() restituisce l'errore standard aggregato di z-test della differenza media per una serie di valori.

```
ZTestw_sterr() restituisce l'errore standard aggregato di z-test della differenza media per una serie di valori. (weight, value [, sigma])
```

ztestw_upper

ZTestw_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori.

```
ZTestw_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori. (weight, value [, sigma])
```

ztestw_z

ZTestw_z() restituisce il valore z aggregato per una serie di valori.

```
ZTestw_z() restituisce il valore z aggregato per una serie di valori. (weight, value [, sigma])
```

ZTest_z

ZTest_z() restituisce il valore z aggregato per una serie di valori.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTest_z(value[, sigma])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. Si assume una popolazione media pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quella media dai valori campione.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTest_z( Value-TestValue )
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTest_sig

ZTest_sig() restituisce il livello di significatività a due code aggregato di z-test per una serie di valori.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTest_sig(value[, sigma])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. Si assume una popolazione media pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quella media dai valori campione.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTest_sig(Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTest_dif

ZTest_dif() restituisce la differenza media aggregata di z-test per una serie di valori.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTest_dif(value[, sigma])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. Si assume una popolazione media pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quella media dai valori campione.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTest_dif(Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTest_sterr

ZTest_sterr() restituisce l'errore standard aggregato di z-test della differenza media per una serie di valori.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTest_sterr(value[, sigma])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. Si assume una popolazione media pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quella media dai valori campione.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTest_sterr(Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTest_conf

ZTest_conf() restituisce il valore z aggregato per una serie di valori.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTest_conf(value[, sigma[, sig]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. Si assume una popolazione media pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quella media dai valori campione.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTest_conf(Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTest_lower

ZTest_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTest_lower (grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
ZTest_lower( Group, Value )
ZTest_lower( Group, Value, sig, false )
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTest_upper

ZTest_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTest_upper (grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
ZTest_upper( Group, Value )
ZTest_upper( Group, Value, sig, false )
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTestw_z

ZTestw_z() restituisce il valore z aggregato per una serie di valori.

Questa funzione viene applicata agli z-test in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTestw_z (weight, value [, sigma])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori devono essere restituiti da **value**. Si assume una media campione pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quel valore dai valori campione.
weight	Ogni valore campione in **value** può essere conteggiato una o più volte in base al corrispondente valore del peso in **weight**.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTestw_z( Weight, Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTestw_sig

ZTestw_sig() restituisce il livello di significatività a due code aggregato di z-test per una serie di valori.

Questa funzione viene applicata agli z-test in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTestw_sig (weight, value [, sigma])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori devono essere restituiti da **value**. Si assume una media campione pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quel valore dai valori campione.
weight	Ogni valore campione in **value** può essere conteggiato una o più volte in base al corrispondente valore del peso in **weight**.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTestw_sig( Weight, Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTestw_dif

ZTestw_dif() restituisce la differenza media aggregata di z-test per una serie di valori.

Questa funzione viene applicata agli z-test in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTestw_dif ( weight, value [, sigma])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori devono essere restituiti da **value**. Si assume una media campione pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quel valore dai valori campione.
weight	Ogni valore campione in **value** può essere conteggiato una o più volte in base al corrispondente valore del peso in **weight**.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTestw_dif( Weight, Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTestw_sterr

ZTestw_sterr() restituisce l'errore standard aggregato di z-test della differenza media per una serie di valori.

Questa funzione viene applicata agli z-test in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTestw_sterr (weight, value [, sigma])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori devono essere restituiti da **value**. Si assume una media campione pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quel valore dai valori campione.
weight	Ogni valore campione in **value** può essere conteggiato una o più volte in base al corrispondente valore del peso in **weight**.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTestw_sterr( Weight, Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTestw_conf

ZTestw_conf() restituisce il valore z aggregato dell'intervallo di confidenza per una serie di valori.

Questa funzione viene applicata agli z-test in cui la serie di dati di input viene fornita in formato a due colonne pesate.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTest_conf(weight, value[, sigma[, sig]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. Si assume una popolazione media pari a 0. Se si desidera che il test sia eseguito su un'altra media, sottrarre quella media dai valori campione.
weight	Ogni valore campione in **value** può essere conteggiato una o più volte in base al corrispondente valore del peso in **weight**.
sigma	Se conosciuta, la deviazione standard può essere dichiarata in **sigma**. Se **sigma** viene omesso, verrà utilizzata la deviazione standard del campione effettiva.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempio:

```
ZTestw_conf( Weight, Value-TestValue)
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTestw_lower

ZTestw_lower() restituisce il valore aggregato per il limite inferiore dell'intervallo di confidenza per due serie indipendenti di valori.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTestw_lower (grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
ZTestw_lower( Group, Value )
ZTestw_lower( Group, Value, sig, false )
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

ZTestw_upper

ZTestw_upper() restituisce il valore aggregato per il limite superiore dell'intervallo di confidenza per due serie indipendenti di valori.

Questa funzione viene applicata ai t-test di student con campioni indipendenti.

Se la funzione viene utilizzata nello script di caricamento dei dati, i valori verranno ripetuti su un insieme di record, come definito da una clausola group by.

Se la funzione viene utilizzata in un'espressione grafica, i valori verranno ripetuti sulle dimensioni del grafico.

Sintassi:

```
ZTestw_upper (grp, value [, sig [, eq_var]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	I valori campione da valutare. I valori campione devono essere raggruppati n modo logico come specificato esattamente dai due valori in **group**. Se nello script Load non è fornito un nome di campo per i valori di esempio, al campo verrà automaticamente assegnato il nome **Value**.
grp	Il campo contenente i nomi di ciascuno dei due gruppi di esempi. Se nello script Load non è fornito un nome di campo per il gruppo, al campo verrà automaticamente assegnato il nome **Type**.
sig	Il livello di significatività a due code può essere specificato in **sig**. Se omesso, **sig** viene impostato su 0,025, per un intervallo di confidenza pari al 95%.
eq_var	Se **eq_var** è specificato come False (0), verranno presupposte varianze separate dei due esempi. Se **eq_var** è specificato come True (1), verranno presupposte varianze identiche dei due esempi.

Limiti:

I valori di testo, i valori NULL e i valori mancanti nel valore dell'espressione determinano la restituzione di NULL.

Esempi:

```
ZTestw_upper( Group, Value )
ZTestw_upper( Group, Value, sig, false )
```

Vedere anche:

p *Esempi di utilizzo delle funzioni z-test (page 523)*

Esempi delle funzioni di test statistici

In questa sezione sono inclusi esempi di funzioni di test statistici applicati ai grafici e allo script di caricamento dei dati.

Esempi di utilizzo delle funzioni chi2-test nei grafici

Le funzioni chi2-test vengono utilizzate per trovare i valori associati all'analisi statistica del chi quadrato.

In questa sezione viene descritto come creare visualizzazioni utilizzando dati campione per trovare i valori delle funzioni del test di distribuzione del chi quadrato disponibili in Qlik Sense. Per le descrizioni della sintassi e degli argomenti, fare riferimento ai singoli argomenti delle funzioni dei grafici di chi2-test.

Caricamento dei dati per i campioni

Esistono tre set di dati campione che descrivono tre differenti campioni statistici da caricare nello script.

Procedere come indicato di seguito:

1. Creare una nuova app.
2. Nell'editor caricamento dati immettere quanto segue:

```
// Sample_1 data is pre-aggregated... Note: make sure you set your DecimalSep='.' at the
top of the script.
Sample_1:
LOAD * inline [
Grp,Grade,Count
I,A,15
I,B,7
I,C,9
I,D,20
I,E,26
I,F,19
II,A,10
II,B,11
II,C,7
II,D,15
II,E,21
II,F,16
];
// Sample_2 data is pre-aggregated: If raw data is used, it must be aggregated using
count()...
Sample_2:
LOAD * inline [
Sex,Opinion,OpCount
1,2,58
1,1,11
1,0,10
2,2,35
2,1,25
2,0,23 ] (delimiter is ',');
// Sample_3a data is transformed using the crosstable statement...
Sample_3a:
crosstable(Gender, Actual) LOAD
Description,
[Men (Actual)] as Men,
[Women (Actual)] as Women;
LOAD * inline [
Men (Actual),Women (Actual),Description
58,35,Agree
11,25,Neutral
10,23,Disagree ] (delimiter is ',');
// Sample_3b data is transformed using the crosstable statement...
Sample_3b:
crosstable(Gender, Expected) LOAD
Description,
[Men (Expected)] as Men,
```

```
[Women (Expected)] as Women;
LOAD * inline [
Men (Expected),Women (Expected),Description
45.35,47.65,Agree
17.56,18.44,Neutral
16.09,16.91,Disagree ] (delimiter is ',');
// Sample_3a and Sample_3b will result in a (fairly harmless) Synthetic Key...
```

3. Fare clic su per caricare dati.

Creazione di visualizzazioni delle funzioni grafiche chi2-test

Esempio: Campione 1

Procedere come indicato di seguito:

1. Nell'editor caricamento dati, fare clic su per accedere alla panoramica App, quindi fare clic sul foglio creato in precedenza.
   Viene aperta la vista foglio.
2. Fare clic su **Modifica foglio** per modificare il foglio.
3. Da **Grafici** aggiungere una tabella e da **Campi** aggiungere Grp, Grade e Count come dimensioni. In questa tabella sono mostrati i dati campione.
4. Aggiungere un'altra tabella con la seguente espressione come dimensione:
   `ValueList('p','df','Chi2')`
   Viene utilizzata la funzione delle dimensioni sintetiche per creare etichette per le dimensioni con i nomi delle tre funzioni chi2-test.
5. Aggiungere la seguente espressione alla tabella come misura:
   ```
   IF(ValueList('p','df','Chi2')='p',Chi2Test_p(Grp,Grade,Count),
   IF(ValueList('p','df','Chi2')='df',Chi2Test_df(Grp,Grade,Count),
   Chi2Test_Chi2(Grp,Grade,Count)))
   ```
   Ciò ha l'effetto di inserire il valore risultante di ciascuna funzione chi2-test presente nella tabella accanto alla dimensione sintetica associata.
6. Impostare la **Formattazione numero** della misura su **Numero** e **3Cifre significative**.

> *Nell'espressione per la misura, è possibile utilizzare invece la seguente espressione:* `Pick(Match(ValueList('p','df','Chi2'),'p','df','Chi2'),Chi2Test_p(Grp,Grade,Count),Chi2Test_df(Grp,Grade,Count),Chi2Test_Chi2(Grp,Grade,Count))`

Risultato:

La tabella risultante per le funzioni chi2-test per i dati del Campione 1 conterrà i seguenti valori:

Tabella dei risultati

p	df	Chi2
0.820	5	2.21

Esempio: Campione 2

Procedere come indicato di seguito:

1. Nel foglio che si stava modificando nell'esempio Campione 1, da **Grafici** aggiungere una tabella e da **Campi** aggiungere Sex, Opinion e OpCount come dimensioni.
2. Creare una copia della tabella dei risultati del Campione 1 utilizzando i comandi **Copia** e **Incolla**. Modificare l'espressione nella misura e sostituire gli argomenti in tutte e tre le funzioni chi2-test con i nomi dei campi utilizzati nei dati del campione 2, ad esempio: `Chi2Test_p(Sex,Opinion,OpCount)`.

Risultato:

La tabella risultante per le funzioni chi2-test per i dati del Campione 2 conterrà i seguenti valori:

Tabella dei risultati

p	df	Chi2
0.000309	2	16.2

Esempio: Campione 3

Procedere come indicato di seguito:

1. Creare altre due tabelle nello stesso modo degli esempi per i dati del Campione 1 e del Campione 2. Nella tabella delle dimensioni utilizzare i seguenti campi come dimensioni: Gender, Description, Actual e Expected.
2. Nella tabella dei risultati utilizzare i nomi dei campi utilizzati nei dati del Campione 3, ad esempio: `Chi2Test_p(Gender,Description,Actual,Expected)`.

Risultato:

La tabella risultante per le funzioni chi2-test per i dati del Campione 3 conterrà i seguenti valori:

Tabella dei risultati

p	df	Chi2
0.000308	2	16.2

Esempi di utilizzo delle funzioni chi2-test negli script di caricamento dei dati

Le funzioni chi2-test vengono utilizzate per trovare i valori associati all'analisi statistica del chi quadrato. In questa sezione viene descritto come utilizzare le funzioni del test di distribuzione del chi quadrato disponibili negli script di caricamento dei dati in Qlik Sense. Per le descrizioni della sintassi e degli argomenti, fare riferimento ai singoli argomenti delle funzioni di script di chi2-test.

In questo esempio viene utilizzata una tabella contenente il numero di studenti che ha ottenuto un voto (da A a F) per due gruppi di studenti (I e II).

Data table

Group	A	B	C	D	E	F
I	15	7	9	20	26	19
II	10	11	7	15	21	16

Caricamento dei dati campione

Procedere come indicato di seguito:

1. Creare una nuova app.
2. Nell'editor caricamento dati immettere quanto segue:
```
// Sample_1 data is pre-aggregated... Note: make sure you set your DecimalSep='.' at the top of the script.
Sample_1:
LOAD * inline [
Grp,Grade,Count
I,A,15
I,B,7
I,C,9
I,D,20
I,E,26
I,F,19
II,A,10
II,B,11
II,C,7
II,D,15
II,E,21
II,F,16
];
```
3. Fare clic su per caricare dati.

I dati campione sono stati ora caricati.

Caricamento dei valori delle funzioni chi2-test

Verranno ora caricati i valori di chi2-test basati sui dati campione in una nuova tabella, raggruppati per Grp.

Procedere come indicato di seguito:

1. Nell'editor caricamento dati aggiungere alla fine dello script il codice seguente:
```
// Sample_1 data is pre-aggregated... Note: make sure you set your DecimalSep='.' at the top of the script.
Chi2_table:
LOAD Grp,
Chi2Test_chi2(Grp, Grade, Count) as chi2,
Chi2Test_df(Grp, Grade, Count) as df,
Chi2Test_p(Grp, Grade, Count) as p
resident Sample_1 group by Grp;
```
2. Fare clic su per caricare dati.

Sono stati ora caricati i valori di chi2-test in una tabella denominata Chi2_table.

Risultati

È possibile ora visualizzare i valori di chi2-test risultanti nel sistema di visualizzazione modello dati in **Anteprima**. Dovrebbero ora avere l'aspetto seguente:

Results

Grp	chi2	df	p
I	16.00	5	0.007
II	9.40	5	0.094

Creazione di un report t-test tipico

Un tipico report t-test per studenti può includere tabelle con risultati **Group Statistics** e **Independent Samples Test**.

Nelle sezioni successive verrà trattata la creazione di queste tabelle utilizzando le funzioni t-test di Qlik Sense applicate a due gruppi di campioni indipendenti, Observation e Comparison. Le tabelle corrispondenti per questi campioni avranno l'aspetto seguente:

Statistiche di gruppo

Type	N	Mean	Standard Deviation	Standard Error Mean
Comparison	20	11.95	14.61245	3.2674431
Observation	20	27.15	12.507997	2.7968933

Independent Sample Test

Independent Sample Test

Type	t	df	Sig. (2-tailed)	Mean Difference	Standard Error Difference	95% Confidence Interval of the Difference (Lower)	95% Confidence Interval of the Difference (Upper)
Equal Variance not Assumed	3.534	37.116717335823	0.001	15.2	4.30101	6.48625	23.9137
Equal Variance Assumed	3.534	38	0.001	15.2	4.30101	6.49306	23.9069

Caricamento dei dati campione

Procedere come indicato di seguito:

1. Creare una nuova app utilizzando un nuovo foglio, quindi aprire il foglio appena creato.
2. Nell'editor caricamento dati immettere quanto segue:
```
Table1:
crosstable LOAD recno() as ID, * inline [
Observation|Comparison
35|2
40|27
12|38
15|31
21|1
14|19
46|1
10|34
28|3
48|1
16|2
30|3
32|2
48|1
31|2
22|1
12|3
39|29
19|37
25|2 ] (delimiter is '|');
```
   Nello script Load **recno()** è incluso perché **crosstable** richiede tre argomenti. Pertanto, **recno()** fornisce semplicemente un argomento aggiuntivo, in questo caso un ID per ciascuna riga. Senza di esso i valori di esempio di **Comparison** non verrebbero caricati.
3. Fare clic su per caricare dati.

Creazione della tabella Group Statistics

Procedere come indicato di seguito:

1. Nell'editor caricamento dati, fare clic su per accedere alla panoramica App, quindi fare clic sul foglio creato in precedenza.
   Viene visualizzata la vista foglio.
2. Fare clic su **Modifica foglio** per modificare il foglio.
3. Utilizzare **Grafici** per aggiungere una tabella, mentre utilizzare **Campi** per aggiungere le espressioni seguenti come misure:

   Espressioni di esempio

Etichetta	Espressione
N	Count(Value)
Mean	Avg(Value)
Standard Deviation	Stdev(Value)
Standard Error Mean	Sterr(Value)

4. Aggiungere Type come una dimensione alla tabella.
5. Fare clic su **Ordinamento** e spostare Type nella parte superiore dell'elenco dell'ordinamento.

Risultato:

Una tabella Group Statistics per questi campioni avrà l'aspetto seguente:

Statistiche di gruppo

Type	N	Mean	Standard Deviation	Standard Error Mean
Comparison	20	11.95	14.61245	3.2674431
Observation	20	27.15	12.507997	2.7968933

Creazione della tabella Two Independent Sample Student's T-test

Procedere come indicato di seguito:

1. Fare clic su **Modifica foglio** per modificare il foglio.
2. Aggiungere l'espressione seguente come dimensione alla tabella. `=ValueList (Dual('Equal Variance not Assumed', 0), Dual('Equal Variance Assumed', 1))`
3. Utilizzare **Grafici** per aggiungere una tabella con le espressioni seguenti come misure:

Espressioni di esempio

Etichetta	Espressione
conf	if(ValueList (Dual('Equal Variance not Assumed', 0), Dual('Equal Variance Assumed', 1)),TTest_conf(Type, Value),TTest_conf(Type, Value, 0))
t	if(ValueList (Dual('Equal Variance not Assumed', 0), Dual('Equal Variance Assumed', 1)),TTest_t(Type, Value),TTest_t(Type, Value, 0))
df	if(ValueList (Dual('Equal Variance not Assumed', 0), Dual('Equal Variance Assumed', 1)),TTest_df(Type, Value),TTest_df(Type, Value, 0))
Sig. (2-tailed)	if(ValueList (Dual('Equal Variance not Assumed', 0), Dual('Equal Variance Assumed', 1)),TTest_sig(Type, Value),TTest_sig(Type, Value, 0))
Mean Difference	TTest_dif(Type, Value)
Standard Error Difference	if(ValueList (Dual('Equal Variance not Assumed', 0), Dual('Equal Variance Assumed', 1)),TTest_sterr(Type, Value),TTest_sterr(Type, Value, 0))
95% Confidence Interval of the Difference (Lower)	if(ValueList (Dual('Equal Variance not Assumed', 0), Dual('Equal Variance Assumed', 1)),TTest_lower(Type, Value,(1-(95)/100)/2),TTest_lower (Type, Value,(1-(95)/100)/2, 0))
95% Confidence Interval of the Difference (Upper)	if(ValueList (Dual('Equal Variance not Assumed', 0), Dual('Equal Variance Assumed', 1)),TTest_upper(Type, Value,(1-(95)/100)/2),TTest_upper (Type, Value,(1-(95)/100)/2, 0))

Risultato:

Independent Sample Test

Type	t	df	Sig. (2-tailed)	Mean Difference	Standard Error Difference	95% Confidence Interval of the Difference (Lower)	95% Confidence Interval of the Difference (Upper)
Equal Variance not Assumed	3.534	37.116717335823	0.001	15.2	4.30101	6.48625	23.9137
Equal Variance Assumed	3.534	38	0.001	15.2	4.30101	6.49306	23.9069

Esempi di utilizzo delle funzioni z-test

Le funzioni z-test vengono utilizzate per individuare valori associati con l'analisi statistica z-test per campioni con grandi quantità di dati, in generale superiori a 30, e di cui si conosce la varianza.

In questa sezione viene descritta la procedura di creazione delle visualizzazioni mediante dati campione per trovare i valori delle funzioni z-test disponibili in Qlik Sense. Per le descrizioni della sintassi e degli argomenti, fare riferimento ai singoli argomenti delle funzioni dei grafici di z-test.

Caricamento dei dati campione

I dati campione utilizzati in questo esempio sono identici a quelli utilizzati negli esempi delle funzioni t-test. Di norma, le dimensioni dei dati campione risulterebbero troppo ridotte per le analisi z-test, tuttavia sono sufficienti per illustrare l'utilizzo delle diverse funzioni z-test in Qlik Sense.

Procedere come indicato di seguito:

1. Creare una nuova app utilizzando un nuovo foglio, quindi aprire il foglio appena creato.

   Se è stata creata un'app per le funzioni t-test, è possibile utilizzarla e creare un nuovo foglio per queste funzioni.

2. Nell'editor caricamento dati immettere quanto segue:

```
Table1:
crosstable LOAD recno() as ID, * inline [
Observation|Comparison
35|2
40|27
12|38
```

```
15|31
21|1
14|19
46|1
10|34
28|3
48|1
16|2
30|3
32|2
48|1
31|2
22|1
12|3
39|29
19|37
25|2 ] (delimiter is '|');
```

Nello script Load **recno()** è incluso perché **crosstable** richiede tre argomenti. Pertanto, **recno()** fornisce semplicemente un argomento aggiuntivo, in questo caso un ID per ciascuna riga. Senza di esso i valori di esempio di **Comparison** non verrebbero caricati.

3. Fare clic su per caricare dati.

Creazione di visualizzazioni delle funzioni grafiche di z-test

Procedere come indicato di seguito:

1. Nell'editor caricamento dati, fare clic su per accedere alla panoramica App, quindi fare clic sul foglio creato durante il caricamento dei dati.
   Viene aperta la vista foglio.
2. Fare clic su **Modifica foglio** per modificare il foglio.
3. Da **Grafici** aggiungere una tabella e da **Campi** aggiungere Type come dimensione.
4. Aggiungere alla tabella le espressioni seguenti come misure.

Espressioni di esempio

Etichetta	Espressione
ZTest Conf	ZTest_conf(Value)
ZTest Dif	ZTest_dif(Value)
ZTest Sig	ZTest_sig(Value)
ZTest Sterr	ZTest_sterr(Value)
ZTest Z	ZTest_z(Value)

*Per visualizzare valori validi, potrebbe essere necessario regolare la formattazione numero delle misure. La tabella sarebbe più facilmente leggibile se la formattazione dei numeri sulla maggior parte delle misure fosse impostata su **Numero>Semplice**, anziché su **Auto**. Ma per ZTest Sig, ad esempio, utilizzare la formattazione numerica: **Personalizzato** e quindi modificare il modello di formattazione in **# ##**.*

Risultato:

La tabella risultante per le funzioni z-test per i dati campione conterrà i valori seguenti:

Tabella dei risultati

Type	ZTest Conf	ZTest Dif	ZTest Sig	ZTest Sterr	ZTest Z
Comparison	6.40	11.95	0.000123	3.27	3.66
Value	5.48	27.15	0.001	2.80	9.71

Creazione di visualizzazioni delle funzioni grafiche di z-testw

Le funzioni z-testw devono essere utilizzate quando la serie di dati di input presenta un formato a due colonne pesate. Le espressioni richiedono un valore per l'argomento weight. Gli esempi qui riportati utilizzano il valore 2 dappertutto, tuttavia si potrebbe utilizzare un'espressione, che definisce un valore weight per ciascuna osservazione.

Esempi e risultati:

Utilizzando gli stessi dati campione e la stessa formattazione dei numeri delle funzioni z-test, la tabella risultante per le funzioni z-testw conterrà i seguenti valori:

Tabella dei risultati

Type	ZTestw Conf	ZTestw Dif	ZTestw Sig	ZTestw Sterr	ZTestw Z
Comparison	3.53	2.95	5.27e-005	1.80	3.88
Value	2.97	34.25	0	4.52	20.49

Funzioni di aggregazione delle stringhe

In questa sezione vengono descritte le funzioni di aggregazione relative alle stringhe.

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Funzioni di aggregazione delle stringhe nello script di caricamento dei dati

Concat

Concat() viene utilizzato per unire le stringhe. Questa funzione di script restituisce la concatenazione di stringhe aggregata di tutti i valori dell'espressione ripetuti su un insieme di record, come definito da una clausola **group by**.

```
Concat ([ distinct ] expression [, delimiter [, sort-weight]])
```

FirstValue

FirstValue() restituisce il valore che è stato caricato per primo dai record definiti dall'espressione, ordinato in base a una clausola **group by**.

Questa funzione è disponibile solo come funzione di script.

```
FirstValue  (expression)
```

LastValue

LastValue() restituisce il valore che è stato caricato per ultimo dai record definiti dall'espressione, ordinato in base a una clausola **group by**.

Questa funzione è disponibile solo come funzione di script.

```
LastValue  (expression)
```

MaxString

MaxString() individua i valori di stringa nell'espressione e restituisce l'ultimo valore di testo ordinato su un insieme di record, come definito dalla clausola **group by**.

```
MaxString (expression )
```

MinString

MinString() individua i valori di stringa nell'espressione e restituisce il primo valore di testo ordinato su un insieme di record, come definito dalla clausola **group by**.

```
MinString (expression )
```

Funzioni di aggregazione delle stringhe nei grafici

Le seguenti funzioni grafiche sono disponibili per l'aggregazione delle stringhe nei grafici.

Concat

Concat() viene utilizzata per unire le stringhe. La funzione restituisce la concatenazione di stringhe aggregata di tutti i valori dell'espressione valutata su ciascuna dimensione.

```
Concat - funzione per grafici({[SetExpression] [DISTINCT] [TOTAL [<fld{, fld}>]] string[, delimiter[, sort_weight]])
```

MaxString

MaxString() trova valori di stringa nell'espressione o nel campo e restituisce l'ultimo valore di testo nel criterio di ordinamento alfabetico.

```
MaxString - funzione per grafici({[SetExpression] [TOTAL [<fld{, fld}>]]} expr)
```

MinString

MinString() trova valori di stringa nell'espressione o nel campo e restituisce il primo valore di testo nel criterio di ordinamento alfabetico.

```
MinString - funzione per grafici({[SetExpression] [TOTAL [<fld {, fld}>]]} expr)
```

Concat

Concat() viene utilizzato per unire le stringhe. Questa funzione di script restituisce la concatenazione di stringhe aggregata di tutti i valori dell'espressione ripetuti su un insieme di record, come definito da una clausola **group by**.

Sintassi:

```
Concat ([ distinct ] string [, delimiter [, sort-weight]])
```

Tipo di dati restituiti: stringa

Argomenti:

L'espressione o il campo contenente la stringa da elaborare.

Argomenti

Argomento	Descrizione
string	L'espressione o il campo contenente la stringa da elaborare.
delimiter	Ogni valore può essere separato dalla stringa trovata in delimiter.
sort-weight	L'ordine di concatenazione può essere determinato dal valore della dimensione **sort-weight**, se presente, con la stringa corrispondente al valore più basso che compare per primo nella concatenazione.
distinct	Se la parola **distinct** è riportata prima dell'espressione, tutti i duplicati vengono ignorati.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Esempi e risultati

Esempio	Risultato	Risultati una volta aggiunti a un foglio
TeamData: LOAD * inline [SalesGroup|Team|Date|Amount East|Gamma|01/05/2013|20000 East|Gamma|02/05/2013|20000 West|Zeta|01/06/2013|19000 East|Alpha|01/07/2013|25000 East|Delta|01/08/2013|14000 West|Epsilon|01/09/2013|17000 West|Eta|01/10/2013|14000 East|Beta|01/11/2013|20000 West|Theta|01/12/2013|23000] (delimiter is '|'); Concat1: LOAD SalesGroup,Concat(Team) as TeamConcat1 Resident TeamData Group By SalesGroup;	SalesGroup East West	TeamConcat1 AlphaBetaDeltaGammaGamma EpsilonEtaThetaZeta
Presupponendo che la tabella **TeamData** venga caricata come nell'esempio precedente: LOAD SalesGroup,Concat(distinct Team,'-') as TeamConcat2 Resident TeamData Group By SalesGroup;	SalesGroup East West	TeamConcat2 Alpha-Beta-Delta-Gamma Epsilon-Eta-Theta-Zeta
Presupponendo che la tabella **TeamData** venga caricata come nell'esempio precedente. Poiché è stato aggiunto l'argomento per **sort-weight**, i risultati vengono ordinati in base al valore della dimensione Amount: LOAD SalesGroup,Concat(distinct Team,'- ',Amount) as TeamConcat2 Resident TeamData Group By SalesGroup;	SalesGroup East West	TeamConcat2 Delta-Beta-Gamma-Alpha Eta-Epsilon-Zeta-Theta

Concat - funzione per grafici

Concat() viene utilizzata per unire le stringhe. La funzione restituisce la concatenazione di stringhe aggregata di tutti i valori dell'espressione valutata su ciascuna dimensione.

Sintassi:

```
Concat({[SetExpression] [DISTINCT] [TOTAL [<fld{, fld}>]]} string[, delimiter
[, sort_weight]])
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
string	L'espressione o il campo contenente la stringa da elaborare.
delimiter	Ogni valore può essere separato dalla stringa trovata in delimiter.
sort-weight	L'ordine di concatenazione può essere determinato dal valore della dimensione **sort-weight**, se presente, con la stringa corrispondente al valore più basso che compare per primo nella concatenazione.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se la parola **DISTINCT** è riportata prima degli argomenti della funzione, i duplicati risultanti dalla valutazione degli argomenti della funzione vengono ignorati.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {,fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Results table

SalesGroup	Amount	Concat(Team)	Concat(TOTAL <SalesGroup> Team)
East	25000	Alpha	AlphaBetaDeltaGammaGamma
East	20000	BetaGammaGamma	AlphaBetaDeltaGammaGamma
East	14000	Delta	AlphaBetaDeltaGammaGamma
West	17000	Epsilon	EpsilonEtaThetaZeta
West	14000	Eta	EpsilonEtaThetaZeta
West	23000	Theta	EpsilonEtaThetaZeta
West	19000	Zeta	EpsilonEtaThetaZeta

Esempi di funzioni

Esempio	Risultato
`Concat(Team)`	La tabella viene creata dalle dimensioni SalesGroup e Amount e dalle variazioni sulla misura Concat(Team). Ignorando il risultato di Totali, è necessario ricordare che sebbene esistano dati per otto valori di Team distribuiti tra due valori di SalesGroup, l'unico risultato della misura Concat(Team) che concatena più di un valore di stringa Team nella tabella è la riga contenente la dimensione Amount 20000, che restituisce il risultato BetaGammaGamma. Ciò si verifica perché esistono tre valori per Amount 20000 nei dati di input. Tutti gli altri risultati restano non concatenati quando la misura viene estesa tra le dimensioni perché esiste solo un valore di Team per ciascuna combinazione di SalesGroup e Amount.
`Concat (DISTINCT Team,', ')`	Beta, Gamma perché il qualificatore DISTINCT significa che il risultato Gamma duplicato viene ignorato. Inoltre, l'argomento del delimitatore viene definito come virgola seguita da uno spazio.
`Concat (TOTAL <SalesGroup> Team)`	Tutti i valori di stringa per tutti i valori di Team sono concatenati se viene utilizzato il qualificatore TOTAL. Con la selezione del campo <SalesGroup> specificata, vengono suddivisi i risultati nei due valori della dimensione SalesGroup. Per SalesGroupEast, i risultati sono AlphaBetaDeltaGammaGamma. Per SalesGroupWest, i risultati sono EpsilonEtaThetaZeta.
`Concat (TOTAL <SalesGroup> Team,';', Amount)`	Aggiungendo l'argomento per **sort-weight**: Amount, i risultati vengono ordinati in base al valore della dimensione Amount. Il risultato diventa DeltaBetaGammaGammaAlpha e EtaEpsilonZEtaTheta.

Dati utilizzati nell'esempio:

```
TeamData:
LOAD * inline [
SalesGroup|Team|Date|Amount
East|Gamma|01/05/2013|20000
East|Gamma|02/05/2013|20000
West|Zeta|01/06/2013|19000
East|Alpha|01/07/2013|25000
East|Delta|01/08/2013|14000
West|Epsilon|01/09/2013|17000
West|Eta|01/10/2013|14000
East|Beta|01/11/2013|20000
West|Theta|01/12/2013|23000
] (delimiter is '|');
```

FirstValue

FirstValue() restituisce il valore che è stato caricato per primo dai record definiti dall'espressione, ordinato in base a una clausola **group by**.

Questa funzione è disponibile solo come funzione di script.

Sintassi:

```
FirstValue ( expr)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.

Limiti:

Se non viene trovato nessun valore di testo, viene restituito NULL.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato	Risultati su un foglio
TeamData: LOAD * inline [SalesGroup|Team|Date|Amount East|Gamma|01/05/2013|20000 East|Gamma|02/05/2013|20000 West|Zeta|01/06/2013|19000 East|Alpha|01/07/2013|25000 East|Delta|01/08/2013|14000 West|Epsilon|01/09/2013|17000 West|Eta|01/10/2013|14000 East|Beta|01/11/2013|20000 West|Theta|01/12/2013|23000] (delimiter is '|'); FirstValue1: LOAD SalesGroup,FirstValue(Team) as FirstTeamLoaded Resident TeamData Group By SalesGroup;	SalesGroup East West	FirstTeamLoaded Gamma Zeta

LastValue

LastValue() restituisce il valore che è stato caricato per ultimo dai record definiti dall'espressione, ordinato in base a una clausola **group by**.

Questa funzione è disponibile solo come funzione di script.

Sintassi:

```
LastValue ( expr )
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.

Limiti:

Se non viene trovato nessun valore di testo, viene restituito NULL.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

Per ottenere lo stesso aspetto della colonna dei risultati mostrata di seguito, nel pannello delle proprietà, in Ordinamento passare da Automatico a Personalizza, quindi deselezionare l'ordinamento numerico e alfabetico.

Esempio	Risultato	Risultato con ordinamento personalizzato
TeamData: LOAD * inline [SalesGroup|Team|Date|Amount East|Gamma|01/05/2013|20000 East|Gamma|02/05/2013|20000 West|Zeta|01/06/2013|19000 East|Alpha|01/07/2013|25000 East|Delta|01/08/2013|14000 West|Epsilon|01/09/2013|17000 West|Eta|01/10/2013|14000 East|Beta|01/11/2013|20000 West|Theta|01/12/2013|23000] (delimiter is '|'); LastValue1: LOAD SalesGroup,LastValue(Team) as LastTeamLoaded Resident TeamData Group By SalesGroup;	SalesGroup East West	LastTeamLoaded Beta Theta

MaxString

MaxString() individua i valori di stringa nell'espressione e restituisce l'ultimo valore di testo ordinato su un insieme di record, come definito dalla clausola **group by**.

Sintassi:

```
MaxString ( expr )
```

Tipo di dati restituiti: duale

Argomenti:

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.

Limiti:

Se non viene trovato nessun valore di testo, viene restituito NULL.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Esempio	Risultato	
TeamData: LOAD * inline [SalesGroup|Team|Date|Amount East|Gamma|01/05/2013|20000 East|Gamma|02/05/2013|20000 West|Zeta|01/06/2013|19000 East|Alpha|01/07/2013|25000 East|Delta|01/08/2013|14000 West|Epsilon|01/09/2013|17000 West|Eta|01/10/2013|14000 East|Beta|01/11/2013|20000 West|Theta|01/12/2013|23000] (delimiter is '|'); Concat1: LOAD SalesGroup,MaxString(Team) as MaxString1 Resident TeamData Group By SalesGroup;	SalesGroup East West	MaxString1 Gamma Zeta
Presupponendo che la tabella **TeamData** venga caricata come nell'esempio precedente e che lo script di caricamento dei dati presenti l'istruzione SET: SET DateFormat='DD/MM/YYYY';': LOAD SalesGroup,MaxString(Date) as MaxString2 Resident TeamData Group By SalesGroup;	SalesGroup East West	MaxString2 01/11/2013 01/12/2013

MaxString - funzione per grafici

MaxString() trova valori di stringa nell'espressione o nel campo e restituisce l'ultimo valore di testo nel criterio di ordinamento alfabetico.

Sintassi:

```
MaxString({[SetExpression] [TOTAL [<fld{, fld}>]]} expr)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Limiti:

Se l'espressione non include alcun valore con una rappresentazione di stringa, verrà restituito NULL.

Esempi e risultati:

Tabella dei risultati

SalesGroup	Amount	MaxString(Team)	MaxString(Date)
East	14000	Delta	2013/08/01
East	20000	Gamma	2013/11/01
East	25000	Alpha	2013/07/01
West	14000	Eta	2013/10/01
West	17000	Epsilon	2013/09/01
West	19000	Zeta	2013/06/01
West	23000	Theta	2013/12/01

Esempi di funzioni

Esempio	Risultato
`MaxString (Team)`	Vi sono tre valori di 20000 per la dimensione Amount: due di Gamma (in date differenti), e uno di Beta. Il risultato della misura MaxString (Team) è pertanto Gamma, perché è il valore più elevato nelle stringhe ordinate.
`MaxString (Date)`	2013/11/01 è il valore Date più elevato dei tre associati alla dimensione Amount. Questo presuppone che lo script includa l'istruzione SET `SET DateFormat='YYYY-MM-DD';'`

Dati utilizzati nell'esempio:

```
TeamData:
LOAD * inline [
SalesGroup|Team|Date|Amount
East|Gamma|01/05/2013|20000
East|Gamma|02/05/2013|20000
West|Zeta|01/06/2013|19000
East|Alpha|01/07/2013|25000
East|Delta|01/08/2013|14000
West|Epsilon|01/09/2013|17000
West|Eta|01/10/2013|14000
East|Beta|01/11/2013|20000
West|Theta|01/12/2013|23000
] (delimiter is '|');
```

MinString

MinString() individua i valori di stringa nell'espressione e restituisce il primo valore di testo ordinato su un insieme di record, come definito dalla clausola **group by**.

Sintassi:

```
MinString ( expr )
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.

Limiti:

Se non viene trovato nessun valore di testo, viene restituito NULL.

Esempi e risultati:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

Dati risultanti

Esempio	Risultato	
TeamData: LOAD * inline [SalesGroup|Team|Date|Amount East|Gamma|01/05/2013|20000 East|Gamma|02/05/2013|20000 West|Zeta|01/06/2013|19000 East|Alpha|01/07/2013|25000 East|Delta|01/08/2013|14000 West|Epsilon|01/09/2013|17000 West|Eta|01/10/2013|14000 East|Beta|01/11/2013|20000 West|Theta|01/12/2013|23000] (delimiter is '|'); Concat1: LOAD SalesGroup,MinString(Team) as MinString1 Resident TeamData Group By SalesGroup;	SalesGroup East West	MinString1 Alpha Epsilon
Presupponendo che la tabella **TeamData** venga caricata come nell'esempio precedente e che lo script di caricamento dei dati presenti l'istruzione SET: SET DateFormat='DD/MM/YYYY';': LOAD SalesGroup,MinString(Date) as MinString2 Resident TeamData Group By SalesGroup;	SalesGroup East West	MinString2 01/05/2013 01/06/2013

MinString - funzione per grafici

MinString() trova valori di stringa nell'espressione o nel campo e restituisce il primo valore di testo nel criterio di ordinamento alfabetico.

Sintassi:

```
MinString({[SetExpression] [TOTAL [<fld {, fld}>]]} expr)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.

Argomento	Descrizione
TOTAL	Se la parola **TOTAL** viene riportata prima degli argomenti della funzione, il calcolo verrà effettuato su tutti i valori possibili dati dalle selezioni correnti e non solo su quelli relativi al valore dimensionale attuale, vale a dire che verranno ignorate le dimensioni del grafico. Utilizzando **TOTAL [<fld {.fld}>]**, dove il qualificatore **TOTAL** è seguito da un elenco di uno o più nomi di campo come sottogruppo delle variabili di dimensione del grafico, si crea un sottogruppo di tutti i valori possibili.

Esempi e risultati:

Dati campione

SalesGroup	Amount	MinString(Team)	MinString(Date)
East	14000	Delta	2013/08/01
East	20000	Beta	2013/05/01
East	25000	Alpha	2013/07/01
West	14000	Eta	2013/10/01
West	17000	Epsilon	2013/09/01
West	19000	Zeta	2013/06/01
West	23000	Theta	2013/12/01

Esempi di funzioni

Esempi	Risultati
`MinString (Team)`	Vi sono tre valori di 20000 per la dimensione Amount: due di Gamma (in date differenti), e uno di Beta. Il risultato della misura MinString (Team) è pertanto Beta, perché questo è il primo valore nelle stringhe ordinate.
`MinString (Date)`	2013/11/01 è il primo valore Date dei tre valori associati alla dimensione Amount. Questo presuppone che lo script includa l'istruzione SET `SET DateFormat='YYYY-MM-DD';'`

Dati utilizzati nell'esempio:

```
TeamData:
LOAD * inline [
SalesGroup|Team|Date|Amount
East|Gamma|01/05/2013|20000
East|Gamma|02/05/2013|20000
West|Zeta|01/06/2013|19000
East|Alpha|01/07/2013|25000
East|Delta|01/08/2013|14000
West|Epsilon|01/09/2013|17000
West|Eta|01/10/2013|14000
East|Beta|01/11/2013|20000
```

```
West|Theta|01/12/2013|23000
] (delimiter is '|');
```

Funzioni di dimensione sintetica

Una dimensione sintetica viene creata nell'app a partire dai valori generati dalle funzioni di dimensione sintetica e non direttamente dai campi nel modello dati. Quando in un grafico i valori generati da una funzione di dimensione sintetica vengono utilizzati come dimensione calcolata, viene creata una dimensione sintetica. Le dimensioni sintetiche consentono, ad esempio, di creare grafici con dimensioni con valori derivanti dai dati, vale a dire dimensioni dinamiche.

Le dimensioni sintetiche non vengono influenzate dalle selezioni.

Le seguenti funzioni di dimensione sintetica possono essere utilizzate nei grafici.

ValueList

ValueList() restituisce una serie di valori elencati che, se utilizzati in una dimensione calcolata, formano una dimensione sintetica.

```
ValueList - funzione per grafici (v1 {, Expression})
```

ValueLoop

ValueLoop() restituisce un set di valori ripetuti che, se utilizzati in una dimensione calcolata, formano una dimensione sintetica.

```
ValueLoop - funzione per grafici(from [, to [, step ]])
```

ValueList - funzione per grafici

ValueList() restituisce una serie di valori elencati che, se utilizzati in una dimensione calcolata, formano una dimensione sintetica.

*Nei grafici con una dimensione sintetica creata con la funzione **ValueList**, è possibile fare riferimento al valore di dimensione corrispondente a una specifica cella di espressione dichiarando nuovamente la funzione **ValueList** con gli stessi parametri nell'espressione grafica. La funzione può essere ovviamente utilizzata ovunque nel layout, ma, tranne quando viene utilizzata per le dimensioni sintetiche, avrà significato solamente all'interno di una funzione di aggregazione.*

Le dimensioni sintetiche non vengono influenzate dalle selezioni.

Sintassi:

```
ValueList(v1 {,...})
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
v1	Valore statico (generalmente una stringa, anche se può essere un numero).
{,...}	Elenco opzionale di valori statici.

Esempi e risultati:

Esempi di funzioni

Esempio	Risultato
`ValueList ('Number of Orders', 'Average Order Size', 'Total Amount')`	Se utilizzato per creare una dimensione in una tabella, ad esempio, i tre valori della stringa verranno utilizzati come etichette della riga nella tabella, a cui è possibile fare riferimento in un'espressione.
`=IF( ValueList ('Number of Orders', 'Average Order Size', 'Total Amount') = 'Number of Orders', count (SaleID), IF( ValueList ('Number of Orders', 'Average Order Size', 'Total Amount') = 'Average Order Size', avg (Amount), sum (Amount) ))`	Questa espressione prende i valori dalla dimensione creata e vi fa riferimento in un'istruzione IF nidificata come input per tre funzioni di aggregazione: **ValueList()** Created dimension / Year / Added expression 522.00 Number of Orders 2012 5.00 Number of Orders 2013 7.00 Average Order Size 2012 13.20 Average Order Size 2013 15.43 Total Amount 2012 66.00 Total Amount 2013 108.00

Dati utilizzati negli esempi:

```
SalesPeople:
LOAD * INLINE [
SaleID|SalesPerson|Amount|Year
1|1|12|2013
2|1|23|2013
3|1|17|2013
4|2|9|2013
5|2|14|2013
6|2|29|2013
```

```
7|2|4|2013
8|1|15|2012
9|1|16|2012
10|2|11|2012
11|2|17|2012
12|2|7|2012
] (delimiter is '|');
```

ValueLoop - funzione per grafici

ValueLoop() restituisce un set di valori ripetuti che, se utilizzati in una dimensione calcolata, formano una dimensione sintetica.
I valori generati iniziano con il valore **from** e finiscono con il valore **to** includendo i valori intermedi con incrementi di step.

*Nei grafici con una dimensione sintetica creata con la funzione **ValueLoop**, è possibile fare riferimento al valore di dimensione corrispondente a una specifica cella di espressione dichiarando nuovamente la funzione **ValueLoop** con gli stessi parametri nell'espressione grafica. La funzione può essere ovviamente utilizzata ovunque nel layout, ma, tranne quando viene utilizzata per le dimensioni sintetiche, avrà significato solamente all'interno di una funzione di aggregazione.*

Le dimensioni sintetiche non vengono influenzate dalle selezioni.

Sintassi:

```
ValueLoop(from [, to [, step ]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomenti	Descrizione
from	Valore iniziale nel set di valori da generare.
to	Valore finale nel set di valori da generare.
step	Grandezza dell'incremento tra i valori.

Esempi e risultati:

Esempi di funzioni

Esempio	Risultato
`ValueLoop (1, 10)`	Ciò crea una dimensione in una tabella che, ad esempio, può essere utilizzata per l'etichettatura numerata. Questo esempio restituisce valori numerati da 1 a 10. È possibile fare riferimento a questi valori in un'espressione.

Esempio	Risultato
`ValueLoop (2, 10,2)`	Questo esempio restituisce valori numerati 2, 4, 6, 8 e 10 perché l'argomento step presenta un valore di 2.

Aggregazioni nidificate

Si potrebbero verificare situazioni in cui è necessario applicare un'aggregazione al risultato di un'altra aggregazione. Questa operazione è denominata nidificazione delle applicazioni.

Non è possibile nidificare le aggregazioni in gran parte delle espressioni del grafico. Tuttavia, è possibile nidificare le aggregazioni se si utilizza il qualificatore **TOTAL** nella funzione di aggregazione interna.

È consentito un massimo di 100 livelli di nidificazione.

Aggregazioni nidificate con il qualificatore TOTAL

Esempio:

Si desidera calcolare la somma del campo **Sales**, includendo solo le transazioni con **OrderDate** uguali all'anno scorso. L'anno precedente può essere ottenuto tramite la funzione di aggregazione **`Max(TOTAL`** `Year(OrderDate)`**`)`**.

La seguente aggregazione restituirebbe il risultato desiderato:

```
Sum(If(Year(OrderDate)=Max(TOTAL Year(OrderDate)), Sales))
```

Qlik Sense richiede l'inclusione del qualificatore **TOTAL** in questo tipo di nidificazione. È necessario per il confronto desiderato. Questo tipo di nidificazione è abbastanza comune e dovrebbe essere utilizzata quando richiesto.

Vedere anche:

p *Aggr - funzione per grafici (page 541)*

5.3 Aggr - funzione per grafici

Aggr() restituisce una matrice di valori per l'espressione calcolata in base alla dimensione o alle dimensioni dichiarate. Ad esempio, il valore massimo delle vendite, per cliente, per regione.

La funzione **Aggr** è utilizzata per le aggregazioni nidificate, in cui il relativo primo parametro (l'aggregazione interna) è calcolato una volta per valore dimensionale. Le dimensioni sono specificate nel secondo parametro (e nei parametri successivi).

Inoltre, la funzione **Aggr** deve essere racchiusa in una funzione di aggregazione esterna, utilizzando la gamma di risultati dalla funzione **Aggr** come input per l'aggregazione in cui risulta nidificata.

Sintassi:

```
Aggr({SetExpression}[DISTINCT] [NODISTINCT ] expr, StructuredParameter{, StructuredParameter})
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	Un'espressione costituita da una funzione di aggregazione. Per impostazione predefinita, la funzione di aggregazione aggrega la serie dei possibili record definiti dalla selezione.
StructuredParameter	StructuredParameter è costituito da una dimensione e, facoltativamente, da criteri di ordinamento nel formato: `(Dimension(Sort-type, Ordering))` La dimensione è un campo singolo e non può essere un'espressione. Viene utilizzata per determinare la matrice di valori per cui viene calcolata l'espressione Aggr. Se sono inclusi criteri di ordinamento, la matrice di valori creata dalla funzione Aggr, calcolata per la dimensione, verrà ordinata. Questo è importante quando l'ordinamento influisce sul risultato dell'espressione in cui è racchiusa la funzione Aggr. Per informazioni dettagliate su come utilizzare i criteri di ordinamento, vedere Aggiunta di criteri di ordinamento alla dimensione nel parametro strutturato.
SetExpression	Per impostazione predefinita, la funzione di aggregazione aggrega la serie di possibili record definiti dalla selezione. È possibile definire una serie di record alternativa mediante un'espressione Set Analysis.
DISTINCT	Se l'argomento dell'espressione è preceduto dal qualificatore **distinct** o se non viene utilizzato alcun qualificatore, ogni combinazione distinta di valori di dimensione restituirà un solo valore. Questa è la modalità normale in cui vengono create le aggregazioni; ciascuna combinazione distinta di valori di dimensione restituisce una linea nel grafico.
NODISTINCT	Se l'argomento dell'espressione è preceduto dal qualificatore **nodistinct**, ogni combinazione di valori di dimensione può restituire più valori, a seconda della struttura dati sottostante. Se esiste una sola dimensione, la funzione **aggr** restituirà una matrice con un numero di elementi uguale al numero di righe presenti nei dati sorgente.

Le funzioni di aggregazione di base, quali **Sum**, **Min** e **Avg**, restituiscono un singolo valore numerico, mentre la funzione Aggr() può essere paragonata alla creazione di una serie di risultati temporanei (una tabella virtuale) su cui è possibile effettuare un'altra aggregazione. Ad esempio, è possibile calcolare il valore medio delle vendite sommando le vendite per cliente in un'istruzione **Aggr()** e calcolando quindi la media dei risultati sommati: **Avg(TOTAL Aggr(Sum(Sales),Customer))**.

È possibile utilizzare la funzione Aggr() nelle dimensioni calcolate se si desidera creare aggregazioni di grafici nidificate su più livelli.

Limiti:

Ogni dimensione presente in una funzione Aggr() deve corrispondere a un campo singolo e non può essere un'espressione (dimensione calcolata).

Aggiunta di criteri di ordinamento alla dimensione nel parametro strutturato

Nella sua forma di base, l'argomento StructuredParameter nella sintassi della funzione Aggr è una dimensione singola. L'espressione Aggr(Sum(Sales, Month)) trova il valore totale delle vendite per ogni mese. Tuttavia, se viene inclusa in un'altra funzione di aggregazione, potrebbero venire restituiti risultati imprevisti, a meno che non vengano utilizzati i criteri di ordinamento. Ciò è dovuto al fatto che alcune dimensioni possono essere ordinate con criterio numerico o con criterio alfabetico e così via.

Nell'argomento StructuredParameter all'interno della funzione Aggr è possibile specificare criteri di ordinamento per la dimensione nell'espressione. In questo modo si impone un ordinamento alla tabella virtuale prodotta dalla funzione Aggr.

La sintassi dell'argomento StructuredParameter è la seguente:

```
(FieldName, (Sort-type, Ordering))
```

I parametri strutturati possono essere nidificati:

```
(FieldName, (FieldName2, (Sort-type, Ordering)))
```

Sort-type può essere: NUMERIC, TEXT, FREQUENCY o LOAD_ORDER.

I tipi di ordinamento associati a ciascun Sort-type sono i seguenti:

Tipi di ordinamento consentiti

Sort-type	Valori di Ordering consentiti
NUMERIC	ASCENDING, DESCENDING o REVERSE
TEXT	ASCENDING, A2Z, DESCENDING, REVERSE o Z2A
FREQUENCY	DESCENDING, REVERSE o ASCENDING
LOAD_ORDER	ASCENDING, ORIGINAL, DESCENDING o REVERSE

I tipi di ordinamento REVERSE e DESCENDING sono equivalenti.

Per Sort-type uguale a TEXT, i tipi di ordinamento ASCENDING e A2Z sono equivalenti e DESCENDING, REVERSE e Z2A sono equivalenti.

Per Sort-type uguale a LOAD_ORDER, i tipi di ordinamento ASCENDING e ORIGINAL sono equivalenti.

Esempi: Espressioni del grafico mediante Aggr

Esempi - espressioni del grafico

Esempio 1 espressione del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare l'esempio di espressione del grafico in basso.

```
ProductData:  LOAD * inline [ Customer|Product|UnitSales|UnitPrice Astrida|AA|4|16
Astrida|AA|10|15 Astrida|BB|9|9 Betacab|BB|5|10 Betacab|CC|2|20 Betacab|DD|25|25
Canutility|AA|8|15 Canutility|CC|0|19 ] (delimiter is '|');
```

Espressione del grafico

Creare una visualizzazione KPI in un foglio Qlik Sense. Aggiungere l'espressione seguente al KPI, come misura:

```
Avg(Aggr(Sum(UnitSales*UnitPrice), Customer))
```

Risultato

376.7

Spiegazione

L'espressione `Aggr(Sum(UnitSales*UnitPrice), Customer)` trova il valore totale delle vendite per **Customer** e restituisce una matrice di valori: 295, 715 e 120 per i tre valori di **Customer**.

In modo efficiente, è stato creato un elenco temporaneo di valori, senza una tabella esplicita o una colonna contenente tali valori.

Questi valori vengono utilizzati come input nella funzione **Avg()** per trovare il valore medio delle vendite, 376.7.

Esempio 2 espressione del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare l'esempio di espressione del grafico in basso.

```
ProductData:  LOAD * inline [ Customer|Product|UnitSales|UnitPrice Astrida|AA|4|16
Astrida|AA|10|15 Astrida|BB|10|15 Astrida|BB|9|9 Betacab|BB|5|10 Betacab|BB|7|12
Betacab|CC|2|22 Betacab|CC|4|20 Betacab|DD|25|25 Canutility|AA|8|15 Canutility|AA|5|11
Canutility|CC|0|19 ] (delimiter is '|');
```

Espressione del grafico

Creare una visualizzazione tabella in un foglio Qlik Sense con **Customer**, **Product**, **UnitPrice** e **UnitSales** come dimensioni. Aggiungere la seguente espressione alla tabella come misura:

```
Aggr(NODISTINCT Max(UnitPrice), Customer, Product)
```

Risultato

Customer	Product	UnitPrice	UnitSales	Aggr(NODISTINCT Max(UnitPrice), Customer, Product)
Astrida	AA	15	10	16
Astrida	AA	16	4	16
Astrida	BB	9	9	15
Astrida	BB	15	10	15
Betacab	BB	10	5	12
Betacab	BB	12	7	12
Betacab	CC	20	4	22
Betacab	CC	22	2	22
Betacab	DD	25	25	25
Canutility	AA	11	5	15
Canutility	AA	15	8	15
Canutility	CC	19	0	19

Spiegazione

Una matrice di valori: 16, 16, 15, 15, 12, 12, 22, 22, 25, 15, 15 e 19. Il qualificatore **nodistinct** indica che la matrice contiene un elemento per ogni riga dei dati sorgente: ognuno è il valore **UnitPrice** massimo per ogni valore **Customer** e **Product**.

Esempio 3 espressione del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare l'esempio di espressione del grafico in basso.

```
Set vNumberOfOrders = 1000;  OrderLines: Load   RowNo() as OrderLineID,         OrderID,        OrderDate,
Round((Year(OrderDate)-2005)*1000*Rand()*Rand()*Rand1) as Sales         While Rand()<=0.5 or IterNo
()=1; Load * Where OrderDate<=Today(); Load     Rand() as Rand1,        Date(MakeDate(2013)+Floor
((365*4+1)*Rand())) as OrderDate,       RecNo() as OrderID      Autogenerate vNumberOfOrders;
Calendar: Load distinct         Year(OrderDate) as Year,        Month(OrderDate) as Month,      OrderDate
Resident OrderLines;
```

Espressioni del grafico

Crear una visualizzazione tabella in un foglio Qlik Sense con **Anno** e **Mese** come dimensioni. Aggiungere alla tabella le espressioni seguenti come misure:

- `Sum(Sales)`
- `Sum(Aggr( Rangesum(Above(Sum(Sales),0,12)), (Year, (Numeric, Ascending)), (Month, (Numeric, Ascending)) ))` etichettato come Structured Aggr() nella tabella.

Risultato

Year	Month	Sum(Sales)	Structured Aggr()
2013	Jan	53495	53495
2013	Feb	48580	102075
2013	Mar	25651	127726
2013	Apr	36585	164311
2013	May	61211	225522
2013	Jun	23689	249211
2013	Jul	42311	291522
2013	Aug	41913	333435
2013	Sep	28886	362361
2013	Oct	25977	388298
2013	Nov	44455	432753
2013	Dec	64144	496897
2014	Jan	67775	67775

Spiegazione

Questo esempio visualizza i valori aggregati su un periodo di dodici mesi per ogni anno in ordine cronologico crescente, da cui i parametri strutturati (Numeric, Ascending) parte dell'espressione **Aggr()**. Sono richieste due dimensioni specifiche come parametri strutturati: **Anno** e **Mese**, ordinato (1) **Anno** (numerico) e (2) **Mese** (numerico). Queste due dimensioni devono essere utilizzate nella visualizzazione tabella o grafico. Ciò risulta necessario affinché l'elenco dimensioni della funzione **Aggr()** corrisponda alle dimensioni dell'oggetto utilizzato nella visualizzazione.

È possibile confrontare le differenze tra queste misure in una tabella o in grafici lineari separati:

- `Sum(Aggr( Rangesum(Above(Sum(Sales),0,12)), (Year), (Month) ))`
- `Sum(Aggr( Rangesum(Above(Sum(Sales),0,12)), (Year, (Numeric, Ascending)), (Month, (Numeric, Ascending)) ))`

Dovrebbe risultare evidente che solo l'ultima espressione esegue l'accumulo desiderato dei valori aggregati.

Vedere anche:

p *Funzioni di aggregazione di base (page 329)*

5.4 Funzioni colore

Queste funzioni possono essere utilizzate nelle espressioni associate con l'impostazione e la valutazione delle proprietà del colore degli oggetti dei grafici così come negli script di caricamento dei dati.

Qlik Sense supporta le funzioni colore ***Color()****,* ***qliktechblue*** *e* ***qliktechgray*** *per compatibilità con le versioni precedenti, sebbene il loro utilizzo non sia consigliato.*

ARGB

ARGB() viene utilizzata nelle espressioni per impostare o valutare le proprietà del colore di un oggetto del grafico, in cui il colore è definito da un componente rosso **r**, un componente verde **g** e un componente blu **b**, con un fattore alfa (opacità) di **alpha**.

```
ARGB (alpha, r, g, b)
```

HSL

HSL() viene utilizzata nelle espressioni per impostare o valutare le proprietà del colore di un oggetto del grafico, dove il colore è definito dai valori **hue**, **saturation** e **luminosity** tra 0 e 1.

```
HSL  (hue, saturation, luminosity)
```

RGB

RGB() restituisce un intero corrispondente al codice colore del colore definito dai tre parametri: la componente rossa r, la componente verde g e la componente blu b. Queste componenti devono avere valori interi compresi tra 0 e 255. La funzione può essere usata nelle espressioni per impostare o valutare le proprietà di colore di un oggetto grafico.

```
RGB  (r, g, b)
```

Colormix1

La funzione **Colormix1()** viene utilizzata nelle espressioni per restituire una rappresentazione cromatica ARGB da un gradiente di due colori, basato su un valore compreso tra 0 e 1.

```
Colormix1 (Value , ColorZero , ColorOne)
```

Value è un numero reale compreso tra 0 e 1.

- Se Value = 0, viene restituito ColorZero .
- Se Value = 1, viene restituito ColorOne .
- Se 0 < Value < 1 verrà restituita la sfumatura intermedia appropriata.

ColorZero è una rappresentazione di colore RGB valida per il colore da associare con il limite minimo dell'intervallo.

ColorOne è una rappresentazione cromatica RGB valida per il colore da associare con il livello finale massimo dell'intervallo.

Esempio:

```
Colormix1(0.5, red(), blue())
```
restituisce:

```
ARGB(255,64,0,64) (purple)
```

Colormix2

La funzione **Colormix2()** viene utilizzata nelle espressioni per restituire una rappresentazione cromatica ARGB da un gradiente di due colori, basato su un valore compreso tra -1 e 1 con la possibilità di specificare un colore intermedio per la posizione centrale (0).

```
Colormix2 (Value ,ColorMinusOne , ColorOne[ , ColorZero])
```
Value è un numero reale compreso tra -1 e 1.

- Se Value = -1 verrà restituito il primo colore.
- Se Value = 1 verrà restituito il secondo colore.
- Se -1 < Value < 1, verrà restituita la combinazione di colori appropriata.

ColorMinusOne è una rappresentazione di colore RGB valida per il colore da associare con il limite minimo dell'intervallo.

ColorOne è una rappresentazione cromatica RGB valida per il colore da associare con il livello finale massimo dell'intervallo.

ColorZero è una rappresentazione cromatica RGB opzionale valida per il colore da associare con il centro dell'intervallo.

SysColor

SysColor() restituisce la rappresentazione cromatica ARGB per il colore di sistema di Windows nr, dove nr corrisponde al parametro della funzione **GetSysColor(nr)** dell'API di Windows.

```
SysColor (nr)
```

ColorMapHue

ColorMapHue() restituisce il valore ARGB di un colore da una mappa dei colori che varia il componente di tonalità del modello cromatico HSV. La mappa dei colori inizia con il rosso, quindi passa al giallo, verde, ciano, blu, magenta per poi tornare al rosso. x deve essere specificato come un valore compreso tra 0 e 1.

```
ColorMapHue (x)
```

ColorMapJet

ColorMapJet() restituisce un valore ARGB di un colore da una mappa dei colori che inizia con il blu, passando dal ciano, giallo e arancione per poi tornare al rosso. x deve essere specificato come un valore compreso tra 0 e 1.

```
ColorMapJet (x)
```

Funzioni colori predefiniti

Per i colori predefiniti nelle espressioni è possibile utilizzare le funzioni descritte di seguito. Ciascuna funzione restituisce una rappresentazione cromatica RGB.

In alternativa, è possibile specificare un parametro per il fattore alfa, nel qual caso verrà restituita la rappresentazione cromatica ARGB. Un fattore alfa pari a 0 corrisponde alla trasparenza completa, mentre un fattore alfa pari a 255 corrisponde all'opacità completa. Se non viene immesso un valore per alfa, verrà utilizzato 255.

Funzioni colori predefiniti

Funzione colore	RGB Valore
black ([alpha])	(0,0,0)
blue([alpha])	(0,0,128)
brown([alpha])	(128,128,0)
cyan([alpha])	(0,128,128)
darkgray([alpha])	(128,128,128)
green([alpha])	(0,128,0)
lightblue([alpha])	(0,0,255)
lightcyan([alpha])	(0,255,255)
lightgray([alpha])	(192,192,192)
lightgreen([alpha])	(0,255,0)
lightmagenta([alpha])	(255,0,255)
lightred([alpha])	(255,0,0)
magenta([alpha])	(128,0,128)
red([alpha])	(128,0,0)
white([alpha])	(255,255,255)
yellow([alpha])	(255,255,0)

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Blue()`	RGB(0,0,128)
`Blue(128)`	ARGB(128,0,0,128)

ARGB

ARGB() viene utilizzata nelle espressioni per impostare o valutare le proprietà del colore di un oggetto del grafico, in cui il colore è definito da un componente rosso **r**, un componente verde **g** e un componente blu **b**, con un fattore alfa (opacità) di **alpha**.

Sintassi:

```
ARGB(alpha, r, g, b)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
alpha	Valore della trasparenza nella scala 0-255. 0 corrisponde alla trasparenza completa e 255 corrisponde all'opacità completa.
r, g, b	Valori del componente rosso, verde e blu. Un componente a colori pari a 0 corrisponde a nessun contributo e uno pari a 255 al contributo completo.

Tutti gli argomenti devono essere espressioni che si risolvono in numeri interi nella scala da 0 a 255.

Se si interpreta il componente numerico e lo si formatta in notazione esadecimale, i valori dei componenti a colori saranno più facilmente visibili. Ad esempio, il verde chiaro ha il numero 4 278 255 360, che in notazione esadecimale è FF00FF00. Le prime due posizioni 'FF' (255) indicano il canale **alpha**. Le due posizioni successive '00' indicano la quantità di **red**, le due posizioni successive 'FF' indicano la quantità di **green** e le sue posizioni finali '00' indicano la quantità di **blue**.

RGB

RGB() restituisce un intero corrispondente al codice colore del colore definito dai tre parametri: la componente rossa r, la componente verde g e la componente blu b. Queste componenti devono avere valori interi compresi tra 0 e 255. La funzione può essere usata nelle espressioni per impostare o valutare le proprietà di colore di un oggetto grafico.

Sintassi:

```
RGB (r, g, b)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
r, g, b	Valori del componente rosso, verde e blu. Un componente a colori pari a 0 corrisponde a nessun contributo e uno pari a 255 al contributo completo.

Tutti gli argomenti devono essere espressioni che si risolvono in numeri interi nella scala da 0 a 255.

Se si interpreta il componente numerico e lo si formatta in notazione esadecimale, i valori dei componenti a colori saranno più facilmente visibili. Ad esempio, il verde chiaro ha il numero 4 278 255 360, che in notazione esadecimale è FF00FF00. Le prime due posizioni 'FF' (255) indicano il canale **alpha**. Nelle funzioni **RGB** e **HSL**, è sempre 'FF' (opaco). Le due posizioni successive '00' indicano la quantità di **red**, le due posizioni successive 'FF' indicano la quantità di **green** e le sue posizioni finali '00' indicano la quantità di **blue**.

Esempio: Espressione del grafico
Questo esempio applica un colore personalizzato a un grafico:

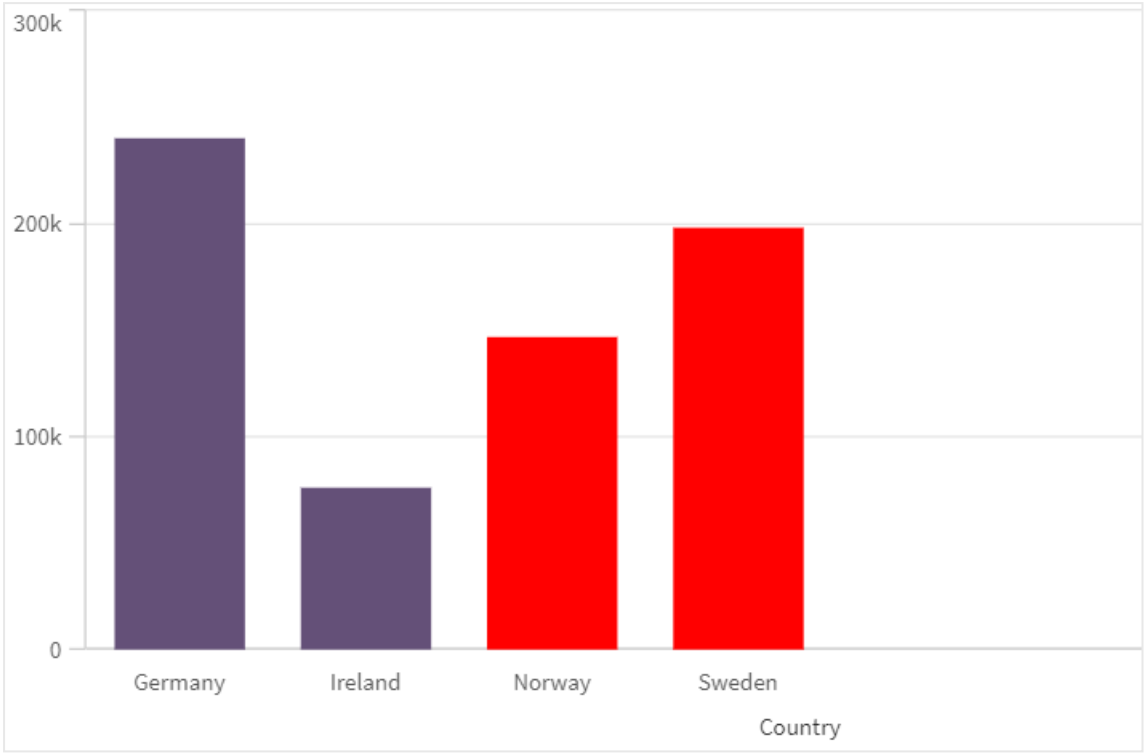
Dati utilizzati in questo esempio:

```
ProductSales: Load * Inline [Country,Sales,Budget Sweden,100000,50000 Germany, 125000, 175000 Norway, 74850, 68500 Ireland, 45000, 48000 Sweden,98000,50000 Germany, 115000, 175000 Norway, 71850, 68500 Ireland, 31000, 48000 ] (delimiter is ',');
```

Inserire l'espressione seguente nel pannello proprietà **Colori e legenda**:

```
If (Sum(Sales)>Sum(Budget),RGB(255,0,0),RGB(100,80,120))
```

Risultato:



Esempio: Script di caricamento

L'esempio seguente mostra i valori RGB equivalenti per i valori nel formato esadecimale:

```
Load Text(R & G & B) as Text, RGB(R,G,B)      as Color; Load Num#(R,'(HEX)') as R, Num#(G,'(HEX)') as G, Num#(B,'(HEX)') as B Inline [R,G,B 01,02,03 AA,BB,CC];
```

Risultato:

Testo	Colore
010203	RGB(1,2,3)
AABBCC	RGB(170,187,204)

HSL

HSL() viene utilizzata nelle espressioni per impostare o valutare le proprietà del colore di un oggetto del grafico, dove il colore è definito dai valori **hue**, **saturation** e **luminosity** tra 0 e 1.

Sintassi:

```
HSL (hue, saturation, luminosity)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
hue, saturation, luminosity	I valori dei componenti hue, saturation e luminosity compresi tra 0 e 1.

Tutti gli argomenti devono essere espressioni che si risolvono in numeri interi nella scala da 0 a 1.

Se si interpreta il componente numerico e lo si formatta in notazione esadecimale, i valori RGB dei componenti a colori saranno più facilmente visibili. Ad esempio, il verde chiaro presenta il numero 4 278 255 360, che in notazione esadecimale è FF00FF00 e RGB (0,255,0). Ciò equivale a HSL (80/240, 240/240, 120/240) , vale a dire un valore HSL di (0.33, 1, 0.5).

5.5 Funzioni condizionali

Tutte le funzioni condizionali valutano una condizione e restituiscono differenti risposte in base al valore della condizione. Le funzioni possono essere utilizzate nello script di caricamento dei dati e nelle espressioni grafiche.

Panoramica sulle funzioni condizionali

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

alt

La funzione **alt** restituisce il primo dei parametri che presenta una rappresentazione numerica valida. Se nessuna corrispondenza viene trovata, verrà restituito l'ultimo parametro. Può essere utilizzato un numero qualsiasi di parametri.

```
alt (expr1[ , expr2 , expr3 , ...] , else)
```

class

La funzione **class** assegna il primo parametro a un intervallo di classe. Viene restituito un valore duale in cui a<=x<b rappresenta il valore testuale dove a e b rappresentano i limiti superiore e inferiore del contenitore e il limite inferiore è un valore numerico.

```
class (expression, interval [ , label [ , offset ]])
```

coalesce

La funzione **coalesce** restituisce il primo dei parametri che presenta una rappresentazione non-NULL valida. Può essere utilizzato un numero qualsiasi di parametri.

```
coalesce(expr1[ , expr2 , expr3 , ...])
```

if

La funzione **if** restituisce un valore a seconda che la condizione fornita con la funzione esegua la valutazione come True o come False.

```
if (condition , then , else)
```

match

La funzione **match** confronta il primo parametro con tutti i parametri seguenti e restituisce la posizione numerica delle espressioni corrispondenti. Il confronto rispetta la distinzione maiuscole/minuscole.

```
match ( str, expr1 [ , expr2,...exprN ])
```

mixmatch

La funzione **mixmatch** confronta il primo parametro con tutti i parametri seguenti e restituisce la posizione numerica delle espressioni corrispondenti. Il confronto non rispetta la distinzione maiuscole/minuscole.

```
mixmatch ( str, expr1 [ , expr2,...exprN ])
```

pick

La funzione pick restituisce l'espressione numero *n* nell'elenco.

```
pick (n, expr1[ , expr2,...exprN])
```

wildmatch

La funzione **wildmatch** confronta il primo parametro con tutti i parametri seguenti e restituisce il numero dell'espressione corrispondente. Consente l'utilizzo di caratteri jolly (***** e **?**) nelle stringhe di confronto. ***** corrisponde a qualsiasi sequenza di caratteri. **?** corrisponde a qualsiasi carattere singolo. Il confronto non rispetta la distinzione maiuscole/minuscole.

```
wildmatch ( str, expr1 [ , expr2,...exprN ])
```

alt

La funzione **alt** restituisce il primo dei parametri che presenta una rappresentazione numerica valida. Se nessuna corrispondenza viene trovata, verrà restituito l'ultimo parametro. Può essere utilizzato un numero qualsiasi di parametri.

Sintassi:

```
alt(expr1[ , expr2 , expr3 , ...] , else)
```

Argomenti:

Argomenti

Argomento	Descrizione
expr1	La prima espressione da controllare per una valida rappresentazione numerica.
expr2	La seconda espressione da controllare per una valida rappresentazione numerica.
expr3	La terza espressione da controllare per una valida rappresentazione numerica.
else	Valore restituito se nessuno dei parametri precedenti ha una valida rappresentazione numerica.

La funzione alt viene spesso utilizzata con le funzioni di interpretazione numerica o della data. In questo modo Qlik Sense può provare differenti formati di data in un ordine con priorità. Può anche essere utilizzata per gestire valori NULL in espressioni numeriche.

Esempi:

Esempi

Esempio	Risultato

`alt( date#( dat , 'YYYY/MM/DD' ),`
 	
`date#( dat , 'MM/DD/YYYY' ),`
 	
`date#( dat , 'MM/DD/YY' ),`
 	
`'No valid date' )`
	Questa espressione verificherà se la data del campo contiene una data conforme a uno dei tre formati specificati. In tal caso, restituirà un valore duale contenente la stringa originale e una rappresentazione numerica valida di una data. Se non viene trovata nessuna corrispondenza, verrà restituito il testo 'No valid date' (senza alcuna rappresentazione numerica valida).

`alt(Sales,0) + alt(Margin,0)`	L'espressione aggiunge i campi Sales e Margin, sostituendo qualsiasi valore mancante (NULL) con uno 0.

class

La funzione **class** assegna il primo parametro a un intervallo di classe. Viene restituito un valore duale in cui a<=x<b rappresenta il valore testuale dove a e b rappresentano i limiti superiore e inferiore del contenitore e il limite inferiore è un valore numerico.

Sintassi:

```
class(expression, interval [ , label [ , offset ]])
```

Argomenti:

Argomenti

Argomento	Descrizione
interval	Un numero che specifica la larghezza del contenitore.
label	Una stringa arbitraria che può sostituire la 'x' nel testo del risultato.
offset	Un numero che può essere utilizzato come offset dal punto di partenza predefinito della classificazione. Il punto di partenza predefinito è in genere 0.

Esempi:

Esempi

Esempio	Risultato
`class( var,10 )` con `var = 23`	restituisce '20<=x<30'
`class( var,5,'value' )` con `var = 23`	restituisce '20<= value <25'
`class( var,10,'x',5 )` con `var = 23`	restituisce '15<=x<25'

Esempio - Script di caricamento usando class

Esempio: script di caricamento

Script di caricamento

In questo esempio viene caricata una tabella contenente il nome e l'età delle persone. Si desidera aggiungere un campo che classifichi ciascun utente in base a un gruppo di età con un intervallo temporale di dieci anni. La tabella di origine originale ha un aspetto simile al seguente.

Risultati

Name	Age
John	25
Karen	42
Yoshi	53

Per aggiungere il campo di classificazione del gruppo di età, è possibile aggiungere un'istruzione preceding load tramite la funzione **class**.

Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Creare la tabella in basso in Qlik Sense per visualizzare i risultati.

```
 LOAD *, class(Age, 10, 'age') As Agegroup;  LOAD * INLINE [ Age, Name 25, John 42, Karen 53, Yoshi];
```

Risultati

Risultati

Name	Age	Agegroup
John	25	20 <= age < 30
Karen	42	40 <= age < 50
Yoshi	53	50 <= age < 60

coalesce

La funzione **coalesce** restituisce il primo dei parametri che presenta una rappresentazione non-NULL valida. Può essere utilizzato un numero qualsiasi di parametri.

Sintassi:

```
coalesce(expr1[ , expr2 , expr3 , ...])
```

Argomenti:

Argomenti

Argomento	Descrizione
expr1	La prima espressione da controllare con una rappresentazione non-NULL valida.
expr2	La seconda espressione da controllare con una rappresentazione non-NULL valida.
expr3	La terza espressione da controllare con una rappresentazione non-NULL valida.

Esempi:

Esempi

Esempio	Risultato
	Questa espressione modifica tutti i valori NULL di un campo a 'N/A'.
`Coalesce(ProductDescription, ProductName, ProductCode, 'no description available')`	Questa espressione selezionerà tra tre diversi campi di descrizione prodotti, utili quando alcuni campi possono non avere valori per il prodotto. Il primo dei campi, nell'ordine dato, verrà restituito con un valore non null. Se nessuno dei campi contiene un valore, il risultato sarà 'nessuna descrizione disponibile'.
`Coalesce(TextBetween(FileName, '"', '"'), FileName)`	Questa espressione rimuoverà potenziali virgolette di chiusura dal campo *FileName*. Se il *FileName* fornito appare tra virgolette, queste verranno rimosse e verrà restituito un *FileName* racchiuso, senza virgolette. Se la funzione *TextBetween* non trova i delimitatori restituisce null, che **Coalesce** rifiuterà, restituendo invece il valore raw *FileName*.

if

La funzione **if** restituisce un valore a seconda che la condizione fornita con la funzione esegua la valutazione come True o come False.

Sintassi:

```
if(condition , then [, else])
```

Argomenti

Argomento	Descrizione
condition	Espressione interpretata in modo logico.
then	Espressione che può essere di qualsiasi tipo. Se *condition* è True, quindi la funzione if restituisce il valore dell'espressione *then*.

Argomento	Descrizione
else	Espressione che può essere di qualsiasi tipo. Se *condition* è False, quindi la funzione if restituisce il valore dell'espressione *else*. Questo parametro è facoltativo. Se *condition* è False, verrà restituito NULL se non si è specificato else.

Esempio

Esempio	Risultato
`if( Amount>= 0, 'OK', 'Alarm' )`	Questa espressione verifica se l'importo è un numero positivo (0 o superiore) e restituisce 'OK' se lo è. Se l'importo è inferiore a 0, viene restituito 'Alarm'.

Esempio - Script di caricamento usando if

Esempio: script di caricamento

Script di caricamento

If può essere utilizzato in script di caricamento con altri metodi e oggetti, comprese le variabili. Ad esempio, se si imposta una variabile *threshold* (soglia) e si desidera includere nel modello dati un campo basato su tale soglia, è possibile usare il codice seguente.

Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Creare la tabella in basso in Qlik Sense per visualizzare i risultati.

```
Transactions:
Load * Inline [
transaction_id, transaction_date, transaction_amount, transaction_quantity, customer_id, size,
color_code
3750, 20180830, 23.56, 2, 2038593, L, Red
3751, 20180907, 556.31, 6, 203521, m, orange
3752, 20180916, 5.75, 1, 5646471, S, blue
3753, 20180922, 125.00, 7, 3036491, l, Black
3754, 20180922, 484.21, 13, 049681, xs, Red
3756, 20180922, 59.18, 2, 2038593, M, Blue
3757, 20180923, 177.42, 21, 203521, XL, Black
];

set threshold = 100;

/* Create new table called Transaction_Buckets
Compare transaction_amount field from Transaction table to threshold of 100.
Output results into a new field called Compared to Threshold
 */

Transaction_Buckets:
Load
   transaction_id,
   If(transaction_amount > $(threshold),'Greater than $(threshold)','Less than $(threshold)')
```

```
as [Compared to Threshold]
Resident Transactions;
```

Risultati

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione *if* nello script di caricamento.

transaction_id	In confronto alla soglia
3750	Inferiore a 100
3751	Superiore a 100
3752	Inferiore a 100
3753	Superiore a 100
3754	Superiore a 100
3756	Inferiore a 100
3757	Superiore a 100

Esempi - Espressioni del grafico usando if

Esempi: Espressioni del grafico

Espressione del grafico 1

Script di caricamento

Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Dopo aver caricato i dati, creare gli esempi di espressione del grafico in basso in una tabella Qlik Sense.

```
MyTable:
LOAD * inline [Date, Location, Incidents
1/3/2016, Beijing, 0
1/3/2016, Boston, 12
1/3/2016, Stockholm, 3
1/3/2016, Toronto, 0
1/4/2016, Beijing, 0
1/4/2016, Boston, 8];
```

Tabella Qlik Sense che mostra esempi della funzione *if* in un'espressione del grafico.

Date	Località	Incidents	if(Incidents>=10, 'Critical', 'Ok')	if(Incidents>=10, 'Critical', If(Incidents>=1 and Incidents<10, 'Warning', 'Ok'))
1/3/2016	Beijing	0	Ok	Ok

Date	Località	Incidents	if(Incidents>=10, 'Critical', 'Ok')	if(Incidents>=10, 'Critical', If(Incidents>=1 and Incidents<10, 'Warning', 'Ok'))
1/3/2016	Boston	12	Critical	Critical
1/3/2016	Stockholm	3	Ok	Warning
1/3/2016	Toronto	0	Ok	Ok
1/4/2016	Beijing	0	Ok	Ok
1/4/2016	Boston	8	Ok	Avviso

Espressione del grafico 2

In una nuova app, aggiungere lo script seguente in una scheda nell'editor caricamento dati, quindi caricare i dati. È quindi possibile creare la tabella con le espressioni del grafico in basso.

```
SET FirstWeekDay=0;
Load
Date(MakeDate(2022)+RecNo()-1) as Date
Autogenerate 14;
```

Tabella Qlik Sense che mostra un esempio della funzione *if* in un'espressione del grafico.

Date	WeekDay(Date)	If(WeekDay (Date)>=5,'WeekEnd','Normal Day')
1/1/2022	Sat	WeekEnd
1/2/2022	Dom	WeekEnd
1/3/2022	Lun	Giorno feriale
1/4/2022	Mar	Giorno feriale
1/5/2022	Mer	Giorno feriale
1/6/2022	Gio	Giorno feriale
1/7/2022	Fri	Giorno feriale
1/8/2022	Sat	WeekEnd
1/9/2022	Dom	WeekEnd
1/10/2022	Lun	Giorno feriale
1/11/2022	Mar	Giorno feriale
1/12/2022	Mer	Giorno feriale
1/13/2022	Gio	Giorno feriale
1/14/2022	Fri	Giorno feriale

match

La funzione **match** confronta il primo parametro con tutti i parametri seguenti e restituisce la posizione numerica delle espressioni corrispondenti. Il confronto rispetta la distinzione maiuscole/minuscole.

Sintassi:

```
match( str, expr1 [ , expr2,...exprN ])
```

*Se si desidera utilizzare il confronto senza distinzione tra lettere minuscole e maiuscole, servirsi della funzione **mixmatch**. Se si desidera utilizzare il confronto senza distinzione tra lettere minuscole e maiuscole e i caratteri speciali, servirsi della funzione **wildmatch**.*

Esempio: Script di caricamento usando match

Esempio: Script di caricamento

Script di caricamento

È possibile utilizzare match per caricare un sottogruppo dei dati. Ad esempio, è possibile restituire un valore numerico per un'espressione nella funzione. È quindi possibile limitare i dati caricati in base al valore numerico. Match restituisce 0 se non vi sono corrispondenze. Tutte le espressioni senza corrispondenze in questo esempio restituiranno quindi 0 e saranno escluse dal caricamento dei dati dall'istruzione WHERE.

Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Creare la tabella in basso in Qlik Sense per visualizzare i risultati.

```
Transactions: Load * Inline [ transaction_id, transaction_date, transaction_amount,
transaction_quantity, customer_id, size, color_code 3750, 20180830, 23.56, 2, 2038593, L, Red
3751, 20180907, 556.31, 6, 203521, m, orange 3752, 20180916, 5.75, 1, 5646471, S, blue 3753,
20180922, 125.00, 7, 3036491, l, Black 3754, 20180922, 484.21, 13, 049681, xs, Red 3756,
20180922, 59.18, 2, 2038593, M, Blue 3757, 20180923, 177.42, 21, 203521, XL, Black ];   /*
Create new table called Transaction_Buckets Create new fields called Customer, and Color code
- Blue and Black Load Transactions table. Match returns 1 for 'Blue', 2 for 'Black'.  Does not
return a value for 'blue' because match is case sensitive. Only values that returned numeric
value greater than 0  are loaded by WHERE statment into Transactions_Buckets table. */
Transaction_Buckets: Load  customer_id,  customer_id as [Customer],   color_code as [Color
Code Blue and Black] Resident Transactions Where match(color_code,'Blue','Black') > 0;
```

Risultati

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione match nello script di caricamento

Color Code Blue and Black	Customer
Nero	203521
Nero	3036491
Blu	2038593

Esempi - Espressioni del grafico usando match

Esempi: Espressioni del grafico

Espressione del grafico 1

Script di caricamento

Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Dopo aver caricato i dati, creare gli esempi di espressione del grafico in basso in una tabella Qlik Sense.

```
MyTable: Load * inline [Cities, Count Toronto, 123 Toronto, 234 Toronto, 231 Boston, 32
Boston, 23 Boston, 1341 Beijing, 234 Beijing, 45 Beijing, 235 Stockholm, 938 Stockholm, 39
Stockholm, 189 zurich, 2342 zurich, 9033 zurich, 0039];
```

La prima espressione nella tabella sottostante restituisce 0 per Stockholm perché “Stockholm” non è inclusa nell'elenco di espressioni nella funzione **match**. Restituisce 0 anche per “Zurich”, perché il confronto **match** rispetta la distinzione tra maiuscole e minuscole.

Tabella Qlik Sense che mostra esempi della funzione *match* in un'espressione del grafico

Cities	match(Cities,'Toronto','Boston','Beijing','Zurich')	match(Cities,'Toronto','Boston','Beijing','Stockholm','zurich')
Beijing	3	3
Boston	2	2
Stockholm	0	4
Toronto	1	1
zurich	0	5

Espressione del grafico 2

È possibile utilizzare match per eseguire un ordinamento personalizzato per un'espressione.

Per impostazione predefinita, le colonne vengono ordinate numericamente o alfabeticamente, a seconda dei dati.

Tabella Qlik Sense che mostra un esempio dell'impostazione di ordinamento predefinita

Cities
Beijing
Boston
Stockholm
Toronto
zurich

Per modificare l'ordine, procedere come segue:

1. Aprire la sezione **Ordinamento** del grafico nel pannello **Proprietà**.
2. Disattivare l'ordinamento automatico per la colonna su cui si desidera eseguire un ordinamento personalizzato.
3. Deselezionare **Ordina per numero** e **Ordina per lettera**.
4. Selezionare **Ordina per espressione** e quindi inserire un'espressione simile alla seguente:
   `=match( Cities, 'Toronto','Boston','Beijing','Stockholm','zurich')`
   L'ordinamento della colonna Cities verrà modificato.

Tabella Qlik Sense che mostra un esempio di modifica dell'ordinamento mediante la funzione *match*

Cities
Toronto
Boston
Beijing
Stockholm
zurich

È inoltre possibile visualizzare il valore numerico restituito.

Tabella Qlik Sense che mostra un esempio dei valori numerici restituiti dalla funzione *match*

Cities	**Cities & ' - ' & match (Cities, 'Toronto','Boston', 'Beijing','Stockholm','zurich')**
Toronto	Toronto - 1
Boston	Boston - 2
Beijing	Beijing - 3
Stockholm	Stockholm - 4
zurich	zurich - 5

mixmatch

La funzione **mixmatch** confronta il primo parametro con tutti i parametri seguenti e restituisce la posizione numerica delle espressioni corrispondenti. Il confronto non rispetta la distinzione maiuscole/minuscole.

Sintassi:

```
mixmatch( str, expr1 [ , expr2,...exprN ])
```

Se invece si desidera utilizzare il confronto con distinzione tra lettere minuscole e maiuscole, servirsi della funzione **match**. Se si desidera utilizzare il confronto senza distinzione tra lettere minuscole e maiuscole e i caratteri speciali, servirsi della funzione **wildmatch**.

Esempio - Script di caricamento usando mixmatch

Esempio: Script di caricamento

Script di caricamento

È possibile utilizzare mixmatch per caricare un sottogruppo dei dati. Ad esempio, è possibile restituire un valore numerico per un'espressione nella funzione. È quindi possibile limitare i dati caricati in base al valore numerico. Mixmatch restituisce 0 se non vi sono corrispondenze. Tutte le espressioni senza corrispondenze in questo esempio restituiranno quindi 0 e saranno escluse dal caricamento dei dati dall'istruzione WHERE.

Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Creare la tabella in basso in Qlik Sense per visualizzare i risultati.

```
Load * Inline [ transaction_id, transaction_date, transaction_amount, transaction_quantity,
customer_id, size, color_code 3750, 20180830, 23.56, 2, 2038593, L, Red 3751, 20180907,
556.31, 6, 203521, m, orange 3752, 20180916, 5.75, 1, 5646471, S, blue 3753, 20180922, 125.00,
7, 3036491, l, Black 3754, 20180922, 484.21, 13, 049681, xs, Red 3756, 20180922, 59.18, 2,
2038593, M, Blue 3757, 20180923, 177.42, 21, 203521, XL, Black ];  /* Create new table called
Transaction_Buckets Create new fields called Customer, and Color code - Black, Blue, blue Load
Transactions table. Mixmatch returns 1 for 'Black', 2 for 'Blue'.  Also returns 3 for 'blue'
because mixmatch is not case sensitive. Only values that returned numeric value greater than 0
 are loaded by WHERE statement into Transactions_Buckets table. */  Transaction_Buckets: Load
  customer_id,     customer_id as [Customer],    color_code as [Color Code - Black, Blue,
blue] Resident Transactions Where mixmatch(color_code,'Black','Blue') > 0;
```

Risultati

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione mixmatch nello script di caricamento.

Color Code Black, Blue, blue	Customer
Nero	203521

Color Code Black, Blue, blue	Customer
Nero	3036491
Blu	2038593
blue	5646471

Esempi - Espressioni del grafico usando mixmatch

Esempi: Espressioni del grafico
Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Dopo aver caricato i dati, creare gli esempi di espressione del grafico in basso in una tabella Qlik Sense.

Espressione del grafico 1

```
MyTable: Load * inline [Cities, Count Toronto, 123 Toronto, 234 Toronto, 231 Boston, 32
Boston, 23 Boston, 1341 Beijing, 234 Beijing, 45 Beijing, 235 Stockholm, 938 Stockholm, 39
Stockholm, 189 zurich, 2342 zurich, 9033 zurich, 0039];
```

La prima espressione nella tabella sottostante restituisce 0 per Stockholm perché “Stockholm” non è inclusa nell'elenco di espressioni nella funzione **mixmatch**. Restituisce 4 per “Zurich”, perché il confronto **mixmatch** rispetta la distinzione tra maiuscole e minuscole.

Tabella Qlik Sense che mostra esempi della funzione *mixmatch* in un'espressione del grafico

Cities	mixmatch(Cities,'Toronto','Boston','Beijing','Zurich')	mixmatch(Cities,'Toronto','Boston','Beijing','Stockholm','Zurich')
Beijing	3	3
Boston	2	2
Stockholm	0	4
Toronto	1	1
zurich	4	5

Espressione del grafico 2

È possibile utilizzare mixmatch per eseguire un ordinamento personalizzato per un'espressione.

Per impostazione predefinita, le colonne vengono ordinate alfabeticamente o numericamente, a seconda dei dati.

Tabella Qlik Sense che mostra un esempio dell'impostazione di ordinamento predefinita

Cities
Beijing

Cities
Boston
Stockholm
Toronto
zurich

Per modificare l'ordine, procedere come segue:

1. Aprire la sezione **Ordinamento** del grafico nel pannello **Proprietà**.
2. Disattivare l'ordinamento automatico per la colonna su cui si desidera eseguire un ordinamento personalizzato.
3. Deselezionare **Ordina per numero** e **Ordina per lettera**.
4. Selezionare **Ordina per espressione**, quindi inserire la seguente espressione:
   `=mixmatch( Cities, 'Toronto','Boston','Beijing','Stockholm','Zurich')`
   L'ordinamento della colonna Cities verrà modificato.

Tabella Qlik Sense che mostra un esempio di modifica dell'ordinamento mediante la funzione *mixmatch*.

Cities
Toronto
Boston
Beijing
Stockholm
zurich

È inoltre possibile visualizzare il valore numerico restituito.

Tabella Qlik Sense che mostra un esempio dei valori numerici restituiti dalla funzione *mixmatch*.

Cities	Cities & ' - ' & mixmatch (Cities, 'Toronto','Boston', 'Beijing','Stockholm','Zurich')
Toronto	Toronto - 1
Boston	Boston - 2
Beijing	Beijing - 3
Stockholm	Stockholm - 4
zurich	zurich - 5

pick

La funzione pick restituisce l'espressione numero *n* nell'elenco.

Sintassi:

```
pick(n, expr1[ , expr2,...exprN])
```

Argomenti:

Argomenti

Argomento	Descrizione
n	*n* è un numero intero compreso tra 1 e N.

Esempio:

Esempio

Esempio	Risultato
`pick( N, 'A','B',4, 6 )`	restituisce 'B' se N = 2 restituisce 4 se N = 3

wildmatch

La funzione **wildmatch** confronta il primo parametro con tutti i parametri seguenti e restituisce il numero dell'espressione corrispondente. Consente l'utilizzo di caratteri jolly (***** e **?**) nelle stringhe di confronto. ***** corrisponde a qualsiasi sequenza di caratteri. **?** corrisponde a qualsiasi carattere singolo. Il confronto non rispetta la distinzione maiuscole/minuscole.

Sintassi:

```
wildmatch( str, expr1 [ , expr2,...exprN ])
```

Se si desidera utilizzare il confronto senza caratteri speciali, servirsi delle funzioni **match** o **mixmatch**.

Esempio: Script di caricamento usando wildmatch

Esempio: Script di caricamento

Script di caricamento

È possibile utilizzare wildmatch per caricare un sottogruppo dei dati. Ad esempio, è possibile restituire un valore numerico per un'espressione nella funzione. È quindi possibile limitare i dati caricati in base al valore numerico. Wildmatch restituisce 0 se non vi sono corrispondenze. Tutte le espressioni senza corrispondenze in questo esempio restituiranno quindi 0 e saranno escluse dal caricamento dei dati dall'istruzione WHERE.

Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Creare la tabella in basso in Qlik Sense per visualizzare i risultati.

```
Transactions: Load * Inline [ transaction_id, transaction_date, transaction_amount,
transaction_quantity, customer_id, size, color_code 3750, 20180830, 23.56, 2, 2038593, L, Red
3751, 20180907, 556.31, 6, 203521, m, orange 3752, 20180916, 5.75, 1, 5646471, S, blue 3753,
20180922, 125.00, 7, 3036491, l, Black 3754, 20180922, 484.21, 13, 049681, xs, Red 3756,
20180922, 59.18, 2, 2038593, M, Blue 3757, 20180923, 177.42, 21, 203521, XL, Black ];   /*
Create new table called Transaction_Buckets Create new fields called Customer, and Color code
- Black, Blue, blue, red Load Transactions table. Wildmatch returns 1 for 'Black', 'Blue', and
```

```
'blue', and 2 for 'Red'.  Only values that returned numeric value greater than 0  are loaded
by WHERE statement into Transactions_Buckets table. */   Transaction_Buckets: Load
customer_id,    customer_id as [Customer],    color_code as [Color Code Black, Blue, blue,
Red] Resident Transactions Where wildmatch(color_code,'Bl*','R??') > 0;
```

Risultati

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione *wildmatch* nello script di caricamento

Color Code Black, Blue, blue, Red	Customer
Nero	203521
Nero	3036491
Blu	2038593
blue	5646471
Rosso	049681
Rosso	2038593

Esempi: Espressioni del grafico usando wildmatch

Esempio: Espressione del grafico

Espressione del grafico 1

Creare una nuova scheda nell'editor caricamento dati, quindi caricare i seguenti dati come caricamento inline. Dopo aver caricato i dati, creare gli esempi di espressione del grafico in basso in una tabella Qlik Sense.

```
MyTable: Load * inline [Cities, Count Toronto, 123 Toronto, 234 Toronto, 231 Boston, 32
Boston, 23 Boston, 1341 Beijing, 234 Beijing, 45 Beijing, 235 Stockholm, 938 Stockholm, 39
Stockholm, 189 zurich, 2342 zurich, 9033 zurich, 0039];
```

La prima espressione nella tabella sottostante restituisce 0 per Stockholm perché “Stockholm” non è inclusa nell'elenco di espressioni nella funzione **wildmatch**. Restituisce 0 anche per “Boston” perché ? corrisponde a un singolo carattere.

Tabella Qlik Sense che mostra esempi della funzione *wildmatch* in un'espressione del grafico

Cities	wildmatch(Cities,'Tor*','?ton','Beijing','*urich')	wildmatch(Cities,'Tor*','???ton','Beijing','Stockholm','*urich')
Beijing	3	3
Boston	0	2
Stockholm	0	4

Cities	wildmatch(Cities,'Tor*','?ton','Beijing','*urich')	wildmatch(Cities,'Tor*','???ton','Beijing','Stockholm','*urich')
Toronto	1	1
zurich	4	5

Espressione del grafico 2

È possibile utilizzare wildmatch per eseguire un ordinamento personalizzato per un'espressione.

Per impostazione predefinita, le colonne vengono ordinate numericamente o alfabeticamente, a seconda dei dati.

Tabella Qlik Sense che mostra un esempio dell'impostazione di ordinamento predefinita

Cities
Beijing
Boston
Stockholm
Toronto
zurich

Per modificare l'ordine, procedere come segue:

1. Aprire la sezione **Ordinamento** del grafico nel pannello **Proprietà**.
2. Disattivare l'ordinamento automatico per la colonna su cui si desidera eseguire un ordinamento personalizzato.
3. Deselezionare **Ordina per numero** e **Ordina per lettera**.
4. Selezionare **Ordina per espressione** e quindi inserire un'espressione simile alla seguente:
   `=wildmatch( Cities, 'Tor*','???ton','Beijing','Stockholm','*urich')`
   L'ordinamento della colonna Cities verrà modificato.

Tabella Qlik Sense che mostra un esempio di modifica dell'ordinamento mediante la funzione *wildmatch*.

Cities
Toronto
Boston
Beijing
Stockholm
zurich

È inoltre possibile visualizzare il valore numerico restituito.

Tabella Qlik Sense che mostra un esempio dei valori numerici restituiti dalla funzione *wildmatch*

Cities	Cities & ' - ' & wildmatch (Cities, 'Tor*','???ton','Beijing','Stockholm','*urich')
Toronto	Toronto - 1
Boston	Boston - 2
Beijing	Beijing - 3
Stockholm	Stockholm - 4
zurich	zurich - 5

5.6 Funzioni di conteggio

In questa sezione vengono descritte le funzioni correlate ai contatori di record durante la valutazione dell'istruzione **LOAD** nello script di caricamento dei dati. L'unica funzione che è possibile utilizzare nelle espressioni grafiche è **RowNo()**.

Ad alcune funzioni di conteggio non sono associati parametri, ma sono comunque richieste le parentesi finali.

Prospetto delle funzioni di conteggio

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

autonumber

Questa funzione dello script restituisce un valore intero univoco per ciascun valore calcolato distinto di *expression* rilevato durante l'esecuzione dello script. Questa funzione può essere utilizzata per creare una rappresentazione compatta di memoria che rappresenta una chiave complessa.

```
autonumber (expression[ , AutoID])
```

autonumberhash128

Questa funzione di script calcola un hash a 128 bit dei valori di espressione di input combinati e restituisce un valore intero univoco per ciascun valore hash distinto rilevato durante l'esecuzione dello script. Questa funzione può essere, ad esempio, utilizzata per creare una rappresentazione compatta di memoria che rappresenta una chiave complessa.

```
autonumberhash128 (expression {, expression})
```

autonumberhash256

Questa funzione di script calcola un hash a 256 bit dei valori di espressione di input combinati e restituisce un valore intero univoco per ciascun valore hash distinto rilevato durante l'esecuzione dello script. Questa funzione può essere utilizzata per creare una rappresentazione compatta di memoria che rappresenta una chiave complessa.

```
autonumberhash256 (expression {, expression})
```

IterNo

Questa funzione dello script restituisce un numero intero che indica il numero di volte in cui verrà valutato un singolo record in un'istruzione **LOAD** con una clausola **while**. La prima ripetizione ha valore 1. La funzione **IterNo** è significativa solo se utilizzata in combinazione con una clausola **while**.

```
IterNo ( )
```

RecNo

Questa funzione di script restituisce un valore intero relativo al numero della riga della tabella corrente attualmente in corso di lettura. Il primo record è il numero 1.

```
RecNo ( )
```

RowNo - script function

Questa funzione dello script restituisce un numero intero relativo alla posizione della riga attuale nella tabella interna risultante in Qlik Sense. La prima riga è il numero 1.

```
RowNo ( )
```

RowNo - chart function

RowNo() restituisce il numero della riga attuale nel segmento di colonna attuale in una tabella. Per i grafici bitmap, **RowNo()** restituisce il numero della riga attuale nell'equivalente di tabella lineare del grafico.

```
RowNo - funzione per grafici([TOTAL])
```

autonumber

Questa funzione dello script restituisce un valore intero univoco per ciascun valore calcolato distinto di *expression* rilevato durante l'esecuzione dello script. Questa funzione può essere utilizzata per creare una rappresentazione compatta di memoria che rappresenta una chiave complessa.

*È possibile connettere solo chiavi **autonumber** che sono state generate nello stesso caricamento di dati, poiché il numero intero viene generato in base all'ordine di lettura della tabella. Se si necessita di utilizzare chiavi permanenti tra i caricamenti di dati indipendentemente dall'ordinamento dei dati sorgente è necessario utilizzare le funzioni **hash128**, **hash160** o **hash256**.*

Sintassi:

```
autonumber(expression[ , AutoID])
```

Argomenti:

Argomento	Descrizione
AutoID	Per creare più istanze di conteggio qualora si utilizzi la funzione **autonumber** su chiavi differenti all'interno dello script, è possibile utilizzare un parametro opzionale *AutoID* per il nome di ciascun contatore.

Esempio: Creazione di una chiave composita

Nell'esempio viene creata una chiave composita tramite la funzione **autonumber** per conservare la memoria. L'esempio è breve a scopo dimostrativo, ma risulterebbe significativo con una tabella contenente un elevato numero di righe.

Dati di esempio

Region	Year	Month	Sales
North	2014	May	245
North	2014	May	347
North	2014	June	127
South	2014	June	645
South	2013	May	367
South	2013	May	221

I dati sorgente vengono caricati tramite dati inline. Quindi viene aggiunto un'istruzione preceding load che crea una chiave composita dai campi Region, Year e Month.

```
RegionSales:
LOAD *,
AutoNumber(Region&Year&Month) as RYMkey;

LOAD * INLINE
[ Region, Year, Month, Sales
North, 2014,   May,    245
North, 2014,   May,    347
North, 2014,   June,   127
South, 2014,   June,   645
South, 2013,   May, 367
South, 2013,   May,    221
];
```

La tabella risultante avrà l'aspetto seguente:

Tabella dei risultati

Region	Year	Month	Sales	RYMkey
North	2014	May	245	1
North	2014	May	347	1
North	2014	June	127	2
South	2014	June	645	3
South	2013	May	367	4
South	2013	May	221	4

In questo esempio è possibile fare riferimento a RYMkey, per l'esempio 1, anziché alla stringa 'North2014May' se si desidera effettuare il collegamento a un'altra tabella.

Ora viene caricata una tabella sorgente dei costi in modo simile. I campi Region, Year e Month vengono esclusi nell'istruzione preceding load per evitare di creare una chiave sintetica. È stata già creata una chiave composita con la funzione **autonumber** tramite il collegamento delle tabelle.

```
RegionCosts:
LOAD Costs,
AutoNumber(Region&Year&Month) as RYMkey;

LOAD * INLINE
[ Region, Year, Month, Costs
South, 2013,   May,    167
North, 2014,   May,    56
North, 2014,   June,   199
South, 2014,   June,   64
South, 2013,   May, 172
South, 2013,   May,    126
];
```

Ora è possibile aggiungere una visualizzazione tabella a un foglio e aggiungere i campi Region, Year e Month così come le misure Sum per le vendite e i costi. La tabella avrà il seguente aspetto:

Tabella dei risultati

Region	Year	Month	Sum([Sales])	Sum([Costs])
Totals	-	-	1952	784
North	2014	June	127	199
North	2014	May	592	56
South	2014	June	645	64
South	2013	May	588	465

autonumberhash128

Questa funzione di script calcola un hash a 128 bit dei valori di espressione di input combinati e restituisce un valore intero univoco per ciascun valore hash distinto rilevato durante l'esecuzione dello script. Questa funzione può essere, ad esempio, utilizzata per creare una rappresentazione compatta di memoria che rappresenta una chiave complessa.

*È possibile connettere solo chiavi **autonumberhash128** che sono state generate nello stesso caricamento di dati, poiché il numero intero viene generato in base all'ordine di lettura della tabella. Se si necessita di utilizzare chiavi permanenti tra i caricamenti di dati indipendentemente dall'ordinamento dei dati sorgente è necessario utilizzare le funzioni **hash128**, **hash160** o **hash256**.*

Sintassi:

```
autonumberhash128(expression {, expression})
```

Esempio: Creazione di una chiave composita

Nell'esempio viene creata una chiave composita tramite la funzione **autonumberhash128** per conservare la memoria. L'esempio è breve a scopo dimostrativo, ma risulterebbe significativo con una tabella contenente un elevato numero di righe.

Dati di esempio

Region	Year	Month	Sales
North	2014	May	245
North	2014	May	347
North	2014	June	127
South	2014	June	645
South	2013	May	367
South	2013	May	221

I dati sorgente vengono caricati tramite dati inline. Quindi viene aggiunto un'istruzione preceding load che crea una chiave composita dai campi Region, Year e Month.

```
RegionSales:
LOAD *,
AutoNumberHash128(Region, Year, Month) as RYMkey;

LOAD * INLINE
[ Region, Year, Month, Sales
North, 2014,   May,    245
North, 2014,   May,    347
North, 2014,   June,   127
South, 2014,   June,   645
South, 2013,   May, 367
South, 2013,   May,    221
];
```

La tabella risultante avrà l'aspetto seguente:

Tabella dei risultati

Region	Year	Month	Sales	RYMkey
North	2014	May	245	1
North	2014	May	347	1
North	2014	June	127	2

Region	Year	Month	Sales	RYMkey
South	2014	June	645	3
South	2013	May	367	4
South	2013	May	221	4

In questo esempio è possibile fare riferimento a RYMkey, per l'esempio 1, anziché alla stringa 'North2014May' se si desidera effettuare il collegamento a un'altra tabella.

Ora viene caricata una tabella sorgente dei costi in modo simile. I campi Region, Year e Month vengono esclusi nell'istruzione preceding load per evitare di creare una chiave sintetica. È stata già creata una chiave composita con la funzione **autonumberhash128** tramite il collegamento delle tabelle.

```
RegionCosts:
LOAD Costs,
AutoNumberHash128(Region, Year, Month) as RYMkey;

LOAD * INLINE
[ Region, Year, Month, Costs
South, 2013,   May,    167
North, 2014,   May,    56
North, 2014,   June,   199
South, 2014,   June,   64
South, 2013,   May, 172
South, 2013,   May,    126
];
```

Ora è possibile aggiungere una visualizzazione tabella a un foglio e aggiungere i campi Region, Year e Month così come le misure Sum per le vendite e i costi. La tabella avrà il seguente aspetto:

Tabella dei risultati

Region	Year	Month	Sum([Sales])	Sum([Costs])
Totals	-	-	1952	784
North	2014	June	127	199
North	2014	May	592	56
South	2014	June	645	64
South	2013	May	588	465

autonumberhash256

Questa funzione di script calcola un hash a 256 bit dei valori di espressione di input combinati e restituisce un valore intero univoco per ciascun valore hash distinto rilevato durante l'esecuzione dello script. Questa funzione può essere utilizzata per creare una rappresentazione compatta di memoria che rappresenta una chiave complessa.

È possibile connettere solo chiavi ***autonumberhash256*** *che sono state generate nello stesso caricamento di dati, poiché il numero intero viene generato in base all'ordine di lettura della tabella. Se si necessita di utilizzare chiavi permanenti tra i caricamenti di dati indipendentemente dall'ordinamento dei dati sorgente è necessario utilizzare le funzioni* ***hash128****,* ***hash160*** *o* ***hash256****.*

Sintassi:

```
autonumberhash256(expression {, expression})
```

Esempio: Creazione di una chiave composita

Nell'esempio viene creata una chiave composita tramite la funzione **autonumberhash256** per conservare la memoria. L'esempio è breve a scopo dimostrativo, ma risulterebbe significativo con una tabella contenente un elevato numero di righe.

Tabella di esempio

Region	Year	Month	Sales
North	2014	May	245
North	2014	May	347
North	2014	June	127
South	2014	June	645
South	2013	May	367
South	2013	May	221

I dati sorgente vengono caricati tramite dati inline. Quindi viene aggiunto un'istruzione preceding load che crea una chiave composita dai campi Region, Year e Month.

```
RegionSales:
LOAD *,
AutoNumberHash256(Region, Year, Month) as RYMkey;

LOAD * INLINE
[ Region, Year, Month, Sales
North, 2014,   May,    245
North, 2014,   May,    347
North, 2014,   June,   127
South, 2014,   June,   645
South, 2013,   May, 367
South, 2013,   May,    221
];
```

La tabella risultante avrà l'aspetto seguente:

Tabella dei risultati

Region	Year	Month	Sales	RYMkey
North	2014	May	245	1
North	2014	May	347	1
North	2014	June	127	2
South	2014	June	645	3
South	2013	May	367	4
South	2013	May	221	4

In questo esempio è possibile fare riferimento a RYMkey, per l'esempio 1, anziché alla stringa 'North2014May' se si desidera effettuare il collegamento a un'altra tabella.

Ora viene caricata una tabella sorgente dei costi in modo simile. I campi Region, Year e Month vengono esclusi nell'istruzione preceding load per evitare di creare una chiave sintetica. È stata già creata una chiave composita con la funzione **autonumberhash256** tramite il collegamento delle tabelle.

```
RegionCosts:
LOAD Costs,
AutoNumberHash256(Region, Year, Month) as RYMkey;

LOAD * INLINE
[ Region, Year, Month, Costs
South, 2013,   May,    167
North, 2014,   May,    56
North, 2014,   June,   199
South, 2014,   June,   64
South, 2013,   May, 172
South, 2013,   May,    126
];
```

Ora è possibile aggiungere una visualizzazione tabella a un foglio e aggiungere i campi Region, Year e Month così come le misure Sum per le vendite e i costi. La tabella avrà il seguente aspetto:

Tabella dei risultati

Region	Year	Month	Sum([Sales])	Sum([Costs])
Totals	-	-	1952	784
North	2014	June	127	199
North	2014	May	592	56
South	2014	June	645	64
South	2013	May	588	465

IterNo

Questa funzione dello script restituisce un numero intero che indica il numero di volte in cui verrà valutato un singolo record in un'istruzione **LOAD** con una clausola **while**. La prima ripetizione ha valore 1. La funzione **IterNo** è significativa solo se utilizzata in combinazione con una clausola **while**.

Sintassi:

```
IterNo( )
```

Esempi e risultati:

Esempio:

```
LOAD
   IterNo() as Day,
   Date( StartDate + IterNo() - 1 ) as Date
   While StartDate + IterNo() - 1 <= EndDate;

LOAD * INLINE
[StartDate, EndDate
2014-01-22, 2014-01-26
];
```

L'istruzione **LOAD** genererà un record per la data compresa nella scala definita da **StartDate** e **EndDate**.

In questo esempio la tabella risultante sarà simile alla seguente:

Tabella dei risultati

Day	Date
1	2014-01-22
2	2014-01-23
3	2014-01-24
4	2014-01-25
5	2014-01-26

RecNo

Questa funzione di script restituisce un valore intero relativo al numero della riga della tabella corrente attualmente in corso di lettura. Il primo record è il numero 1.

Sintassi:

```
RecNo( )
```

Diversamente da **RowNo( )**, che conta le righe nella tabella di Qlik Sense risultante, la funzione **RecNo( )** conta i record nella tabella di dati non elaborati e viene reimpostata quando una tabella di dati non elaborati risulta concatenata con un'altra.

Esempio: Script di caricamento dei dati

Caricamento di tabelle di dati non elaborati:

```
Tab1:
LOAD * INLINE
[A, B
1, aa
2,cc
3,ee];

Tab2:
LOAD * INLINE
[C, D
5, xx
4,yy
6,zz];
```

Caricamento dei numeri di record e riga per le righe selezionate:

```
QTab:
LOAD *,
RecNo( ),
RowNo( )
resident Tab1 where A<>2;

LOAD
C as A,
D as B,
RecNo( ),
RowNo( )
resident Tab2 where A<>5;

//We don't need the source tables anymore, so we drop them
Drop tables Tab1, Tab2;
```

Tabella interna di Qlik Sense risultante:

Tabella dei risultati

A	B	RecNo( )	RowNo( )
1	aa	1	1
3	ee	3	2
4	yy	2	3
6	zz	3	4

RowNo

Questa funzione dello script restituisce un numero intero relativo alla posizione della riga attuale nella tabella interna risultante in Qlik Sense. La prima riga è il numero 1.

Sintassi:

```
RowNo( [TOTAL])
```

Diversamente da **RecNo( )**, che conta i record nella tabella di dati non elaborati, la funzione **RowNo( )** non conta i record che sono esclusi da clausole **where** e non viene reimpostata quando una tabella di dati non elaborati risulta concatenata con un'altra.

*Se si utilizza un'istruzione preceding load, ossia più istruzioni **LOAD** in pila, lette dalla stessa tabella, è possibile utilizzare solo **RowNo( )** nell'istruzione **LOAD** del livello superiore. Se si utilizza **RowNo( )** nelle istruzioni **LOAD** successive, viene restituito 0.*

Esempio: Script di caricamento dei dati

Caricamento di tabelle di dati non elaborati:

```
Tab1:
LOAD * INLINE
[A, B
1, aa
2,cc
3,ee];

Tab2:
LOAD * INLINE
[C, D
5, xx
4,yy
6,zz];
```

Caricamento dei numeri di record e riga per le righe selezionate:

```
QTab:
LOAD *,
RecNo( ),
RowNo( )
resident Tab1 where A<>2;

LOAD
C as A,
D as B,
RecNo( ),
RowNo( )
resident Tab2 where A<>5;

//We don't need the source tables anymore, so we drop them
Drop tables Tab1, Tab2;
```

Tabella interna di Qlik Sense risultante:

Tabella dei risultati

A	B	RecNo( )	RowNo( )
1	aa	1	1
3	ee	3	2
4	yy	2	3
6	zz	3	4

RowNo - funzione per grafici

RowNo() restituisce il numero della riga attuale nel segmento di colonna attuale in una tabella. Per i grafici bitmap, **RowNo()** restituisce il numero della riga attuale nell'equivalente di tabella lineare del grafico.

Se la tabella o l'equivalente di tabella include più dimensioni verticali, il segmento colonna attuale includerà solo righe contenenti gli stessi valori della riga attuale in tutte le colonne di dimensione, eccetto la colonna che mostra l'ultima dimensione nell'ordinamento tra campi.

Segmenti delle colonne

L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Sintassi:

```
RowNo([TOTAL])
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomento	Descrizione
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Esempio: Espressione del grafico usando RowNo

Esempio - Espressione del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare gli esempi di espressione del grafico in basso.

```
Temp:
LOAD * inline [
Customer|Product|OrderNumber|UnitSales|UnitPrice
Astrida|AA|1|4|16
Astrida|AA|7|10|15
Astrida|BB|4|9|9
Betacab|CC|6|5|10
Betacab|AA|5|2|20
Betacab|BB|1|25| 25
Canutility|AA|3|8|15
Canutility|CC|5|4|19
Divadip|CC|2|4|16
Divadip|DD|3|1|25
] (delimiter is '|');
```

Espressione del grafico

Crea una visualizzazione tabella in un foglio Qlik Sense con **Customer** e **UnitSales** come dimensioni. Aggiungere rispettivamente `RowNo( )` e `RowNo(TOTAL)` come misure etichettate **Riga nel Segmento** e **Row Number**. Aggiungere la seguente espressione alla tabella come misura:

```
If( RowNo( )=1, 0, UnitSales / Above( UnitSales ))
```

Risultato

Customer	UnitSales	Row in Segment	Row Number	If(RowNo( )=1, 0, UnitSales / Above(UnitSales))
Astrida	4	1	1	0
Astrida	9	2	2	2.25
Astrida	10	3	3	1.1111111111111
Betacab	2	1	4	0
Betacab	5	2	5	2.5
Betacab	25	3	6	5
Canutility	4	1	7	0
Canutility	8	2	8	2

Customer	UnitSales	Row in Segment	Row Number	If(RowNo( )=1, 0, UnitSales / Above(UnitSales))
Divadip	1	1	9	0
Divadip	4	2	10	4

Spiegazione

La colonna **Row in Segment** restituisce i risultati 1,2,3 per il segmento di colonna contenente i valori di UnitSales per il cliente Astrida. La numerazione delle righe riparte da 1 per il segmento di colonna successivo, vale a dire Betacab.

La colonna **Row Number** ignora le dimensioni a causa dell'argomento `TOTAL` per `RowNo()` e conteggia le righe nella tabella.

Questa espressione restituisce 0 per la prima riga di ogni segmento di colonna, quindi la colonna mostrerà:

0, 2.25, 1.1111111, 0, 2.5, 5, 0, 2, 0 e 4.

Vedere anche:

p *Above - funzione per grafici (page 1267)*

5.7 Funzioni data e ora

Le funzioni data e ora di Qlik Sense consentono di trasformare e convertire i valori di data e ora. Tutte le funzioni possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche.

Le funzioni sono basate su un numero seriale di data e ora che equivale al numero di giorni trascorsi dal 30 dicembre 1899. Il valore di numero intero rappresenta il giorno e il valore frazionale rappresenta l'ora del giorno.

Qlik Sense utilizza il valore numerico del parametro, pertanto un numero è valido come parametro anche quando non è formattato come data oppure come ora. Se il parametro non corrisponde a un valore numerico, ad esempio una stringa, Qlik Sense tenta di interpretare la stringa in base alle variabili di ambiente della data e dell'ora.

Se il formato dell'ora utilizzato nel parametro non corrisponde a quello impostato nelle variabili di ambiente, Qlik Sense non sarà in grado di eseguire un'interpretazione corretta. Per risolvere questo problema, modificare le impostazioni o utilizzare una funzione di interpretazione.

Negli esempi per ciascuna funzione vengono utilizzati i formati predefiniti di ora e data hh:mm:ss e YYYY-MM-DD (ISO 8601).

Quando elabora un valore di data e ora con una funzione data o ora, Qlik Sense ignora eventuali parametri relativi all'ora legale, a meno che la funzione data o ora non comprenda una posizione geografica.

Ad esempio, la funzione `ConvertToLocalTime( filetime('Time.qvd'), 'Paris')` *utilizzerà i parametri relativi all'ora legale, mentre la funzione* `ConvertToLocalTime(filetime ('Time.qvd'), 'GMT-01:00')` *non li utilizzerà.*

Prospetto delle funzioni data e ora

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Espressioni di numero intero dell'ora

second

Questa funzione restituisce un numero intero che rappresenta il secondo in cui la frazione di **expression** viene interpretata come ora in base all'interpretazione numerica standard.

```
second (expression)
```

minute

Questa funzione restituisce un numero intero che rappresenta il minuto in cui la frazione di **expression** viene interpretata come ora in base all'interpretazione numerica standard.

```
minute (expression)
```

hour

Questa funzione restituisce un numero intero che rappresenta l'ora in cui la frazione di **expression** viene interpretata come ora in base all'interpretazione numerica standard.

```
hour (expression)
```

day

Questa funzione restituisce un numero intero che rappresenta il giorno in cui la frazione di **expression** viene interpretata come data in base all'interpretazione numerica standard.

```
day (expression)
```

week

Questa funzione restituisce un numero intero che rappresenta il numero della settimana in base allo standard ISO 8601. Il numero della settimana viene calcolato a partire dall'interpretazione della data dell'espressione in base all'interpretazione numerica standard.

```
week (expression)
```

month

Questa funzione restituisce un valore duale: il nome del mese come definito nella variabile di ambiente **MonthNames** e un numero intero compreso tra 1 e 12. Il numero del mese viene calcolato a partire dall'interpretazione della data dell'espressione in base all'interpretazione numerica standard.

```
month (expression)
```

year

Questa funzione restituisce un numero intero che rappresenta l'anno in cui **expression** viene interpretato come data in base all'interpretazione numerica standard.

```
year (expression)
```

weekyear

Questa funzione restituisce l'anno a cui appartiene il numero della settimana in base alle variabili di ambiente. I numeri della settimana rientrano in un intervallo approssimativo compreso tra 1 e 52.

```
weekyear (expression)
```

weekday

Questa funzione restituisce un valore duale con:

- Il nome di un giorno come definito nella variabile di ambiente **DayNames**.
- Un numero intero compreso tra 0 e 6 che corrisponde al giorno nominale della settimana (0-6).

```
weekday (date)
```

Funzioni timestamp

now

Questa funzione restituisce un indicatore temporale dell'ora attuale. La funzione restituisce i valori nel formato della variabile di sistema **TimeStamp**. Il valore predefinito **timer_mode** è 1.

```
now ([ timer_mode])
```

today

Questa funzione restituisce la data attuale. La funzione restituisce i valori nel formato della variabile di sistema `DateFormat`.

```
today ([timer_mode])
```

LocalTime

Questa funzione restituisce un indicatore temporale dell'ora attuale fornita per il fuso orario specificato.

```
localtime ([timezone [, ignoreDST ]])
```

Funzioni make

makedate

Questa funzione restituisce una data calcolata dall'anno **YYYY**, dal mese **MM** e dal giorno **DD**.

```
makedate (YYYY [ , MM [ , DD ] ])
```

makeweekdate

Questa funzione restituisce una data calcolata dall'anno, dal numero di settimana e dal giorno della settimana.

```
makeweekdate (YYYY [ , WW [ , D ] ])
```

maketime

Questa funzione restituisce una data calcolata dall'ora **hh**, dal minuto **mm** e dal secondo **ss**.

```
maketime (hh [ , mm [ , ss [ .fff ] ] ])
```

Altre funzioni date

AddMonths

Questa funzione restituisce la data che ricorre **n** mesi dopo **startdate** oppure, se **n** è negativo, la data che ricorre **n** mesi prima di **startdate**.

```
addmonths (startdate, n , [ , mode])
```

AddYears

Questa funzione restituisce la data che ricorre **n** anni dopo **startdate** oppure, se **n** è negativo, la data che ricorre **n** anni prima di **startdate**.

```
addyears (startdate, n)
```

yeartodate

Questa funzione stabilisce se l'indicatore temporale di input ricade all'interno dell'anno in cui lo script è stato caricato per l'ultima volta e restituisce True in caso affermativo e False in caso negativo.

```
yeartodate (date [ , yearoffset [ , firstmonth [ , todaydate] ] ])
```

Funzioni timezone

timezone

Questa funzione restituisce il fuso orario impostato sul computer dove è in esecuzione il motore Qlik.

```
timezone ( )
```

GMT

Questa funzione restituisce il valoreGreenwich Mean Timeattuale ricavato dalle impostazioni locali.

```
GMT ( )
```

UTC

Restituisce il Coordinated Universal Time attuale.

```
UTC ( )
```

daylightsaving

Restituisce il valore di regolazione attuale per l'ora legale, come definito in Windows.

```
daylightsaving ( )
```

converttolocaltime

Converte un indicatore temporale UTC o GMT in ora locale come valore duale. Il luogo può essere qualsiasi città, località o fuso orario nel mondo.

```
converttolocaltime (timestamp [, place [, ignore_dst=false]])
```

Funzioni set time

setdateyear

Questa funzione utilizza come input un **timestamp** e un **year** e aggiorna il **timestamp** con l'**year** specificato nell'input.

```
setdateyear  (timestamp, year)
```

setdateyearmonth

Questa funzione utilizza come input un **timestamp**, un **month** e un **year** e aggiorna il **timestamp** con l'**year** e il **month** specificati nell'input.

```
setdateyearmonth  (timestamp, year, month)
```

Funzioni in...

inyear

Questa funzione restituisce True se **timestamp** ricade all'interno dell'anno contenente **base_date**.

```
inyear (date, basedate , shift [, first_month_of_year = 1])
```

inyeartodate

Questa funzione restituisce True se **timestamp** ricade all'interno della parte dell'anno contenente **base_date** fino a includere l'ultimo millisecondo di **base_date**.

```
inyeartodate  (date, basedate , shift [, first_month_of_year = 1])
```

inquarter

Questa funzione restituisce True se **timestamp** ricade all'interno del trimestre contenente **base_date**.

```
inquarter  (date, basedate , shift [, first_month_of_year = 1])
```

inquartertodate

Questa funzione restituisce True se **timestamp** ricade all'interno della parte del trimestre contenente **base_date** fino a includere l'ultimo millisecondo di **base_date**.

```
inquartertodate  (date, basedate , shift [, first_month_of_year = 1])
```

inmonth

Questa funzione restituisce True se **timestamp** ricade all'interno del mese contenente **base_date**.

```
inmonth (date, basedate , shift)
```

inmonthtodate

Restituisce True se **date** ricade nella parte di mese contenente **basedate** fino a includere l'ultimo millisecondo di **basedate**.

```
inmonthtodate  (date, basedate , shift)
```

inmonths

Questa funzione consente di verificare se un indicatore temporale cade nello stesso mese, bimestre, trimestre, quadrimestre o semestre di una data base. È inoltre possibile stabilire se l'indicatore temporale ricade all'interno di un periodo di tempo precedente o successivo.

```
inmonths (n, date, basedate , shift [, first_month_of_year = 1])
```

inmonthstodate

Questa funzione stabilisce se un indicatore temporale ricade all'interno della parte di un periodo di un mese, bimestre, trimestre, quadrimestre o semestre fino a includere l'ultimo millisecondo di `base_date`. È inoltre possibile stabilire se l'indicatore temporale ricade all'interno di un periodo di tempo precedente o successivo.

```
inmonthstodate  (n, date, basedate , shift [, first_month_of_year = 1])
```

inweek

Questa funzione restituisce True se **timestamp** ricade all'interno della settimana contenente **base_date**.

```
inweek (date, basedate , shift [, weekstart])
```

inweektodate

Questa funzione restituisce True se **timestamp** ricade all'interno della parte della settimana contenente **base_date** fino a includere l'ultimo millisecondo di **base_date**.

```
inweektodate  (date, basedate , shift [, weekstart])
```

inlunarweek

Questa funzione determina se **timestamp** ricade all'interno della settimana lunare contenente **base_date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana, a parte l'ultima settimana dell'anno, ogni settimana conterrà esattamente sette giorni.

```
inlunarweek  (date, basedate , shift [, weekstart])
```

inlunarweektodate

Questa funzione stabilisce se **timestamp** ricade all'interno della parte della settimana lunare fino a includere l'ultimo millisecondo di **base_date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana e, a parte l'ultima settimana dell'anno, conterranno esattamente sette giorni.

```
inlunarweektodate  (date, basedate , shift [, weekstart])
```

inday

Questa funzione restituisce True se **timestamp** ricade all'interno del giorno contenente **base_timestamp**.

```
inday (timestamp, basetimestamp , shift [, daystart])
```

indaytotime

Questa funzione restituisce True se **timestamp** ricade nella parte del giorno contenente **base_timestamp** fino a includere il millisecondo esatto di **base_timestamp**.

```
indaytotime (timestamp, basetimestamp , shift [, daystart])
```

Funzioni start ... end

yearstart

Questa funzione restituisce un indicatore temporale corrispondente all'inizio del primo giorno dell'anno contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

```
yearstart ( date [, shift = 0 [, first_month_of_year = 1]])
```

yearend

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo dell'ultimo giorno dell'anno contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

```
yearend ( date [, shift = 0 [, first_month_of_year = 1]])
```

yearname

Questa funzione restituisce un anno di quattro cifre come valore di visualizzazione con un valore numerico sottostante corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno dell'anno contenente **date**.

```
yearname (date [, shift = 0 [, first_month_of_year = 1]] )
```

quarterstart

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante il primo millisecondo del trimestre contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

```
quarterstart (date [, shift = 0 [, first_month_of_year = 1]])
```

quarterend

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo del trimestre contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

```
quarterend (date [, shift = 0 [, first_month_of_year = 1]])
```

quartername

Questa funzione restituisce un valore di visualizzazione che mostra i mesi del trimestre (formattati in base alla variabile di script **MonthNames**) e l'anno con valore numerico sottostante corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno del trimestre.

```
quartername (date [, shift = 0 [, first_month_of_year = 1]])
```

monthstart

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno del mese contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

```
monthstart (date [, shift = 0])
```

monthend

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo dell'ultimo giorno del mese contenente `date`. Il formato di output predefinito sarà il formato `DateFormat` impostato nello script.

```
monthend (date [, shift = 0])
```

monthname

Questa funzione restituisce un valore di visualizzazione che mostra il mese (formattato in base alla variabile di script **MonthNames**) e l'anno il cui valore numerico sottostante corrisponde a un indicatore temporale recante il primo millisecondo del primo giorno del mese.

```
monthname (date [, shift = 0])
```

monthsstart

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante il primo millisecondo del mese, del bimestre, del trimestre, del quadrimestre o del semestre contenente una data di base. È inoltre possibile individuare l'indicatore temporale per un periodo di tempo precedente o successivo.Il formato di output predefinito è il formato **DateFormat** impostato nello script.

```
monthsstart (n, date [, shift = 0 [, first_month_of_year = 1]])
```

monthsend

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo del mese, del bimestre, del trimestre, del quadrimestre o del semestre contenente una data di base. È inoltre possibile individuare l'indicatore temporale per un periodo di tempo precedente o successivo.

```
monthsend (n, date [, shift = 0 [, first_month_of_year = 1]])
```

monthsname

Questa funzione restituisce un valore di visualizzazione che rappresenta l'intervallo dei mesi del periodo (formattati in base alla variabile di script **MonthNames**) e l'anno. Il valore numerico sottostante corrisponde a un indicatore temporale recante il primo millisecondo del mese, del bimestre, del trimestre, del quadrimestre o del semestre contenente una data di base.

```
monthsname (n, date [, shift = 0 [, first_month_of_year = 1]])
```

weekstart

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno della settimana di calendario contenente **date**. Il formato di output predefinito è il formato **DateFormat** impostato nello script.

```
weekstart (date [, shift = 0 [,weekoffset = 0]])
```

weekend

Questa funzione restituisce un valore corrispondente a un timestamp recante l'ultimo millisecondo dell'ultimo giorno della settimana di calendario contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

```
weekend (date [, shift = 0 [,weekoffset = 0]])
```

weekname

Questa funzione restituisce un valore che mostra l'anno e il numero della settimana con un valore numerico sottostante corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno della settimana contenente **date**.

```
weekname (date [, shift = 0 [,weekoffset = 0]])
```

lunarweekstart

Questa funzione restituisce un valore corrispondente a un indicatore temporale del primo millisecondo del primo giorno della settimana lunare contenente **date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana e, a parte l'ultima settimana dell'anno, conterranno esattamente sette giorni.

```
lunarweekstart (date [, shift = 0 [,weekoffset = 0]])
```

lunarweekend

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo dell'ultimo giorno della settimana lunare contenente **date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana e, a parte l'ultima settimana dell'anno, conterranno esattamente sette giorni.

```
lunarweekend (date [, shift = 0 [,weekoffset = 0]])
```

lunarweekname

Questa funzione restituisce un valore di visualizzazione che mostra l'anno e il numero della settimana lunare corrispondente a un indicatore temporale del primo millisecondo del primo giorno della settimana lunare contenente **date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana e, a parte l'ultima settimana dell'anno, conterranno esattamente sette giorni.

```
lunarweekname (date [, shift = 0 [,weekoffset = 0]])
```

daystart

Questa funzione restituisce un valore corrispondente a un indicatore temporale con il primo millisecondo del giorno contenuto nell'argomento **time**. Il formato di output predefinito sarà il formato **TimestampFormat** impostato nello script.

```
daystart (timestamp [, shift = 0 [, dayoffset = 0]])
```

dayend

Questa funzione restituisce un valore corrispondente a un indicatore temporale dell'ultimo millisecondo del giorno contenuto in **time**. Il formato di output predefinito sarà il formato **TimestampFormat** impostato nello script.

```
dayend (timestamp [, shift = 0 [, dayoffset = 0]])
```

dayname

Questa funzione restituisce un valore che mostra la data con un valore numerico sottostante corrispondente a un indicatore temporale recante il primo millisecondo del giorno contenente **time**.

```
dayname (timestamp [, shift = 0 [, dayoffset = 0]])
```

Funzioni di numerazione del giorno

age

La funzione **age** restituisce l'età al momento dell'indicazione della data e dell'ora **timestamp** (in anni completi) di un soggetto nato nella data **date_of_birth**.

```
age (timestamp, date_of_birth)
```

networkdays

La funzione **networkdays** restituisce il numero di giorni lavorativi (dal lunedì al venerdì) compresi tra e inclusi in **start_date** e **end_date**, tenendo in considerazione qualsiasi eventuale valore di festività **holiday** nel calendario.

```
networkdays  (start:date, end_date {, holiday})
```

firstworkdate

La funzione **firstworkdate** restituisce la data di inizio più recente per fare in modo che il valore **no_of_workdays** (dal lunedì al venerdì) non termini oltre la data **end_date**, tenendo in considerazione tutte le festività eventualmente in calendario. **end_date** e **holiday** devono essere date o indicatori temporali validi.

```
firstworkdate (end_date, no_of_workdays {, holiday} )
```

lastworkdate

La funzione **lastworkdate** restituisce la data di fine più prossima per ottenere **no_of_workdays** (dal lunedì al venerdì) se si inizia dalla data **start_date** tenendo in considerazione tutte le festività **holiday** eventualmente in calendario. **start_date** e **holiday** devono essere date o indicatori temporali validi.

```
lastworkdate (start_date, no_of_workdays {, holiday})
```

daynumberofyear

Questa funzione calcola il numero del giorno dell'anno a cui è stato assegnato un indicatore temporale. Il calcolo viene eseguito partendo dal primo millisecondo del primo giorno dell'anno, tuttavia il primo mese può essere differito.

```
daynumberofyear (date[,firstmonth])
```

daynumberofquarter
Questa funzione calcola il numero del giorno del trimestre a cui è stato assegnato un indicatore temporale. Questa funzione viene utilizzata al momento di creare un Calendario principale.

```
daynumberofquarter (date[,firstmonth])
```

addmonths

Questa funzione restituisce la data che ricorre **n** mesi dopo **startdate** oppure, se **n** è negativo, la data che ricorre **n** mesi prima di **startdate**.

Sintassi:

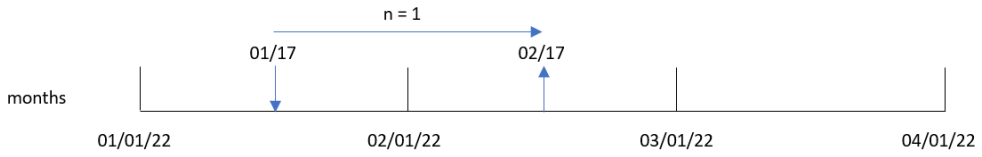
```
AddMonths(startdate, n , [ , mode])
```

Tipo di dati restituiti: duale

La funzione `addmonths()` aggiunge o sottrae un numero definito di mesi, `n`, da un `startdate` e restituisce la data risultante.

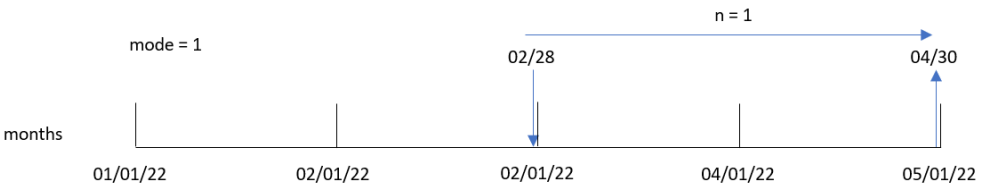
L'argomento `mode` avrà un impatto sui valori `startdate` non prima del 28 del mese. Impostando l'argomento `mode` a 1, la funzione `addmonths()` restituisce una data che equivale nella distanza relativa alla fine del mese come `startdate`.

Schema esemplificativo della funzione `addmonths()`



Ad esempio, il 28 febbraio è l'ultimo giorno del mese. Se nella funzione `addmonths()`, con un `mode` di 1, è utilizzato per restituire la data due mesi dopo, la funzione restituirà l'ultima data di aprile, il 30 aprile.

Schema di esempio della funzione `addmonths()`, con `mode=1`.



Argomenti

Argomento	Descrizione
startdate	La data di inizio come indicazione di data e ora, ad esempio '2012-10-12'.
n	Numero di mesi come numero intero positivo o negativo.

Argomento	Descrizione
mode	Specifica se il mese viene aggiunto relativamente all'inizio o alla fine del mese. La modalità predefinita è 0 per le aggiunte relative all'inizio del mese. Impostare la modalità su 1 per le aggiunte relative alla fine del mese. Quando la modalità è impostata su 1 e la data di input è il 28 o successiva, la funzione controlla quanti giorni mancano per raggiungere la fine del mese nella data di inizio. Sulla data restituita viene impostato lo stesso numero di giorni per raggiungere la fine del mese.

Casi di utilizzo

La funzione `addmonths()` verrà utilizzata comunemente in un'espressione per trovare una data un certo numero di mesi prima o dopo un periodo di tempo.

Ad esempio, la funzione `addmonths()` può essere utilizzata per identificare la data di fine dei contratti per la telefonia mobile.

Esempi di funzioni

Esempio	Risultato
`addmonths ('01/29/2003' ,3)`	Restituisce '04/29/2003'.
`addmonths ('01/29/2003',3,0)`	Restituisce '04/29/2003'.
`addmonths ('01/29/2003',3,1)`	Restituisce '04/28/2003'.
`addmonths ('01/29/2003',1,0)`	Restituisce '02/28/2003'.
`addmonths ('01/29/2003',1,1)`	Restituisce '02/26/2003'.
`addmonths ('02/28/2003',1,0)`	Restituisce '03/28/2003'.
`addmonths ('02/28/2003',1,1)`	Restituisce '03/31/2003'.
`addmonths ('01/29/2003',-3)`	Restituisce '10/29/2002'.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `two_months_later`, che restituisce la data per due mesi dopo l'avvenuta transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        addmonths(date,2) as two_months_later
        ;
Load
*
Inline
[
id,date,amount
8188,'01/10/2020',37.23
8189,'02/28/2020',17.17
8190,'04/09/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'02/02/2022',46.23
8205,'02/26/2022',84.21
8206,'03/07/2022',96.24
8207,'03/11/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

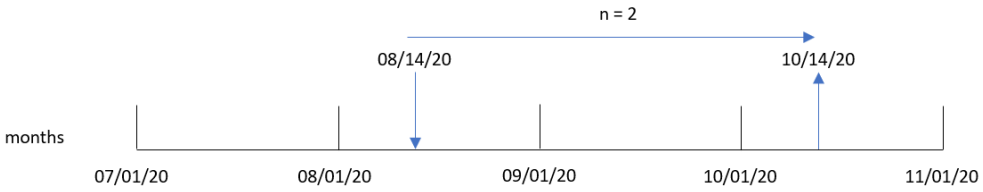
- `date`
- `two_months_later`

Tabella dei risultati

date	two_months_later
01/10/2020	03/10/2020
02/28/2020	04/28/2020
04/09/2020	06/09/2020
04/16/2020	06/16/2020
05/21/2020	07/21/2020
08/14/2020	10/14/2020
10/07/2020	12/07/2020
12/05/2020	02/05/2021
01/22/2021	03/22/2021
02/03/2021	04/03/2021
03/17/2021	05/17/2021
04/23/2021	06/23/2021
05/04/2021	07/04/2021
06/30/2021	08/30/2021
07/26/2021	09/26/2021
12/27/2021	02/27/2022
02/02/2022	04/02/2022
02/26/2022	04/26/2022
03/07/2022	05/07/2022
03/11/2022	05/11/2022

Il campo `two_months_later` viene creato nell'istruzione LOAD precedente utilizzando la funzione `addmonths()`. Il primo argomento fornito identifica la data da valutare. Il secondo argomento è il numero di mesi da aggiungere o sottrarre all'intervallo `startdate`. In questo caso, viene fornito il valore 2.

Schema della funzione addmonths(), esempio senza argomenti aggiuntivi



La transazione 8193 è avvenuta il 14 agosto. Pertanto, la funzione `addmonths()` restituisce il 14 ottobre 2020 per il campo `two_months_later`.

Esempio 2 - Fine mese relativo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente una serie di transazioni di fine mese nel 2022, che viene caricato in una tabella chiamata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `relative_two_months_prior`, che restituisce la data di fine mese relativa ai due mesi precedenti la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        addmonths(date,-2,1) as relative_two_months_prior
        ;
Load
*
Inline
[
id,date,amount
8188,'01/28/2022',37.23
8189,'01/31/2022',57.54
8190,'02/28/2022',17.17
8191,'04/29/2022',88.27
8192,'04/30/2022',57.42
8193,'05/31/2022',53.80
8194,'08/14/2022',82.06
8195,'10/07/2022',40.39
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

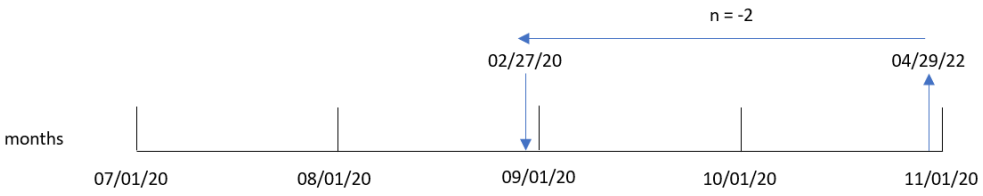
- `date`
- `relative_two_months_prior`

Tabella dei risultati

date	relative_two_months_prior
01/28/2022	11/27/2021
01/31/2022	11/30/2021
02/28/2022	12/31/2021
04/29/2022	02/27/2022
04/30/2022	02/28/2022
05/31/2022	03/31/2022
08/14/2022	06/14/2022
10/07/2022	08/07/2022

Il campo `relative_two_months_prior` viene creato nell'istruzione LOAD precedente utilizzando la funzione `addmonths()`. Il primo argomento fornito identifica la data da valutare. Il secondo argomento è il numero di mesi da aggiungere o sottrarre all'intervallo `startdate`. In questo caso, viene fornito il valore -2. L'ultimo argomento è la modalità, con un valore di 1, che costringe la funzione a calcolare la data di fine mese relativa per tutte le date maggiori o uguali a 28.

Schema della funzione `addmonths()`, esempio con `n=-2`.



La transazione 8191 avrà luogo il 29 aprile 2022. Inizialmente, due mesi prima si sarebbe impostato il mese a febbraio. Quindi, dato che il terzo argomento della funzione imposta la modalità a 1 e il valore del giorno è successivo al 27, la funzione calcola il valore relativo alla fine del mese. La funzione identifica che il 29 è il penultimo giorno di aprile e pertanto restituisce il penultimo giorno di febbraio, il 27.

Esempio 3 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce la data di due mesi dopo la transazione viene creato come misura in un oggetto grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'01/10/2020',37.23
8189,'02/28/2020',17.17
8190,'04/09/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'02/02/2022',46.23
8205,'02/26/2022',84.21
8206,'03/07/2022',96.24
8207,'03/11/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Creare la seguente misura:

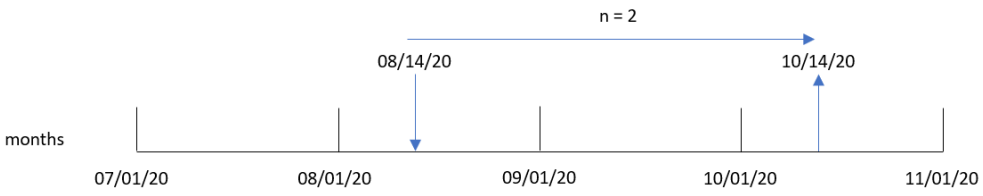
```
=addmonths(date,2)
```

Tabella dei risultati

date	=addmonths(date,2)
01/10/2020	03/10/2020
02/28/2020	04/28/2020
04/09/2020	06/09/2020
04/16/2020	06/16/2020
05/21/2020	07/21/2020
08/14/2020	10/14/2020
10/07/2020	12/07/2020
12/05/2020	02/05/2021
01/22/2021	03/22/2021
02/03/2021	04/03/2021
03/17/2021	05/17/2021
04/23/2021	06/23/2021
05/04/2021	07/04/2021
06/30/2021	08/30/2021
07/26/2021	09/26/2021
12/27/2021	02/27/2022
02/02/2022	04/02/2022
02/26/2022	04/26/2022
03/07/2022	05/07/2022
03/11/2022	05/11/2022

La misura `two_months_later` viene creata nell'oggetto grafico utilizzando la funzione `addmonths()`. Il primo argomento fornito identifica la data da valutare. Il secondo argomento è il numero di mesi da aggiungere o sottrarre all'intervallo `startdate`. In questo caso, viene fornito il valore 2.

Schema della funzione `addmonths()`, esempio di oggetto grafico



La transazione 8193 è avvenuta il 14 agosto. Pertanto, la funzione `addmonths()` restituisce il 14 ottobre 2020 per il campo `two_months_later`.

Esempio 4 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Mobile_Plans`.
- Informazioni con l'ID del contratto, la data di inizio, la durata del contratto e il canone mensile.

L'utente finale vorrebbe un oggetto grafico che visualizzi, per ID contratto, la data di scadenza di ogni contratto telefonico.

Script di caricamento

```
Mobile_Plans:
Load
*
Inline
[
contract_id,start_date,contract_length,monthly_fee
8188,'01/13/2020',18,37.23
8189,'02/26/2020',24,17.17
8190,'03/27/2020',36,88.27
8191,'04/16/2020',24,57.42
8192,'05/21/2020',24,53.80
8193,'08/14/2020',12,82.06
8194,'10/07/2020',18,40.39
8195,'12/05/2020',12,87.21
8196,'01/22/2021',12,95.93
8197,'02/03/2021',18,45.89
8198,'03/17/2021',24,36.23
8199,'04/23/2021',24,25.66
8200,'05/04/2021',12,82.77
8201,'06/30/2021',12,69.98
8202,'07/26/2021',12,76.11
8203,'12/27/2021',36,25.12
8204,'06/06/2022',24,46.23
8205,'07/18/2022',12,84.21
8206,'11/14/2022',12,96.24
8207,'12/12/2022',18,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- contract_id
- start_date
- contract_length

Crea la seguente misura per calcolare la data di scadenza di ogni contratto:

```
=addmonths(start_date,contract_length, 0)
```

Tabella dei risultati

contract_id	start_date	contract_length	=addmonths(start_date,contract_length,0)
8188	01/13/2020	18	07/13/2021
8189	02/26/2020	24	02/26/2022
8190	03/27/2020	36	03/27/2023
8191	04/16/2020	24	04/16/2022
8192	05/21/2020	24	05/21/2022
8193	08/14/2020	12	08/14/2021
8194	10/07/2020	18	04/07/2022
8195	12/05/2020	12	12/05/2021
8196	01/22/2021	12	01/22/2022
8197	02/03/2021	18	08/03/2022
8198	03/17/2021	24	03/17/2023
8199	04/23/2021	24	04/23/2023
8200	05/04/2021	12	05/04/2022
8201	06/30/2021	12	06/30/2022
8202	07/26/2021	12	07/26/2022
8203	12/27/2021	36	12/27/2024
8204	06/06/2022	24	06/06/2024
8205	07/18/2022	12	07/18/2023
8206	11/14/2022	12	11/14/2023
8207	12/12/2022	18	06/12/2024

addyears

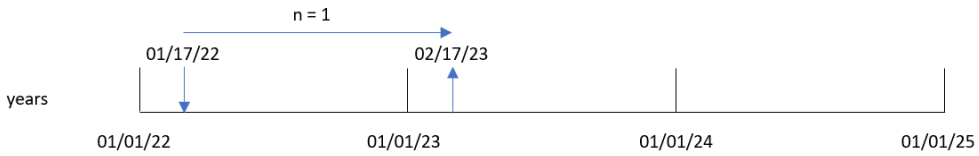
Questa funzione restituisce la data che ricorre **n** anni dopo **startdate** oppure, se **n** è negativo, la data che ricorre **n** anni prima di **startdate**.

Sintassi:

```
AddYears(startdate, n)
```

Tipo di dati restituiti: duale

Schema esemplificativo della funzione `addyears()`



La funzione `addyears()` aggiunge o sottrae un numero definito di anni, `n`, da un valore di `startdate`. Quindi restituisce la data risultante.

Argomenti

Argomento	Descrizione
startdate	La data di inizio come indicazione di data e ora, ad esempio '2012-10-12'.
n	Numero di anni come numero intero positivo o negativo.

Esempi di funzioni

Esempio	Risultato
`addyears ('01/29/2010',3)`	Restituisce '01/29/2013'.
`addyears ('01/29/2010',-1)`	Restituisce '01/29/2009'.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Esempio semplice

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `two_years_later`, che restituisce la data per due anni dopo l'avvenuta transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        addyears(date,2) as two_years_later
        ;
Load
*
Inline
[
id,date,amount
8188,'01/10/2020',37.23
8189,'02/28/2020',17.17
8190,'04/09/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'02/02/2022',46.23
8205,'02/26/2022',84.21
8206,'03/07/2022',96.24
8207,'03/11/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

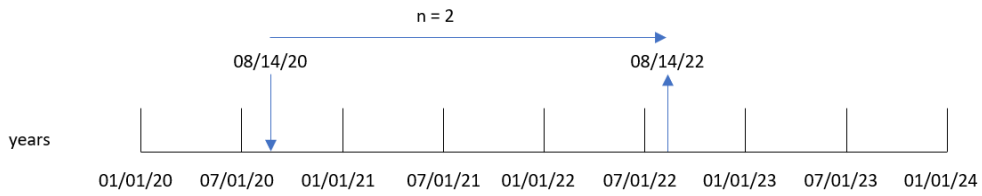
- `date`
- `two_years_later`

Tabella dei risultati

date	two_years_later
01/10/2020	01/10/2022
02/28/2020	02/28/2022
04/09/2020	04/09/2022
04/16/2020	04/16/2022
05/21/2020	05/21/2022
08/14/2020	08/14/2022
10/07/2020	10/07/2022
12/05/2020	12/05/2022
01/22/2021	01/22/2023
02/03/2021	02/03/2023
03/17/2021	03/17/2023
04/23/2021	04/23/2023
05/04/2021	05/04/2023
06/30/2021	06/30/2023
07/26/2021	07/26/2023
12/27/2021	12/27/2023
02/02/2022	02/02/2024
02/26/2022	02/26/2024
03/07/2022	03/07/2024
03/11/2022	03/11/2024

Il campo `two_years_later` viene creato nell'istruzione LOAD precedente utilizzando la funzione `addyears()`. Il primo argomento fornito identifica la data da valutare. Il secondo argomento è il numero di anni da aggiungere o sottrarre dalla data di inizio. In questo caso, viene fornito il valore 2.

Schema della funzione `addyears()`, esempio di base



La transazione 8193 è avvenuta il 14 agosto 2020. Pertanto, la funzione `addyears()` restituisce il 14 agosto 2022 per il campo `two_years_later`.

Esempio 2 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

In un oggetto grafico, crea una misura, `prior_year_date`, che restituisce la data di un anno precedente a quella in cui avviene la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'01/10/2020',37.23
8189,'02/28/2020',17.17
8190,'04/09/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
```

```
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'02/02/2022',46.23
8205,'02/26/2022',84.21
8206,'03/07/2022',96.24
8207,'03/11/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Crea la seguente misura per calcolare la data di un anno prima di ogni transazione:

```
=addyears(date,-1)
```

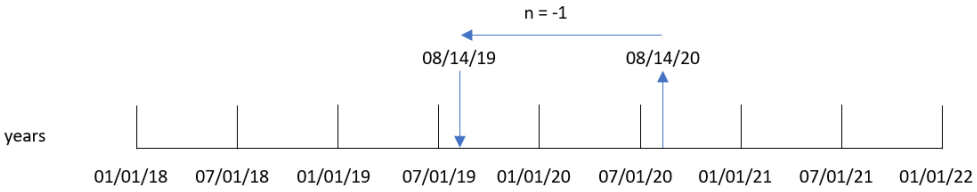
Tabella dei risultati

date	=addyears(date,-1)
01/10/2020	01/10/2019
02/28/2020	02/28/2019
04/09/2020	04/09/2019
04/16/2020	04/16/2019
05/21/2020	05/21/2019
08/14/2020	08/14/2019
10/07/2020	10/07/2019
12/05/2020	12/05/2019
01/22/2021	01/22/2020
02/03/2021	02/03/2020
03/17/2021	03/17/2020
04/23/2021	04/23/2020
05/04/2021	05/04/2020
06/30/2021	06/30/2020
07/26/2021	07/26/2020
12/27/2021	12/27/2020
02/02/2022	02/02/2021
02/26/2022	02/26/2021

date	=addyears(date,-1)
03/07/2022	03/07/2021
03/11/2022	03/11/2021

La misura `one_year_prior` viene creata nell'oggetto grafico utilizzando la funzione `addyears()`. Il primo argomento fornito identifica la data da valutare. Il secondo argomento è il numero di anni da aggiungere o sottrarre da `startdate`. In questo caso, viene fornito il valore -1.

Schema della funzione `addyears()`, esempio di oggetto grafico



La transazione 8193 è avvenuta il 14 agosto. Pertanto, la funzione `addyears()` restituisce il 14 agosto 2019 per il campo `one_year_prior`.

Esempio 3 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Warranties`.
- Informazioni con l'ID del prodotto, la data di acquisto, la durata della garanzia e il prezzo di acquisto.

L'utente finale vorrebbe un oggetto grafico che visualizzi, per ID prodotto, la data di cessazione della garanzia di ogni prodotto.

Script di caricamento

```
Warranties:
Load
*
Inline
[
product_id,purchase_date,warranty_length,purchase_price
8188,'01/13/2020',4,32000
8189,'02/26/2020',2,28000
8190,'03/27/2020',3,41000
8191,'04/16/2020',4,17000
8192,'05/21/2020',2,25000
```

```
8193,'08/14/2020',1,59000
8194,'10/07/2020',2,12000
8195,'12/05/2020',3,12000
8196,'01/22/2021',4,24000
8197,'02/03/2021',1,50000
8198,'03/17/2021',2,80000
8199,'04/23/2021',3,10000
8200,'05/04/2021',4,30000
8201,'06/30/2021',3,30000
8202,'07/26/2021',4,20000
8203,'12/27/2021',4,10000
8204,'06/06/2022',2,25000
8205,'07/18/2022',1,32000
8206,'11/14/2022',1,30000
8207,'12/12/2022',4,22000
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `product_id`
- `purchase_date`
- `warranty_length`

Crea la seguente misura per calcolare la data di scadenza della garanzia di ogni prodotto:

```
=addyears(purchase_date,warranty_length)
```

Tabella dei risultati

product_id	purchase_date	warranty_length	=addyears(purchase_date,warranty_length)
8188	01/13/2020	4	01/13/2024
8189	02/26/2020	2	02/26/2022
8190	03/27/2020	3	03/27/2023
8191	04/16/2020	4	04/16/2024
8192	05/21/2020	2	05/21/2022
8193	08/14/2020	1	08/14/2021
8194	10/07/2020	2	10/07/2022
8195	12/05/2020	3	12/05/2023
8196	01/22/2021	4	01/22/2025
8197	02/03/2021	1	02/03/2022
8198	03/17/2021	2	03/17/2023
8199	04/23/2021	3	04/23/2024

product_id	purchase_date	warranty_length	=addyears(purchase_date,warranty_length)
8200	05/04/2021	4	05/04/2025
8201	06/30/2021	3	06/30/2024
8202	07/26/2021	4	07/26/2025
8203	12/27/2021	4	12/27/2025
8204	06/06/2022	2	06/06/2024
8205	07/18/2022	1	07/18/2023
8206	11/14/2022	1	11/14/2023
8207	12/12/2022	4	12/12/2026

age

La funzione **age** restituisce l'età al momento dell'indicazione della data e dell'ora **timestamp** (in anni completi) di un soggetto nato nella data **date_of_birth**.

Sintassi:

```
age(timestamp, date_of_birth)
```

Può essere un'espressione.

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
timestamp	L'indicatore temporale, o l'espressione che risolve l'indicatore temporale, fino al quale calcolare il numero di anni completati.
date_of_birth	Data di nascita della persona di cui si sta calcolando l'età. Può essere un'espressione.

Esempi e risultati:
In questi esempi viene utilizzato il formato della data **DD/MM/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di script

Esempio	Risultato
`age('25/01/2014', '29/10/2012')`	Restituisce 1.
`age('29/10/2014', '29/10/2012')`	Restituisce 2.

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
Employees:
LOAD * INLINE [
Member|DateOfBirth
John|28/03/1989
Linda|10/12/1990
Steve|5/2/1992
Birg|31/3/1993
Raj|19/5/1994
Prita|15/9/1994
Su|11/12/1994
Goran|2/3/1995
Sunny|14/5/1996
Ajoa|13/6/1996
Daphne|7/7/1998
Biffy|4/8/2000
] (delimiter is |);
AgeTable:
Load *,
age('20/08/2015', DateOfBirth) As Age
Resident Employees;
Drop table Employees;
```

La tabella risultante mostra i valori restituiti in age per ciascun record della tabella.

Tabella dei risultati

Member	DateOfBirth	Age
John	28/03/1989	26
Linda	10/12/1990	24
Steve	5/2/1992	23
Birg	31/3/1993	22
Raj	19/5/1994	21
Prita	15/9/1994	20
Su	11/12/1994	20
Goran	2/3/1995	20
Sunny	14/5/1996	19
Ajoa	13/6/1996	19
Daphne	7/7/1998	17
Biffy	4/8/2000	15

converttolocaltime

Converte un indicatore temporale UTC o GMT in ora locale come valore duale. Il luogo può essere qualsiasi città, località o fuso orario nel mondo.

Sintassi:

```
ConvertToLocalTime(timestamp [, place [, ignore_dst=false]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
timestamp	L'indicatore temporale, o l'espressione che restituisce un indicatore temporale, da convertire.
place	Una località o un fuso orario contenuti nella tabella seguente relativa alle località e ai fusi orari validi. In alternativa, è possibile utilizzare GMT o UTC per definire l'ora locale. I valori e gli intervalli di differimento temporale seguenti sono validi: • GMT • GMT-12:00 - GMT-01:00 • GMT+01:00 - GMT+14:00 • UTC • UTC-12:00 - UTC-01:00 • UTC+01:00 - UTC+14:00 *È possibile utilizzare differimenti temporali standard. Non è possibile utilizzare un differimento temporale arbitrario, ad esempio GMT-04:27.*
ignore_dst	Impostare su True per ignorare DST (ora legale). Impostare su False per regolare per l'ora legale.

Località e fusi orari validi

A-C	D-K	L-R	S-Z
Abu Dhabi	Darwin	La Paz	Samoa
Adelaide	Dhaka	Lima	Santiago
Alaska	Eastern Time (US & Canada)	Lisbon	Sapporo
Amsterdam	Edinburgh	Ljubljana	Sarajevo

A-C	D-K	L-R	S-Z
Arizona	Ekaterinburg	London	Saskatchewan
Astana	Fiji	Madrid	Seoul
Athens	Georgetown	Magadan	Singapore
Atlantic Time (Canada)	Greenland	Mazatlan	Skopje
Auckland	Greenwich Mean Time : Dublin	Melbourne	Sofia
Azores	Guadalajara	Mexico City	Solomon Is.
Baghdad	Guam	Mid-Atlantic	Sri Jayawardenepura
Baku	Hanoi	Minsk	St. Petersburg
Bangkok	Harare	Monrovia	Stockholm
Beijing	Hawaii	Monterrey	Sydney
Belgrade	Helsinki	Moscow	Taipei
Berlin	Hobart	Mountain Time (US & Canada)	Tallinn
Bern	Hong Kong	Mumbai	Tashkent
Bogota	Indiana (East)	Muscat	Tbilisi
Brasilia	International Date Line West	Nairobi	Tehran
Bratislava	Irkutsk	New Caledonia	Tokyo
Brisbane	Islamabad	New Delhi	Urumqi
Brussels	Istanbul	Newfoundland	Warsaw
Bucharest	Jakarta	Novosibirsk	Wellington
Budapest	Jerusalem	Nuku'alofa	West Central Africa
Buenos Aires	Kabul	Osaka	Vienna
Cairo	Kamchatka	Pacific Time (US & Canada)	Vilnius
Canberra	Karachi	Paris	Vladivostok
Cape Verde Is.	Kathmandu	Perth	Volgograd
Caracas	Kolkata	Port Moresby	Yakutsk
Casablanca	Krasnoyarsk	Prague	Yerevan
Central America	Kuala Lumpur	Pretoria	Zagreb

A-C	D-K	L-R	S-Z
Central Time (US & Canada)	Kuwait	Quito	-
Chennai	Kyiv	Riga	-
Chihuahua	-	Riyadh	-
Chongqing	-	Rome	-
Copenhagen	-	-	-

Esempi e risultati:

Esempi di script

Esempio	Risultato
`ConvertToLocalTime('2007-11-10 23:59:00','Paris')`	Restituisce '2007-11-11 00:59:00' e la corrispondente rappresentazione interna di data e ora.
`ConvertToLocalTime(UTC(), 'GMT-05:00')`	Restituisce l'ora della costa Orientale degli Stati Uniti, ad esempio l'ora di New York.
`ConvertToLocalTime(UTC(), 'GMT-05:00', True)`	Restituisce l'ora della costa Orientale degli Stati Uniti, ad esempio l'ora di New York, senza adattamento dell'ora legale.

day

Questa funzione restituisce un numero intero che rappresenta il giorno in cui la frazione di **expression** viene interpretata come data in base all'interpretazione numerica standard.

La funzione restituisce il giorno del mese per una data particolare. In genere viene utilizzata per derivare un campo giorno come parte di una dimensione di calendario.

Sintassi:

```
day(expression)
```

Tipo di dati restituiti: numero intero

Esempi di funzioni

Esempio	Risultato
`day( 1971-10-12 )`	restituisce 12
`day( 35648 )`	restituisce 6 poiché 35648 = 1997-08-06

Esempio 1 – set di dati DateFormat (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di date denominato `Master_Calendar`. La variabile di sistema `DateFormat` è impostata su GG/MM/AAAA.
- Un caricamento precedente che crea un campo aggiuntivo, denominato `day_of_month`, mediante la funzione `day()`.
- Un campo aggiuntivo, denominato `long_date`, che utilizza la funzione `date()` per esprimere il nome mese completo.

Script di caricamento

```
SET DateFormat='DD/MM/YYYY';

Master_Calendar:
Load
        date,
        date(date,'dd-MMMM-YYYY') as long_date,
        day(date) as day_of_month
Inline
[
date
03/11/2022
03/12/2022
03/13/2022
03/14/2022
03/15/2022
03/16/2022
03/17/2022
03/18/2022
03/19/2022
03/20/2022
03/21/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `long_date`
- `day_of_month`

Tabella dei risultati

data	long_date	day_of_month
03/11/2022	11-March- 2022	11
03/12/2022	12-March- 2022	12
03/13/2022	13-March- 2022	13
03/14/2022	14-March- 2022	14
03/15/2022	15-March- 2022	15
03/16/2022	16-March- 2022	16
03/17/2022	17-March- 2022	17
03/18/2022	18-March- 2022	18
03/19/2022	19-March- 2022	19
03/20/2022	20-March- 2022	20
03/21/2022	21-March- 2022	21

Il giorno del mese viene valutato correttamente dalla funzione `day()` nello script.

Esempio 2 – Date ANSI (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di date denominato `Master_Calendar`. Viene utilizzata la variabile di sistema DateFormat GG/MM/AAAA. Tuttavia, le date incluse nel set di dati sono nel formato data standard ANSI.
- Un caricamento precedente che crea un campo aggiuntivo, denominato `day_of_month`, mediante la funzione `date()`.
- Un campo aggiuntivo, denominato `long_date`, che utilizza la funzione `date()` per esprimere la data con il nome mese completo.

Script di caricamento

```
SET DateFormat='DD/MM/YYYY';
Master_Calendar:
Load
       date,
       date(date,'dd-MMMM-YYYY') as long_date,
       day(date) as day_of_month
```

```
Inline
[
date
2022-03-11
2022-03-12
2022-03-13
2022-03-14
2022-03-15
2022-03-16
2022-03-17
2022-03-18
2022-03-19
2022-03-20
2022-03-21
 ];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `long_date`
- `day_of_month`

Tabella dei risultati

data	long_date	day_of_month
03/11/2022	11-March- 2022	11
03/12/2022	12-March- 2022	12
03/13/2022	13-March- 2022	13
03/14/2022	14-March- 2022	14
03/15/2022	15-March- 2022	15
03/16/2022	16-March- 2022	16
03/17/2022	17-March- 2022	17
03/18/2022	18-March- 2022	18
03/19/2022	19-March- 2022	19
03/20/2022	20-March- 2022	20
03/21/2022	21-March- 2022	21

Il giorno del mese viene valutato correttamente dalla funzione `day()` nello script.

Esempio 3 – Date non formattate (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di date denominato `Master_Calendar`. Viene utilizzata la variabile di sistema `DateFormat` GG/MM/AAAA.
- Un caricamento precedente che crea un campo aggiuntivo, denominato `day_of_month`, mediante la funzione `day()`.
- La data non formattata originale, denominata `unformatted_date`.
- Un campo aggiuntivo, denominato `long_date`, che utilizza `date()` viene impiegato per convertire la data numerica in un campo data formattato.

Script di caricamento

```
SET DateFormat='DD/MM/YYYY';

Master_Calendar:
Load
      unformatted_date,
      date(unformatted_date,'dd-MMMM-YYYY') as long_date,
      day(date) as day_of_month

Inline
[
unformatted_date
44868
44898
44928
44958
44988
45018
45048
45078
45008
45038
45068
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- unformatted_date
- long_date
- day_of_month

Tabella dei risultati

unformatted_date	long_date	day_of_month
44868	03-November- 2022	3
44898	03-December- 2022	3
44928	02-January- 2023	2
44958	01-February- 2023	1
44988	03-March- 2023	3
45008	23-March- 2023	23
45018	02-April- 2023	2
45038	22-April- 2023	22
45048	02-May- 2023	2
45068	22-May- 2023	22
45078	01-June- 2023	1

Il giorno del mese viene valutato correttamente dalla funzione `day()` nello script.

Esempio 4 – Calcolo del mese di scadenza (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati degli ordini effettuati a marzo denominato `Orders`. La tabella contiene tre campi:
  - id
  - order_date
  - importo

Script di caricamento

```
Orders:
Load
        id,
        order_date,
        amount
Inline
```

```
[
id,order_date,amount
1,03/01/2022,231.24
2,03/02/2022,567.28
3,03/03/2022,364.28
4,03/04/2022,575.76
5,03/05/2022,638.68
6,03/06/2022,785.38
7,03/07/2022,967.46
8,03/08/2022,287.67
9,03/09/2022,764.45
10,03/10/2022,875.43
11,03/11/2022,957.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `order_date`.

Per calcolare la data di scadenza, creare questa misura: `=day(order_date+5)`.

Tabella dei risultati

order_date	=day(order_date+5)
03/11/2022	16
03/12/2022	17
03/13/2022	18
03/14/2022	19
03/15/2022	20
03/16/2022	21
03/17/2022	22
03/18/2022	23
03/19/2022	24
03/20/2022	25
03/21/2022	26

La funzione `day()` attualmente determina che un ordine effettuato l'11 marzo verrà consegnato il 16 sulla base di una tempistica di consegna di 5 giorni.

dayend

Questa funzione restituisce un valore corrispondente a un indicatore temporale dell'ultimo millisecondo del giorno contenuto in **time**. Il formato di output predefinito sarà il formato **TimestampFormat** impostato nello script.

Sintassi:

```
DayEnd(time[, [period_no[, day_start]])
```

Casi di utilizzo

La funzione `dayend()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione di giorno non ancora trascorsa. Ad esempio, per calcolare il totale delle spese ancora da sostenere durante la giornata.

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
time	L'indicatore temporale da valutare.
period_no	**period_no** è un numero intero o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica il giorno che contiene il valore **time**. I valori negativi di **period_no** indicano i giorni precedenti, mentre i valori positivi indicano i giorni successivi.
day_start	Per specificare che i giorni non iniziano a mezzanotte, indicare un differimento come frazione di un giorno in **day_start**. Ad esempio, 0.125 per denotare 3:00 AM. In altre parole, per creare l'offset dividere l'ora di inizio per 24 ore. Ad esempio, per un giorno che inizia alle 7:00 del mattino, utilizzare la frazione 7/24.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
dayend('01/25/2013 16:45:00')	Returns 01/25/2013 23:59:59. PM
dayend('01/25/2013 16:45:00', -1)	Returns 01/24/2013 23:59:59. PM
dayend('01/25/2013 16:45:00', 0, 0.5)	Returns 01/26/2013 11:59:59. PM

Esempio 1 - Script base

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un elenco di date viene caricato in una tabella denominata "Calendario".
- La variabile di sistema `DateFormat` predefinita `(MM/DD/YYYY)`.
- Un caricamento precedente per creare un campo aggiuntivo, denominato '`EOD_timestamp`', mediante la funzione `dayend()`.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Calendar:
    Load
      date,
      dayend(date) as EOD_timestamp
      ;
Load
date
Inline
[
date
03/11/2022 1:47:15 AM
03/12/2022 4:34:58 AM
03/13/2022 5:15:55 AM
03/14/2022 9:25:14 AM
03/15/2022 10:06:54 AM
03/16/2022 10:44:42 AM
03/17/2022 11:33:30 AM
03/18/2022 12:58:14 PM
03/19/2022 4:23:12 PM
03/20/2022 6:42:15 PM
03/21/2022 7:41:16 PM
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `EOD_timestamp`

Tabella dei risultati

data	EOD_timestamp
03/11/2022 1:47:15 AM	3/11/2022 11:59:59 PM
03/12/2022 4:34:58 AM	3/12/2022 11:59:59 PM
03/13/2022 5:15:55 AM	3/13/2022 11:59:59 PM
03/14/2022 9:25:14 AM	3/14/2022 11:59:59 PM
03/15/2022 10:06:54 AM	3/15/2022 11:59:59 PM
03/16/2022 10:44:42 AM	3/16/2022 11:59:59 PM
03/17/2022 11:33:30 AM	3/17/2022 11:59:59 PM
03/18/2022 12:58:14 PM	3/18/2022 11:59:59 PM
03/19/2022 4:23:12 PM	3/19/2022 11:59:59 PM
03/20/2022 6:42:15 PM	3/20/2022 11:59:59 PM
03/21/2022 7:41:16 PM	3/21/2022 11:59:59 PM

Come indicato nella tabella sopra, la fine del timestamp giorno viene generata per ciascuna data nel nostro set di dati. Il timestamp è nel formato della variabile di sistema `TimestampFormat M/D/YYYY h:mm:ss[.fff] TT`.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Si caricherà un set di dati contenente le prenotazioni di servizio in una tabella denominata 'Servizi'.

Il set di dati include i seguenti campi:

- `service_id`
- `service_date`
- `amount`

Si creeranno due nuovi campi nella tabella:

- `deposit_due_date`: La data in cui si dovrebbe ricevere il deposito. Questa è la fine del giorno tre giorni prima del `service_date`.
- `final_payment_due_date`: La data in cui si dovrebbe ricevere il pagamento finale. Questa è la fine del giorno sette giorni dopo il `service_date`.

I due campi sopra sono creati in un carico precedente mediante la funzione `dayend()` e possono fornire i primi due parametri, `time` e `period_no`.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Services:
   Load
     *,
     dayend(service_date,-3) as deposit_due_date,
     dayend(service_date,7) as final_payment_due_date
     ;
Load
service_id,
service_date,
amount
Inline
[
service_id, service_date,amount
1,03/11/2022 9:25:14 AM,231.24
2,03/12/2022 10:06:54 AM,567.28
3,03/13/2022 10:44:42 AM,364.28
4,03/14/2022 11:33:30 AM,575.76
5,03/15/2022 12:58:14 PM,638.68
6,03/16/2022 4:23:12 PM,785.38
7,03/17/2022 6:42:15 PM,967.46
8,03/18/2022 7:41:16 PM,287.67
9,03/19/2022 8:14:15 PM,764.45
10,03/20/2022 9:23:51 PM,875.43
11,03/21/2022 10:04:41 PM,957.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `service_date`
- `deposit_due_date`
- `final_payment_due_date`

Tabella dei risultati

service_date	deposit_due_date	final_payment_due_date
03/11/2022 9:25:14 AM	3/8/2022 11:59:59 PM	3/18/2022 11:59:59 PM

service_date	deposit_due_date	final_payment_due_date
03/12/2022 10:06:54 AM	3/9/2022 11:59:59 PM	3/19/2022 11:59:59 PM
03/13/2022 10:44:42 AM	3/10/2022 11:59:59 PM	3/20/2022 11:59:59 PM
03/14/2022 11:33:30 AM	3/11/2022 11:59:59 PM	3/21/2022 11:59:59 PM
03/15/2022 12:58:14 PM	3/12/2022 11:59:59 PM	3/22/2022 11:59:59 PM
03/16/2022 4:23:12 PM	3/13/2022 11:59:59 PM	3/23/2022 11:59:59 PM
03/17/2022 6:42:15 PM	3/14/2022 11:59:59 PM	3/24/2022 11:59:59 PM
03/18/2022 7:41:16 PM	3/15/2022 11:59:59 PM	3/25/2022 11:59:59 PM
03/19/2022 8:14:15 PM	3/16/2022 11:59:59 PM	3/26/2022 11:59:59 PM
03/20/2022 9:23:51 PM	3/17/2022 11:59:59 PM	3/27/2022 11:59:59 PM
03/21/2022 10:04:41 PM	3/18/2022 11:59:59 PM	3/28/2022 11:59:59 PM

I valori dei nuovi campi sono in `TimestampFormat M/D/YYYY h:mm:ss[.fff] TT`. Essendo stata usata la funzione `dayend()`, i valori data e ora sono tutti all'ultimo millisecondo del giorno.

I valori della data di scadenza del deposito sono tre giorni prima della data del servizio perché il secondo argomento passato nella funzione `dayend()` è negativo.

I valori della data di scadenza del pagamento finale sono sette giorni dopo la data del servizio perché il secondo argomento passato nella funzione `dayend()` è positivo.

Esempio 3 – day_start script

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Il set di dati e lo scenario utilizzati in questo esempio sono gli stessi dell'esempio precedente.

Come nell'esempio precedente, si creeranno due nuovi campi:

- `deposit_due_date`: La data in cui si dovrebbe ricevere il deposito. Questa è la fine del giorno tre giorni prima del `service_date`.
- `final_payment_due_date`: La data in cui si dovrebbe ricevere il pagamento finale. Questa è la fine del giorno sette giorni dopo il `service_date`.

Tuttavia, la propria azienda desidera operare in base a una policy in cui il giorno lavorativo inizia alle 17:00 e termina alle 17:00 del giorno successivo. La propria azienda può quindi monitorare le transazioni che si verificano in tali ore di lavoro.

Per ottenere questi requisiti, i due campi sopra vengono creati in un carico precedente usando la funzione `dayend()` e impiegando tutti e tre gli argomenti, `time`, `period_no` e `day_start`.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Services:
   Load
     *,
     dayend(service_date,-3,17/24) as deposit_due_date,
     dayend(service_date,7,17/24) as final_payment_due_date
     ;
Load
service_id,
service_date,
amount
Inline
[
service_id, service_date,amount
1,03/11/2022 9:25:14 AM,231.24
2,03/12/2022 10:06:54 AM,567.28
3,03/13/2022 10:44:42 AM,364.28
4,03/14/2022 11:33:30 AM,575.76
5,03/15/2022 12:58:14 PM,638.68
6,03/16/2022 4:23:12 PM,785.38
7,03/17/2022 6:42:15 PM,967.46
8,03/18/2022 7:41:16 PM,287.67
9,03/19/2022 8:14:15 PM,764.45
10,03/20/2022 9:23:51 PM,875.43
11,03/21/2022 10:04:41 PM,957.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- service_date
- deposit_due_date
- final_payment_due_date

Tabella dei risultati

service_date	deposit_due_date	final_payment_due_date
03/11/2022 9:25:14 AM	3/8/2022 4:59:59 PM	3/18/2022 4:59:59 PM
03/12/2022 10:06:54 AM	3/9/2022 4:59:59 PM	3/19/2022 4:59:59 PM
03/13/2022 10:44:42 AM	3/10/2022 4:59:59 PM	3/20/2022 4:59:59 PM
03/14/2022 11:33:30 AM	3/11/2022 4:59:59 PM	3/21/2022 4:59:59 PM
03/15/2022 12:58:14 PM	3/12/2022 4:59:59 PM	3/22/2022 4:59:59 PM
03/16/2022 4:23:12 PM	3/13/2022 4:59:59 PM	3/23/2022 4:59:59 PM

service_date	deposit_due_date	final_payment_due_date
03/17/2022 6:42:15 PM	3/14/2022 4:59:59 PM	3/24/2022 4:59:59 PM
03/18/2022 7:41:16 PM	3/15/2022 4:59:59 PM	3/25/2022 4:59:59 PM
03/19/2022 8:14:15 PM	3/16/2022 4:59:59 PM	3/26/2022 4:59:59 PM
03/20/2022 9:23:51 PM	3/17/2022 4:59:59 PM	3/27/2022 4:59:59 PM
03/21/2022 10:04:41 PM	3/18/2022 4:59:59 PM	3/28/2022 4:59:59 PM

Mentre le date rimangono le stesse dell'Esempio 2, le date ora hanno un timestamp dell'ultimo millisecondo prima delle 17:00 a causa del fatto che il valore del terzo argomento, `day_start`, trasferito nella funzione `dayend()`, è 17/24.

Esempio 4 - Esempio grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Il set di dati e lo scenario utilizzati in questo esempio corrispondono a quelli dei due esempi precedenti. La propria azienda desidera operare in base a una policy in cui il giorno lavorativo inizia alle 17:00 e termina alle 17:00 del giorno successivo.

Come nell'esempio precedente, si creeranno due nuovi campi:

- `deposit_due_date`: La data in cui si dovrebbe ricevere il deposito. Questa è la fine del giorno tre giorni prima del `service_date`.
- `final_payment_due_date`: La data in cui si dovrebbe ricevere il pagamento finale. Questa è la fine del giorno sette giorni dopo il `service_date`.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Services:
Load
service_id,
service_date,
amount
Inline
[
service_id, service_date,amount
1,03/11/2022 9:25:14 AM,231.24
2,03/12/2022 10:06:54 AM,567.28
3,03/13/2022 10:44:42 AM,364.28
4,03/14/2022 11:33:30 AM,575.76
5,03/15/2022 12:58:14 PM,638.68
6,03/16/2022 4:23:12 PM,785.38
```

```
7,03/17/2022 6:42:15 PM,967.46
8,03/18/2022 7:41:16 PM,287.67
9,03/19/2022 8:14:15 PM,764.45
10,03/20/2022 9:23:51 PM,875.43
11,03/21/2022 10:04:41 PM,957.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

`service_date`.

Per creare il campo `deposit_due_date`, creare questa misura:

`=dayend(service_date,-3,17/24)`.

Quindi, per creare il campo `final_payment_due_date`, creare questa misura:

`=dayend(service_date,7,17/24)`.

Tabella dei risultati

service_date	=dayend(service_date,-3,17/24)	=dayend(service_date,7,17/24
03/11/2022	3/8/2022 16:59:59 PM	3/18/2022 16:59:59 PM
03/12/2022	3/9/2022 16:59:59 PM	3/19/2022 16:59:59 PM
03/13/2022	3/10/2022 16:59:59 PM	3/20/2022 16:59:59 PM
03/14/2022	3/11/2022 16:59:59 PM	3/21/2022 16:59:59 PM
03/15/2022	3/12/2022 16:59:59 PM	3/22/2022 16:59:59 PM
03/16/2022	3/13/2022 16:59:59 PM	3/23/2022 16:59:59 PM
03/17/2022	3/14/2022 16:59:59 PM	3/24/2022 16:59:59 PM
03/18/2022	3/15/2022 16:59:59 PM	3/25/2022 16:59:59 PM
03/19/2022	3/16/2022 16:59:59 PM	3/26/2022 16:59:59 PM
03/20/2022	3/17/2022 16:59:59 PM	3/27/2022 16:59:59 PM
03/21/2022	3/18/2022 16:59:59 PM	3/28/2022 16:59:59 PM

I valori dei nuovi campi sono in `TimestampFormat M/D/YYYY h:mm:ss[.fff] TT`. Essendo stata usata la funzione `dayend()`, i valori data e ora sono tutti all'ultimo millisecondo del giorno.

I valori della data di pagamento del deposito sono tre giorni prima della data del servizio perché il secondo argomento passato nella funzione `dayend()` è negativo.

I valori della data di scadenza del pagamento finale sono sette giorni dopo la data del servizio perché il secondo argomento passato nella funzione `dayend()` è positivo.

Le date hanno un timestamp dell'ultimo millisecondo prima delle 17:00 a causa del fatto che il valore del terzo argomento, `day_start`, trasferito nella funzione `dayend()`, è 17/24.

daylightsaving

Restituisce il valore di regolazione attuale per l'ora legale, come definito in Windows.

Sintassi:

```
DaylightSaving( )
```

Tipo di dati restituiti: duale

Esempio:

```
daylightsaving( )
```

dayname

Questa funzione restituisce un valore che mostra la data con un valore numerico sottostante corrispondente a un indicatore temporale recante il primo millisecondo del giorno contenente **time**.

Sintassi:

```
DayName(time[, period_no [, day_start]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
time	L'indicatore temporale da valutare.
period_no	**period_no** è un numero intero o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica il giorno che contiene il valore **time**. I valori negativi di **period_no** indicano i giorni precedenti, mentre i valori positivi indicano i giorni successivi.
day_start	Per specificare che i giorni non iniziano a mezzanotte, indicare un differimento come frazione di un giorno in **day_start**. Ad esempio, 0.125 per denotare 3:00 AM.

Esempi e risultati:
In questi esempi viene utilizzato il formato della data **DD/MM/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di script

Esempio	Risultato
`dayname('25/01/2013 16:45:00')`	Restituisce 25/01/2013.
`dayname('25/01/2013 16:45:00', -1)`	Restituisce 24/01/2013.
`dayname('25/01/2013 16:45:00', 0, 0.5 )`	Restituisce 25/01/2013. La visualizzazione dell'indicatore temporale completo mostra che il valore numerico sottostante corrisponde a '25/01/2013 12:00:00.000.

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

In questo esempio il nome del giorno viene creato a partire dall'indicatore temporale che contrassegna l'inizio del giorno dopo ciascuna data della fattura nella tabella.

```
TempTable:
LOAD RecNo() as InvID, * Inline [
InvDate
28/03/2012
10/12/2012
5/2/2013
31/3/2013
19/5/2013
15/9/2013
11/12/2013
2/3/2014
14/5/2014
13/6/2014
7/7/2014
4/8/2014
];

InvoiceData:
LOAD *,
DayName(InvDate, 1) AS DName
Resident TempTable;
Drop table TempTable;
```

La tabella risultante contiene le date originali e una colonna con il valore restituito della funzione dayname (). È possibile visualizzare l'indicatore temporale completo specificando la formattazione nel pannello delle proprietà.

Tabella dei risultati

InvDate	DName
28/03/2012	29/03/2012 00:00:00

InvDate	DName
10/12/2012	11/12/2012 00:00:00
5/2/2013	07/02/2013 00:00:00
31/3/2013	01/04/2013 00:00:00
19/5/2013	20/05/2013 00:00:00
15/9/2013	16/09/2013 00:00:00
11/12/2013	12/12/2013 00:00:00
2/3/2014	03/03/2014 00:00:00
14/5/2014	15/05/2014 00:00:00
13/6/2014	14/06/2014 00:00:00
7/7/2014	08/07/2014 00:00:00
4/8/2014	05/08/2014 00:00:00

daynumberofquarter

Questa funzione calcola il numero del giorno del trimestre a cui è stato assegnato un indicatore temporale. Questa funzione viene utilizzata al momento di creare un Calendario principale.

Sintassi:

```
DayNumberOfQuarter(timestamp[,start_month])
```

Tipo di dati restituiti: numero intero

Argomenti

Argomento	Descrizione
timestamp	La data o la data e ora da valutare.
start_month	Se si specifica un valore **start_month** compreso tra 2 e 12 (1 se omesso), l'inizio dell'anno potrà essere spostato in avanti sul primo giorno di qualsiasi mese. Se, ad esempio, si intende utilizzare un anno fiscale che inizi il 1° marzo, specificare **start_month** = 3.

In questi esempi viene utilizzato il formato della data **DD/MM/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di funzioni

Esempio	Risultato
`DayNumberOfQuarter('12/09/2014')`	Restituisce 74, il numero del giorno del trimestre attuale.

Esempio	Risultato
`DayNumberOfQuarter ('12/09/2014',3)`	Restituisce 12, il numero del giorno del trimestre attuale. In questo caso, il primo trimestre inizia a marzo (in quanto start_month è specificato come 3). Ciò significa che il trimestre attuale è il terzo trimestre, iniziato il 1 settembre.

Esempio 1 – Gennaio inizio dell'anno (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati semplice contenente un elenco di date, caricato in una tabella denominata `Calendar`. È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- Un caricamento precedente che crea un campo aggiuntivo, denominato `DayNrQtr`, mediante la funzione `DayNumberOfQuarter()`.

Oltre alla data, non viene fornito alla funzione alcun parametro aggiuntivo.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Calendar:
Load
        date,
        DayNumberOfQuarter(date) as DayNrQtr
        ;
Load
date
Inline
[
date
01/01/2022
01/10/2022
01/31/2022
02/01/2022
02/10/2022
02/28/2022
03/01/2022
03/31/2022
04/01/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `daynrqtr`

Tabella dei risultati

data	daynrqtr
01/01/2022	1
01/10/2022	10
01/31/2022	31
02/01/2022	32
02/10/2022	41
02/28/2022	59
03/01/2022	61
03/31/2022	91
04/01/2022	1

Il primo giorno dell'anno era il 1° di gennaio dato che nessun secondo argomento era stato trasferito nella funzione `DayNumberOfQuarter()`.

Il 1° gennaio è il primo giorno del trimestre, mentre il 1° febbraio è il 32esimo giorno del trimestre. Il 31 marzo è il 91esimo e ultimo giorno del trimestre, mentre il 1° aprile è il primo giorno del 2° trimestre.

Esempio 2 – Febbraio inizio dell'anno (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- Un argomento `start_month` che inizia il 1° febbraio. Ciò imposta l'anno finanziario al 1° febbraio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Calendar:
```

```
Load
       date,
       DayNumberOfQuarter(date,2) as DayNrQtr
       ;
Load
date
Inline
[
date
01/01/2022
01/10/2022
01/31/2022
02/01/2022
02/10/2022
02/28/2022
03/01/2022
03/31/2022
04/01/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `daynrqtr`

Tabella dei risultati

data	daynrqtr
01/01/2022	62
01/10/2022	71
01/31/2022	92
02/01/2022	1
02/10/2022	10
02/28/2022	28
03/01/2022	30
03/31/2022	60
04/01/2022	61

Il primo giorno dell'anno è il 1° di febbraio, dato che il secondo argomento trasferito nella funzione `DayNumberOfQuarter()` era 2.

Il primo trimestre dell'anno va da febbraio ad aprile, mentre il quarto trimestre va da novembre a gennaio. Questo viene mostrato nella tabella dei risultati, dove il 1° febbraio è il primo giorno del trimestre mentre il 31 gennaio è il 92esimo e ultimo giorno del trimestre.

Esempio 3 – Gennaio inizio dell'anno (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il valore del giorno del trimestre viene calcolato mediante una misura in un oggetto del grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Calendar:
Load
date
Inline
[
date
01/01/2022
01/10/2022
01/31/2022
02/01/2022
02/10/2022
02/28/2022
03/01/2022
03/31/2022
04/01/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Creare la seguente misura:

```
=daynumberofquarter(date)
```

Tabella dei risultati

data	=daynumberofquarter(date)
01/01/2022	1
01/10/2022	10

data	=daynumberofquarter(date)
01/31/2022	31
02/01/2022	32
02/10/2022	41
02/28/2022	59
03/01/2022	61
03/31/2022	91
04/01/2022	1

Il primo giorno dell'anno era il 1° di gennaio dato che nessun secondo argomento era stato trasferito nella funzione `DayNumberOfQuarter()`.

Il 1° gennaio è il primo giorno del trimestre, mentre il 1° febbraio è il 32esimo giorno del trimestre. Il 31 marzo è il 91esimo e ultimo giorno del trimestre, mentre il 1° aprile è il primo giorno del 2° trimestre.

Esempio 4 – Febbraio inizio dell'anno (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- L'anno finanziario va dal 1° febbraio al 31 gennaio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il valore del giorno del trimestre viene calcolato mediante una misura in un oggetto del grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Calendar:
Load
date
Inline
[
date
01/01/2022
01/10/2022
01/31/2022
02/01/2022
```

```
02/10/2022
02/28/2022
03/01/2022
03/31/2022
04/01/2022
];
```

Oggetto grafico

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Creare la seguente misura:

```
=daynumberofquarter(date,2)
```

Risultati

Tabella dei risultati

data	=daynumberofquarter(date,2)
01/01/2022	62
01/10/2022	71
01/31/2022	92
02/01/2022	1
02/10/2022	10
02/28/2022	28
03/01/2022	30
03/31/2022	60
04/01/2022	61

Il primo giorno dell'anno è il 1° di gennaio, dato che il secondo argomento trasferito nella funzione `DayNumberOfQuarter()` era 2.

Il primo trimestre dell'anno va da febbraio ad aprile, mentre il quarto trimestre va da novembre a gennaio. Questo viene evidenziato nella tabella dei risultati, dove il 1° febbraio è il primo giorno del trimestre mentre il 31 gennaio è il 92esimo e ultimo giorno del trimestre.

daynumberofyear

Questa funzione calcola il numero del giorno dell'anno a cui è stato assegnato un indicatore temporale. Il calcolo viene eseguito partendo dal primo millisecondo del primo giorno dell'anno, tuttavia il primo mese può essere differito.

Sintassi:

```
DayNumberOfYear(timestamp[,start_month])
```

Tipo di dati restituiti: numero intero

Argomenti

Argomento	Descrizione
timestamp	La data o la data e ora da valutare.
start_month	Se si specifica un valore **start_month** compreso tra 2 e 12 (1 se omesso), l'inizio dell'anno potrà essere spostato in avanti sul primo giorno di qualsiasi mese. Se, ad esempio, si intende utilizzare un anno fiscale che inizi il 1° marzo, specificare **start_month** = 3.

In questi esempi viene utilizzato il formato della data **DD/MM/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di funzioni

Esempio	Risultato
`DayNumberOfYear( '12/09/2014' )`	Restituisce 256, il numero del giorno conteggiato dal primo giorno dell'anno.
`DayNumberOfYear( '12/09/2014',3 )`	Restituisce 196, il numero del giorno conteggiato a partire dal 1° marzo.

Esempio 1 – Gennaio inizio dell'anno (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati semplice contenente un elenco di date, caricato in una tabella denominata `Calendar`. È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- Un caricamento precedente che crea un campo aggiuntivo, denominato `daynryear`, mediante la funzione `DayNumberOfYear()`.

Oltre alla data, non viene fornito alla funzione alcun parametro aggiuntivo.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Calendar:
Load
       date,
```

```
       DayNumberOfYear(date) as daynryear
       ;
Load
date
Inline
[
date
01/01/2022
01/10/2022
01/31/2022
02/01/2022
02/10/2022
06/30/2022
07/26/2022
10/31/2022
11/01/2022
12/31/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `daynryear`

Tabella dei risultati

data	daynryear
01/01/2022	1
01/10/2022	10
01/31/2022	31
02/01/2022	32
02/10/2022	41
06/30/2022	182
07/26/2022	208
10/31/2022	305
11/01/2022	306
12/31/2022	366

Il primo giorno dell'anno era il 1° di gennaio dato che nessun secondo argomento era stato trasferito nella funzione `DayNumberOfYear()`.

Il 1° gennaio è il primo giorno del trimestre, mentre il 1° febbraio è il 32esimo giorno dell'anno. Il 30 giugno è il 182esimo giorno dell'anno mentre il 31 dicembre è il 366esimo e ultimo giorno dell'anno.

Esempio 2 – Novembre inizio dell'anno (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA
- Un argomento `start_month` che inizia il 1° novembre. Ciò imposta l'anno finanziario al 1° novembre.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Calendar:
Load
       date,
       DayNumberOfYear(date,11) as daynryear
       ;
Load
date
Inline
[
date
01/01/2022
01/10/2022
01/31/2022
02/01/2022
02/10/2022
06/30/2022
07/26/2022
10/31/2022
11/01/2022
12/31/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `daynryear`

Tabella dei risultati

data	daynryear
01/01/2022	62
01/10/2022	71
01/31/2022	92
02/01/2022	93
02/10/2022	102
06/30/2022	243
07/26/2022	269
10/31/2022	366
11/01/2022	1
12/31/2022	61

Il primo giorno dell'anno è il 1° di novembre, dato che il secondo argomento trasferito nella funzione `DayNumberOfYear()` era 11.

Il 1° gennaio è il primo giorno del trimestre, mentre il 1° febbraio è il 32esimo giorno dell'anno. Il 30 giugno è il 182esimo giorno dell'anno mentre il 31 dicembre è il 366esimo e ultimo giorno dell'anno.

Esempio 3 – Gennaio inizio dell'anno (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il valore del giorno del trimestre viene calcolato mediante una misura in un oggetto del grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Calendar:
Load
date
Inline
[
date
```

```
01/01/2022
01/10/2022
01/31/2022
02/01/2022
02/10/2022
06/30/2022
07/26/2022
10/31/2022
11/01/2022
12/31/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Creare la seguente misura:

```
=daynumberofyear(date)
```

Tabella dei risultati

data	=daynumberofyear(date)
01/01/2022	1
01/10/2022	10
01/31/2022	31
02/01/2022	32
02/10/2022	41
06/30/2022	182
07/26/2022	208
10/31/2022	305
11/01/2022	306
12/31/2022	366

Il primo giorno dell'anno era il 1° di gennaio dato che nessun secondo argomento era stato trasferito nella funzione `DayNumberOfYear()`.

Il 1° gennaio è il primo giorno dell'anno, mentre il 1° febbraio è il 32esimo giorno dell'anno. Il 30 giugno è il 182esimo giorno dell'anno mentre il 31 dicembre è il 366esimo e ultimo giorno dell'anno.

Esempio 4 – Novembre inizio dell'anno (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- L'anno finanziario va dal 1° novembre al 31 ottobre.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il valore del giorno dell'anno viene calcolato mediante una misura in un oggetto del grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
Calendar:
Load
date
Inline
[
date
01/01/2022
01/10/2022
01/31/2022
02/01/2022
02/10/2022
06/30/2022
07/26/2022
10/31/2022
11/01/2022
12/31/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Creare la seguente misura:

```
=daynumberofyear(date)
```

Tabella dei risultati

data	=daynumberofyear(date,11)
01/01/2022	62

data	=daynumberofyear(date,11)
01/10/2022	71
01/31/2022	92
02/01/2022	93
02/10/2022	102
06/30/2022	243
07/26/2022	269
10/31/2022	366
11/01/2022	1
12/31/2022	61

Il primo giorno dell'anno è il 1° di novembre, dato che il secondo argomento trasferito nella funzione `DayNumberOfYear()` era 11.

L'anno finanziario va da novembre a ottobre. Questo viene mostrato nella tabella dei risultati, dove il 1° novembre è il primo giorno dell'anno mentre il 31 ottobre è il 366esimo e ultimo giorno dell'anno.

daystart

Questa funzione restituisce un valore corrispondente a un indicatore temporale con il primo millisecondo del giorno contenuto nell'argomento **time**. Il formato di output predefinito sarà il formato **TimestampFormat** impostato nello script.

Sintassi:

```
DayStart(time[, [period_no[, day_start]])
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
time	Il timestamp da valutare.
period_no	**period_no** è un numero intero o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica il giorno che contiene il valore **time**. I valori negativi di **period_no** indicano i giorni precedenti, mentre i valori positivi indicano i giorni successivi.
day_start	Per specificare che i giorni non iniziano a mezzanotte, indicare un differimento come frazione di un giorno in **day_start**. Ad esempio, 0.125 per denotare 3:00 AM. In altre parole, per creare l'offset dividere l'ora di inizio per 24 ore. Ad esempio, per un giorno che inizia alle 7:00 del mattino, utilizzare la frazione 7/24.

Casi di utilizzo

La funzione `daystart()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione del giorno trascorsa finora. Ad esempio, può essere utilizzata per calcolare il totale dei salari guadagnati dai dipendenti nella giornata fino a quel momento.

Questi esempi utilizzano il formato timestamp `'M/D/YYYY h:mm:ss[.fff] TT'`. Il formato timestamp viene specificato nell'istruzione `SET TimeStamp` in cima allo script di caricamento dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di funzioni

Esempio	Risultato
`daystart('01/25/2013 4:45:00 PM')`	Restituisce 1/25/2013 12:00:00 AM.
`daystart('1/25/2013 4:45:00 PM', -1)`	Restituisce 1/24/2013 12:00:00 AM.
`daystart('1/25/2013 16:45:00',0,0.5 )`	Restituisce 1/25/2013 12:00:00 PM.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Esempio semplice

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati semplice contenente un elenco di date, caricato in una tabella denominata `Calendar`.
- Viene utilizzata la variabile di sistema `TimeStampFormat` predefinita (`(M/D/YYYY h:mm:ss[.fff] TT`).
- Un caricamento precedente che crea un campo aggiuntivo, chiamato `SOD_timestamp`, utilizzando la funzione `daystart()`.

Oltre alla data, non viene fornito alla funzione alcun parametro aggiuntivo.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Calendar:
    Load
        date,
        daystart(date) as SOD_timestamp
        ;
Load
date
Inline
[
date
03/11/2022 1:47:15 AM
03/12/2022 4:34:58 AM
03/13/2022 5:15:55 AM
03/14/2022 9:25:14 AM
03/15/2022 10:06:54 AM
03/16/2022 10:44:42 AM
03/17/2022 11:33:30 AM
03/18/2022 12:58:14 PM
03/19/2022 4:23:12 PM
03/20/2022 6:42:15 PM
03/21/2022 7:41:16 PM
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `SOD_timestamp`

Tabella dei risultati

date	SOD_timestamp
03/11/2022 1:47:15 AM	3/11/2022 12:00:00 AM
03/12/2022 4:34:58 AM	3/12/2022 12:00:00 AM
03/13/2022 5:15:55 AM	3/13/2022 12:00:00 AM
03/14/2022 9:25:14 AM	3/14/2022 12:00:00 AM
03/15/2022 10:06:54 AM	3/15/2022 12:00:00 AM
03/16/2022 10:44:42 AM	3/16/2022 12:00:00 AM
03/17/2022 11:33:30 AM	3/17/2022 12:00:00 AM
03/18/2022 12:58:14 PM	3/18/2022 12:00:00 AM

date	SOD_timestamp
03/19/2022 4:23:12 PM	3/19/2022 12:00:00 AM
03/20/2022 6:42:15 PM	3/20/2022 12:00:00 AM
03/21/2022 7:41:16 PM	3/21/2022 12:00:00 AM

Come appare nella tabella sopra, la fine del timestamp giorno viene generata per ciascuna data nel nostro set di dati. Il timestamp è nel formato della variabile di sistema `TimestampFormat M/D/YYYY h:mm:ss[.fff] TT`.

Esempio 2 - period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente le multe per divieto di sosta, che viene caricato in una tabella denominata `Fines`. Il set di dati include i seguenti campi:
  - `id`
  - `due_date`
  - `number_plate`
  - `amount`
- Un caricamento precedente che utilizza la funzione `daystart()` e fornisce tutti e tre i parametri: `time`, `period_no` e `day_start`. Questo caricamento crea i seguenti due nuovi campi data:
  - Un campo di data `early_repayment_period`, che inizia sette giorni prima della scadenza del pagamento.
  - Un campo di data `late_penalty_period`, che inizia 14 giorni dopo la scadenza del pagamento.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Fines:
    Load
        *,
        daystart(due_date,-7) as early_repayment_period,
        daystart(due_date,14) as late_penalty_period
        ;
Load
*
Inline
[
id, due_date, number_plate,amount
```

```
1,02/11/2022, 573RJG,50.00
2,03/25/2022, SC41854,50.00
3,04/14/2022, 8EHZ378,50.00
4,06/28/2022, 8HSS198,50.00
5,08/15/2022, 1221665,50.00
6,11/16/2022, EAK473,50.00
7,01/17/2023, KD6822,50.00
8,03/22/2023, 1GGLB,50.00
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- due_date
- early_repayment_period
- late_penalty_period

Tabella dei risultati

due_date	early_repayment_period	late_penalty_period
02/11/2022 9:25:14 AM	2/4/2022 12:00:00 AM	2/25/2022 12:00:00 AM
03/25/2022 10:06:54 AM	3/18/2022 12:00:00 AM	4/8/2022 12:00:00 AM
04/14/2022 10:44:42 AM	4/7/2022 12:00:00 AM	4/28/2022 12:00:00 AM
06/28/2022 11:33:30 AM	6/21/2022 12:00:00 AM	7/12/2022 12:00:00 AM
08/15/2022 12:58:14 PM	8/8/2022 12:00:00 AM	8/29/2022 12:00:00 AM
11/16/2022 4:23:12 PM	11/9/2022 12:00:00 AM	11/30/2022 12:00:00 AM
01/17/2023 6:42:15 PM	1/10/2023 12:00:00 AM	1/31/2023 12:00:00 AM
03/22/2023 7:41:16 PM	3/15/2023 12:00:00 AM	4/5/2023 12:00:00 AM

I valori dei nuovi campi sono in `TimestampFormat M/DD/YYYY tt`. Essendo stata usata la funzione `daystart ()`, i valori timestamp sono tutti al primo millisecondo del giorno.

I valori del periodo di rimborso anticipato sono sette giorni prima della data di scadenza, in quanto il secondo argomento passato nella funzione `daystart()` è negativo.

I valori del periodo di rimborso tardivo sono 14 giorni dopo la data di scadenza, in quanto il secondo argomento passato nella funzione `daystart()` è positivo.

Esempio 3 – day_start

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del precedente esempio.
- Lo stesso caricamento precedente dell'esempio precedente.

In questo esempio, abbiamo impostato la giornata lavorativa in modo che inizi e termini alle 7:00 del mattino di ogni giorno.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Fines:
    Load
        *,
        daystart(due_date,-7,7/24) as early_repayment_period,
        daystart(due_date,14, 7/24) as late_penalty_period
        ;
Load
*
Inline
[
id, due_date, number_plate,amount
1,02/11/2022, 573RJG,50.00
2,03/25/2022, SC41854,50.00
3,04/14/2022, 8EHZ378,50.00
4,06/28/2022, 8HSS198,50.00
5,08/15/2022, 1221665,50.00
6,11/16/2022, EAK473,50.00
7,01/17/2023, KD6822,50.00
8,03/22/2023, 1GGLB,50.00
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- due_date
- early_repayment_period
- late_penalty_period

Tabella dei risultati

due_date	early_repayment_period	late_penalty_period
02/11/2022	2/3/2022 7:00:00 AM	2/24/2022 7:00:00 AM
03/25/2022	3/17/2022 7:00:00 AM	4/7/2022 7:00:00 AM
04/14/2022	4/6/2022 7:00:00 AM	4/27/2022 7:00:00 AM
06/28/2022	6/20/2022 7:00:00 AM	7/11/2022 7:00:00 AM

due_date	early_repayment_period	late_penalty_period
08/15/2022	8/7/2022 7:00:00 AM	8/28/2022 7:00:00 AM
11/16/2022	11/8/2022 7:00:00 AM	11/29/2022 7:00:00 AM
01/17/2023	1/9/2023 7:00:00 AM	1/30/2023 7:00:00 AM
03/22/2023	3/14/2023 7:00:00 AM	4/4/2023 7:00:00 AM

Le date ora hanno un timestamp di 7:00 AM perché il valore dell'argomento `day_start` passato alla funzione `daystart()` è 7/24. In questo modo si imposta l'inizio del giorno alle 7:00 AM.

Poiché il campo `due_date` non ha un timestamp, viene trattato come le 12:00 AM, che quindi fa ancora parte del giorno precedente, dato che i giorni iniziano e finiscono alle 7:00 del mattino. Pertanto, il periodo di rimborso anticipato per una multa in scadenza l'11 febbraio inizia il 3 febbraio alle ore 7:00.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Questo esempio utilizza lo stesso set di dati e lo stesso scenario dell'esempio precedente.

Tuttavia, solo la tabella `Fines` originale viene caricata nell'applicazione, mentre i due valori aggiuntivi delle scadenze vengono calcolati in un oggetto grafico.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Fines:
    Load
*
Inline
[
id, due_date, numer_plate,amount
1,02/11/2022 9:25:14 AM, 573RJG,50.00
2,03/25/2022 10:06:54 AM, SC41854,50.00
3,04/14/2022 10:44:42 AM, 8EHZ378,50.00
4,06/28/2022 11:33:30 AM, 8HSS198,50.00
5,08/15/2022 12:58:14 PM, 1221665,50.00
6,11/16/2022 4:23:12 PM, EAK473,50.00
7,01/17/2023 6:42:15 PM, KD6822,50.00
8,03/22/2023 7:41:16 PM, 1GGLB,50.00
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `due_date`.
2. Per creare il campo `early_repayment_period`, creare la seguente misura: `=daystart(due_date,-7,7/24)`
3. Per creare il campo `late_penalty_period`, creare la seguente misura: =daystart(due_date,14,7/24)

Tabella dei risultati

due_date	=daystart(due_date,-7,7/24)	=daystart(due_date,14,7/24)
02/11/2022 9:25:14 AM	2/4/2022 7:00:00 AM	2/25/2022 7:00:00 AM
03/25/2022 10:06:54 AM	3/18/2022 7:00:00 AM	4/8/2022 7:00:00 AM
04/14/2022 10:44:42 AM	4/7/2022 7:00:00 AM	4/28/2022 7:00:00 AM
06/28/2022 11:33:30 AM	6/21/2022 7:00:00 AM	7/12/2022 7:00:00 AM
08/15/2022 12:58:14 PM	8/8/2022 7:00:00 AM	8/29/2022 7:00:00 AM
11/16/2022 4:23:12 PM	11/9/2022 7:00:00 AM	11/30/2022 7:00:00 AM
01/17/2023 6:42:15 PM	1/10/2023 7:00:00 AM	1/31/2023 7:00:00 AM
03/22/2023 7:41:16 PM	3/15/2023 7:00:00 AM	4/5/2023 7:00:00 AM

I valori dei nuovi campi sono in `TimestampFormat M/D/YYYY h:mm:ss[.fff] TT`. Essendo stata usata la funzione `daystart()`, i valori timestamp corrispondono tutti al primo millisecondo del giorno.

I valori del periodo di rimborso anticipato sono sette giorni prima della data di scadenza, poiché il secondo argomento passato nella funzione `daystart()` era negativo.

I valori del periodo di rimborso tardivo sono 14 giorni dopo la data di scadenza, poiché il secondo argomento passato nella funzione `daystart()` era positivo.

Le date hanno un timestamp corrispondente alle 7:00 AM perché il valore del terzo argomento passato alla funzione `daystart()`, `day_start`, era 7/24.

firstworkdate

La funzione **firstworkdate** restituisce la data di inizio più recente per fare in modo che il valore **no_of_workdays** (dal lunedì al venerdì) non termini oltre la data **end_date**, tenendo in considerazione tutte le festività eventualmente in calendario. **end_date** e **holiday** devono essere date o indicatori temporali validi.

Sintassi:

```
firstworkdate(end_date, no_of_workdays {, holiday} )
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
end_date	L'indicatore temporale della data di fine da valutare.
no_of_ workdays	Il numero di giorni lavorativi da raggiungere.
holiday	Periodi di vacanza da escludere dai giorni lavorativi. Una vacanza è definita come una stringa di data costante. È possibile specificare più periodi di vacanza, separati da virgole. **Esempio:** `'12/25/2013', '12/26/2013', '12/31/2013', '01/01/2014'`

Esempi e risultati:
In questi esempi viene utilizzato il formato della data **DD/MM/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di script

Esempio	Risultato
`firstworkdate ('29/12/2014', 9)`	Restituisce '17/12/2014.
`firstworkdate ('29/12/2014', 9, '25/12/2014', '26/12/2014')`	Restituisce 15/12/2014 perché viene considerato un periodo di vacanza di due giorni.

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
ProjectTable:
LOAD *, recno() as InvID, INLINE [
EndDate
28/03/2015
10/12/2015
5/2/2016
31/3/2016
19/5/2016
15/9/2016
] ;
NrDays:
Load *,
FirstWorkDate(EndDate,120) As StartDate
Resident ProjectTable;
Drop table ProjectTable;
```

La tabella risultante mostra i valori restituiti in FirstWorkDate per ciascun record della tabella.

Tabella dei risultati

InvID	EndDate	StartDate
1	28/03/2015	13/10/2014
2	10/12/2015	26/06/2015
3	5/2/2016	24/08/2015
4	31/3/2016	16/10/2015
5	19/5/2016	04/12/2015
6	15/9/2016	01/04/2016

GMT

Questa funzione restituisce il valoreGreenwich Mean Timeattuale ricavato dalle impostazioni locali. La funzione restituisce i valori nel formato della variabile di sistema `TimestampFormat`.

Ogni volta che l'app viene ricaricata, qualsiasi tabella, variabile o oggetto grafico dello script di caricamento che utilizza la funzione `GMT` verrà adattato all'ultimo orario medio di Greenwich derivato dall'orologio di sistema.

Sintassi:

```
GMT( )
```

Tipo di dati restituiti: duale

Questi esempi utilizzano il formato timestamp `M/D/YYYY h:mm:ss[.fff] TT`. Il formato della data viene specificato nell'istruzione `SET TimestampFormat` nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di funzioni

Esempio	Risultato
`GMT()`	3/28/2022 2:47:36 PM

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Variabile (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda. Questo esempio imposta l'attuale ora di Greenwich come variabile nello script di caricamento utilizzando la funzione `GMT`.

Script di caricamento

```
LET vGMT = GMT();
```

Risultati

Caricare i dati e creare un foglio. Creare una casella di testo utilizzando l'oggetto grafico **Testo e immagine**.

Aggiungere questa misura alla casella di testo:

```
=vGMT
```

La casella di testo deve contenere una riga di testo con data e ora, come quella mostrata di seguito:

```
3/28/2022 2:47:36 PM
```

Esempio 2 - Inizio dell'anno a novembre (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente i libri scaduti della biblioteca, caricati in una tabella denominata `Overdue`. È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- La creazione di un nuovo campo chiamato `days_overdue`, che calcola il numero di giorni di ritardo di ogni libro.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Overdue:
    Load
        *,
        Floor(GMT()-due_date) as days_overdue
        ;
Load
*
Inline
[
cust_id,book_id,due_date
1,4,01/01/2021,
2,24,01/10/2021,
6,173,01/31/2021,
31,281,02/01/2021,
86,265,02/10/2021,
52,465,06/30/2021,
26,537,07/26/2021,
92,275,10/31/2021,
27,455,11/01/2021,
27,46,12/31/2021
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- due_date
- book_id
- days_overdue

Tabella dei risultati

due_date	book_id	days_overdue
01/01/2021	4	455
01/10/2021	24	446
01/31/2021	173	425
02/01/2021	281	424
02/10/2021	265	415
06/30/2021	465	275
07/26/2021	537	249
10/31/2021	275	152
11/01/2021	455	151
12/31/2021	46	91

I valori del campo `days_overdue` sono calcolati trovando la differenza tra l'ora di Greenwich attuale, utilizzando la funzione `GMT()`, e la data di scadenza originale. Per calcolare solo i giorni, i risultati vengono arrotondati al numero intero più vicino utilizzando la funzione `Floor()`.

Esempio 3 - oggetto grafico (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda. Lo script di caricamento contiene lo stesso set di dati dell'esempio precedente. È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il valore del numero di giorni di ritardo viene calcolato tramite una misura in un oggetto grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Overdue:
Load
*
Inline
[
cust_id,book_id,due_date
1,4,01/01/2021,
2,24,01/10/2021,
6,173,01/31/2021,
31,281,02/01/2021,
86,265,02/10/2021,
52,465,06/30/2021,
26,537,07/26/2021,
92,275,10/31/2021,
27,455,11/01/2021,
27,46,12/31/2021
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `due_date`
- `book_id`

Creare la seguente misura:

```
=Floor(GMT() - due_date)
```

Tabella dei risultati

due_date	book_id	=Floor(GMT()-due_date)
01/01/2021	4	455
01/10/2021	24	446
01/31/2021	173	425
02/01/2021	281	424
02/10/2021	265	415
06/30/2021	465	275
07/26/2021	537	249
10/31/2021	275	152
11/01/2021	455	151
12/31/2021	46	91

I valori del campo `days_overdue` sono calcolati trovando la differenza tra l'ora di Greenwich attuale, utilizzando la funzione `GMT()`, e la data di scadenza originale. Per calcolare solo i giorni, i risultati vengono arrotondati al numero intero più vicino utilizzando la funzione `Floor()`.

hour

Questa funzione restituisce un numero intero che rappresenta l'ora in cui la frazione di **expression** viene interpretata come ora in base all'interpretazione numerica standard.

Sintassi:

```
hour(expression)
```

Tipo di dati restituiti: numero intero

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`hour( '09:14:36' )`	La stringa di testo fornita è convertita in modo implicito in un data e ora in quanto corrisponde al formato data e ora definito nella variabile TimestampFormat. L'espressione restituisce 9.
`hour( '0.5555' )`	L'espressione restituisce 13 (poiché 0.5555 = 13:19:55).

Esempio 1 – Variabile (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente transazioni per data e ora
- La variabile di sistema `TimeStamp` predefinita `(M/D/YYYY h:mm:ss[.fff] TT)`

Creare un campo, '`hour`', calcolando quando sono avvenuti gli acquisti.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
  Load
    *,
    hour(date) as hour
    ;
Load
*
Inline
[
id,date,amount
9497,'2022-01-05 19:04:57',47.25,
9498,'2022-01-03 14:21:53',51.75,
9499,'2022-01-03 05:40:49',73.53,
9500,'2022-01-04 18:49:38',15.35,
9501,'2022-01-01 22:10:22',31.43,
9502,'2022-01-05 19:34:46',13.24,
9503,'2022-01-04 22:58:34',74.34,
9504,'2022-01-06 11:29:38',50.00,
9505,'2022-01-02 08:35:54',36.34,
9506,'2022-01-06 08:49:09',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `hour`

Tabella dei risultati

data	ora
2022-01-01 22:10:22	22
2022-01-02 08:35:54	8
2022-01-03 05:40:49	5
2022-01-03 14:21:53	14
2022-01-04 18:49:38	18
2022-01-04 22:58:34	22
2022-01-05 19:04:57	19
2022-01-05 19:34:46	19
2022-01-06 08:49:09	8
2022-01-06 11:29:38	11

I valori nel campo ora sono creati usando la funzione `hour()` e trasferendo la data come espressione nell'istruzione di caricamento precedente.

Esempio 2 - Oggetto grafico (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- La variabile di sistema `TimeStamp` predefinita `(M/D/YYYY h:mm:ss[.fff] TT)`.

Tuttavia, in questo esempio, il set di dati, invariato, viene caricato nell'applicazione. I valori '`hour`' sono calcolati mediante una misura in un oggetto grafico.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
```

```
Load
*
Inline
[
id,date,amount
9497,'2022-01-05 19:04:57',47.25,
9498,'2022-01-03 14:21:53',51.75,
9499,'2022-01-03 05:40:49',73.53,
9500,'2022-01-04 18:49:38',15.35,
9501,'2022-01-01 22:10:22',31.43,
9502,'2022-01-05 19:34:46',13.24,
9503,'2022-01-04 22:58:34',74.34,
9504,'2022-01-06 11:29:38',50.00,
9505,'2022-01-02 08:35:54',36.34,
9506,'2022-01-06 08:49:09',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Per calcolare il '`hour`', creare la seguente misura:

```
=hour(date)
```

Tabella dei risultati

due_date	=hour(date)
2022-01-01 22:10:22	22
2022-01-02 08:35:54	8
2022-01-03 05:40:49	5
2022-01-03 14:21:53	14
2022-01-04 18:49:38	18
2022-01-04 22:58:34	22
2022-01-05 19:04:57	19
2022-01-05 19:34:46	19
2022-01-06 08:49:09	8
2022-01-06 11:29:38	11

I valori per '`hour`' sono creati usando la funzione `hour()` e trasferendo la data come espressione in una misura per l'oggetto grafico.

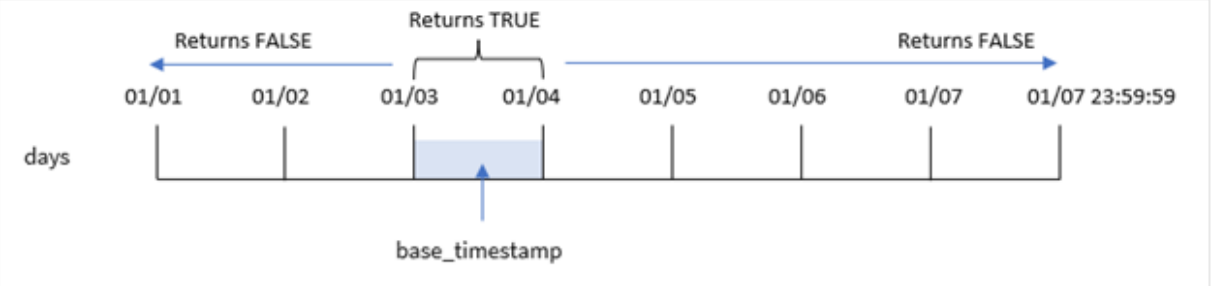
inday

Questa funzione restituisce True se **timestamp** ricade all'interno del giorno contenente **base_timestamp**.

Sintassi:

```
InDay (timestamp, base_timestamp, period_no[, day_start])
```

Schema della funzione `inday`.



La funzione `inday()` utilizza l'argomento `base_timestamp` per identificare il giorno in cui cade il timestamp. L'ora di inizio del giorno è, per impostazione predefinita, la mezzanotte; ma è possibile modificare l'ora di inizio del giorno utilizzando l'argomento `day_start` della funzione `inday()`. Una volta definito il giorno, la funzione restituisce risultati booleani quando si confrontano i valori dei timestamp prescritti con quel giorno.

Casi di utilizzo

La funzione `inday()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. Restituisce un'aggregazione o un calcolo che dipende dal fatto che una data valutata si sia verificata nel giorno del timestamp in questione.

Ad esempio, la funzione `inday()` può essere utilizzata per identificare tutte le apparecchiature prodotte in un determinato giorno.

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Argomenti

Argomento	Descrizione
`timestamp`	La data e l'ora che si desidera confrontare con `base_timestamp`.
`base_timestamp`	La data e l'ora utilizzate per valutare l'indicatore temporale.
`period_no`	Il giorno può essere differito mediante `period_no`. `period_no` è un numero intero, in cui il valore 0 indica il giorno che contiene `base_timestamp`. I valori negativi di `period_no` indicano i giorni precedenti, mentre i valori positivi indicano i giorni successivi.

Argomento	Descrizione
day_start	Se si intende utilizzare giorni che non inizino alla mezzanotte, indicare un differimento come frazione di un giorno in day_start, ad esempio 0,125 per indicare le 3 del mattino.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione SET DateFormat nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
inday ('01/12/2006 12:23:00 PM', '01/12/2006 12:00:00 AM', 0)	Restituisce True
inday ('01/12/2006 12:23:00 PM', '01/13/2006 12:00:00 AM', 0)	Restituisce False
inday ('01/12/2006 12:23:00 PM', '01/12/2006 12:00:00 AM', -1)	Restituisce False
inday ('01/11/2006 12:23:00 PM', '01/12/2006 12:00:00 AM', -1)	Restituisce True
inday ('01/12/2006 12:23:00 PM', '01/12/2006 12:00:00 AM', 0, 0.5)	Restituisce False
inday ('01/12/2006 11:23:00 AM', '01/12/2006 12:00:00 AM', 0, 0.5)	Restituisce True

Esempio 1 – Istruzione LOAD (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente transazioni per data e ora caricate in una tabella chiamata Transactions.
- Un campo data fornito nel formato TimeStamp della variabile di sistema (M/D/YYYY h:mm:ss[.fff] TT).
- Un carico precedente che contiene la funzione inday() impostata come campo in_day.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
    Load
        *,
        inday(date,'01/05/2022 12:00:00 AM', 0) as in_day
        ;
Load
*
Inline
[
id,date,amount
9497,'01/01/2022 7:34:46 PM',13.24
9498,'01/01/2022 10:10:22 PM',31.43
9499,'01/02/2022 8:35:54 AM',36.34
9500,'01/03/2022 2:21:53 PM',51.75
9501,'01/04/2022 6:49:38 PM',15.35
9502,'01/04/2022 10:58:34 PM',74.34
9503,'01/05/2022 5:40:49 AM',73.53
9504,'01/05/2022 11:29:38 AM',50.00
9505,'01/05/2022 7:04:57 PM',47.25
9506,'01/06/2022 8:49:09 AM',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_day`

Tabella dei risultati

data	in_day
01/01/2022 7:34:46 PM	0
01/01/2022 10:10:22 PM	0
01/02/2022 8:35:54 AM	0
01/03/2022 2:21:53 PM	0
01/04/2022 6:49:38 PM	0
01/04/2022 10:58:34 PM	0
01/05/2022 5:40:49 AM	-1
01/05/2022 11:29:38 AM	-1
01/05/2022 7:04:57 PM	-1
01/06/2022 8:49:09 AM	0

Il campo `in_day` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `inday()` e trasferendo il campo data, un timestamp con hard coding per il 5 gennaio e un `period_no` di `0` come argomenti della funzione.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario utilizzati nel primo esempio.

Tuttavia, in questo esempio, l'attività viene calcolata se la data di transazione è occorsa due giorni prima del 5 gennaio.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
    Load
        *,
        inday(date,'01/05/2022 12:00:00 AM', -2) as in_day
        ;
Load
*
Inline
[
id,date,amount
9497,'01/01/2022 7:34:46 PM',13.24
9498,'01/01/2022 10:10:22 PM',31.43
9499,'01/02/2022 8:35:54 AM',36.34
9500,'01/03/2022 2:21:53 PM',51.75
9501,'01/04/2022 6:49:38 PM',15.35
9502,'01/04/2022 10:58:34 PM',74.34
9503,'01/05/2022 5:40:49 AM',73.53
9504,'01/05/2022 11:29:38 AM',50.00
9505,'01/05/2022 7:04:57 PM',47.25
9506,'01/06/2022 8:49:09 AM',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_day`

Tabella dei risultati

data	in_day
01/01/2022 7:34:46 PM	0
01/01/2022 10:10:22 PM	0
01/02/2022 8:35:54 AM	0
01/03/2022 2:21:53 PM	-1
01/04/2022 6:49:38 PM	0
01/04/2022 10:58:34 PM	0
01/05/2022 5:40:49 AM	0
01/05/2022 11:29:38 AM	0
01/05/2022 7:04:57 PM	0
01/06/2022 8:49:09 AM	0

In questa istanza, dato che un `period_no` di `-2` è stato utilizzato come argomento offset nella funzione `inday()`, la funzione determina se ciascuna data di transazione è avvenuta il 3 gennaio. Ciò può essere verificato nella tabella di output in cui una transazione restituisce un risultato booleano di TRUE.

Esempio 3 – day_start

Script di caricamento e risultati

Panoramica

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario utilizzati negli esempi precedenti.

Tuttavia, in questo esempio, la policy aziendale indica che la giornata lavorativa inizia e termina alle 7:00.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
    Load
        *,
        inday(date,'01/05/2022 12:00:00 AM', 0, 7/24) as in_day
        ;
Load
*
Inline
[
id,date,amount
9497,'01/01/2022 7:34:46 PM',13.24
9498,'01/01/2022 10:10:22 PM',31.43
9499,'01/02/2022 8:35:54 AM',36.34
9500,'01/03/2022 2:21:53 PM',51.75
9501,'01/04/2022 6:49:38 PM',15.35
```

```
9502,'01/04/2022 10:58:34 PM',74.34
9503,'01/05/2022 5:40:49 AM',73.53
9504,'01/05/2022 11:29:38 AM',50.00
9505,'01/05/2022 7:04:57 PM',47.25
9506,'01/06/2022 8:49:09 AM',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_day`

Tabella dei risultati

data	in_day
01/01/2022 7:34:46 PM	0
01/01/2022 10:10:22 PM	0
01/02/2022 8:35:54 AM	0
01/03/2022 2:21:53 PM	0
01/04/2022 6:49:38 PM	-1
01/04/2022 10:58:34 PM	-1
01/05/2022 5:40:49 AM	-1
01/05/2022 11:29:38 AM	0
01/05/2022 7:04:57 PM	0
01/06/2022 8:49:09 AM	0

Dato che l'argomento `start_day` di 7/24, che corrisponde alle 7:00, è usato nella funzione `inday()`, la funzione determina se la data di ciascuna transazione è avvenuta il 4 gennaio dalle 7:00 e il 5 gennaio prima delle 7:00.

Ciò può essere verificato nella tabella di output dove le transazioni che hanno avuto luogo dopo le 7:00 il 4 gennaio restituiscono un risultato booleano di TRUE mentre le transazioni che hanno avuto luogo dopo le 7:00 il 5 gennaio restituiscono un risultato booleano di FALSE.

Esempio 4 - Oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario utilizzati negli esempi precedenti.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Si calcolerà per determinare se una transazione è avvenuta il 5 gennaio creando una misura in un oggetto grafico.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
9497,'01/01/2022 7:34:46 PM',13.24
9498,'01/01/2022 10:10:22 PM',31.43
9499,'01/02/2022 8:35:54 AM',36.34
9500,'01/03/2022 2:21:53 PM',51.75
9501,'01/04/2022 6:49:38 PM',15.35
9502,'01/04/2022 10:58:34 PM',74.34
9503,'01/05/2022 5:40:49 AM',73.53
9504,'01/05/2022 11:29:38 AM',50.00
9505,'01/05/2022 7:04:57 PM',47.25
9506,'01/06/2022 8:49:09 AM',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

- date

Per calcolare se una transazione è avvenuta il 5 gennaio, creare la seguente misura:

```
=inday(date,'01/05/2022 12:00:00 AM',0)
```

Tabella dei risultati

data	inday(date,'01/05/2022 12:00:00 AM',0)
01/01/2022 7:34:46 PM	0
01/01/2022 10:10:22 PM	0
01/02/2022 8:35:54 AM	0
01/03/2022 2:21:53 PM	0
01/04/2022 6:49:38 PM	0
01/04/2022 10:58:34 PM	0
01/05/2022 5:40:49 AM	-1
01/05/2022 11:29:38 AM	-1
01/05/2022 7:04:57 PM	-1
01/06/2022 8:49:09 AM	0

Esempio 5 – Scenario

Script di caricamento e risultati

Panoramica

In questo esempio, è stato identificato che, a causa di un errore delle apparecchiature, i prodotti fabbricati il 5 gennaio erano difettosi. L'utente finale desidera un oggetto grafico che visualizzi, per data, lo stato dei prodotti fabbricati "difettosi" o "senza difetti" e il costo dei prodotti fabbricati il 5 gennaio.

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata "Prodotti".
- La tabella contiene i seguenti campi:
    - ID prodotto
    - ora di produzione
    - prezzo di costo

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,manufacture_date,cost_price
9497,'01/01/2022 7:34:46 PM',13.24
9498,'01/01/2022 10:10:22 PM',31.43
9499,'01/02/2022 8:35:54 AM',36.34
9500,'01/03/2022 2:21:53 PM',51.75
9501,'01/04/2022 6:49:38 PM',15.35
9502,'01/04/2022 10:58:34 PM',74.34
9503,'01/05/2022 5:40:49 AM',73.53
9504,'01/05/2022 11:29:38 AM',50.00
9505,'01/05/2022 7:04:57 PM',47.25
9506,'01/06/2022 8:49:09 AM',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

```
=dayname(manufacture_date)
```

Creare le seguenti misure:

- `=if(only(InDay(manufacture_date,makedate(2022,01,05),0)),'Defective','Faultless')`
- `=sum(cost_price)`

Impostare la **Formattazione numero** in misura su **Denaro**.

Sotto **Aspetto**, disattivare **Totali**.

Tabella dei risultati

dayname (manufacture_date)	=if(only(InDay(manufacture_date,makedate (2022,01,05),0)),'Defective','Faultless')	=sum(cost_ price)
01/01/2022	Non difettoso	44.67
01/02/2022	Non difettoso	36.34
01/03/2022	Non difettoso	51.75
01/04/2022	Non difettoso	89.69
01/05/2022	Difettoso	170.78
01/06/2022	Non difettoso	74.23

La funzione `inday()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per qualsiasi prodotto fabbricato il 5 gennaio, la funzione `inday()` restituisce un valore booleano TRUE e contrassegna i prodotti come "difettosi". Per tutti i prodotti che restituiscono un valore FALSE, e che quindi non sono stati fabbricati in quel giorno, vengono contrassegnati come "non difettosi".

indaytotime

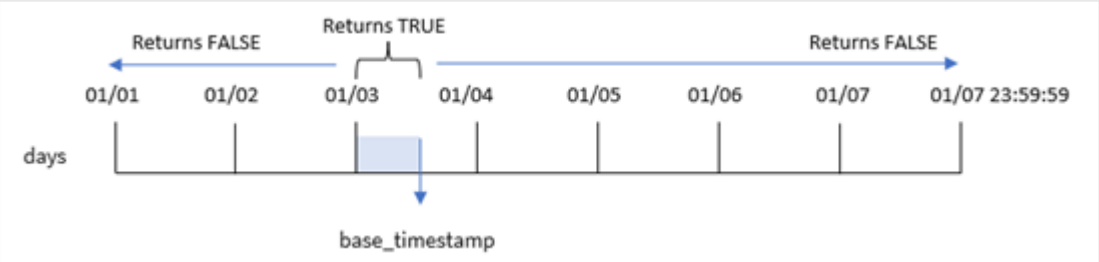
Questa funzione restituisce True se **timestamp** ricade nella parte del giorno contenente **base_timestamp** fino a includere il millisecondo esatto di **base_timestamp**.

Sintassi:

```
InDayToTime (timestamp, base_timestamp, period_no[, day_start])
```

La funzione `indaytotime()` restituisce un risultato booleano a seconda di quando si verifica un valore di timestamp durante il segmento del giorno. Il limite iniziale di questo segmento è l'inizio del giorno, che per impostazione predefinita è la mezzanotte; l'inizio del giorno può essere modificato dall'argomento `day_start` della funzione `indaytotime()`. Il limite finale del segmento giornaliero è determinato da un argomento `base_timestamp` della funzione.

Schema della funzione `indaytotime`.



Casi di utilizzo

La funzione `indaytotime()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. La funzione `indaytotime()` restituisce un'aggregazione o un calcolo a seconda che un timestamp si sia verificato nel segmento del giorno fino al momento del timestamp di base incluso.

Ad esempio, la funzione `indaytotime()` può essere utilizzata per mostrare la somma delle vendite di biglietti per gli spettacoli che hanno avuto luogo fino ad oggi.

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Argomenti

Argomento	Descrizione
`timestamp`	La data e l'ora che si desidera confrontare con `base_timestamp`.
`base_timestamp`	La data e l'ora utilizzate per valutare l'indicatore temporale.
`period_no`	Il giorno può essere differito mediante `period_no`. `period_no` è un numero intero, in cui il valore 0 indica il giorno che contiene `base_timestamp`. I valori negativi di `period_no` indicano i giorni precedenti, mentre i valori positivi indicano i giorni successivi.
`day_start`	(Facoltativo) Se si intende utilizzare giorni che non inizino a mezzanotte, indicare un differimento come frazione di un giorno in `day_start`, ad esempio 0,125 per indicare le 3 del mattino.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`indaytotime` ('01/12/2006 12:23:00 PM', '01/12/2006 11:59:00 PM', 0)	Restituisce True

Esempio	Risultato
`indaytotime` ('01/12/2006 12:23:00 PM', '01/12/2006 12:00:00 AM', 0)	Restituisce False
`indaytotime` '01/11/2006 12:23:00 PM', '01/12/2006 11:59:00 PM', -1)	Restituisce True

Esempio 1 - nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente una serie di transazioni per il periodo tra il 4 e il 5 gennaio viene caricato in una tabella chiamata "Transazioni".
- Un campo data fornito nel formato `TimeStamp` della variabile di sistema `(M/D/YYYY h:mm:ss[.fff] TT)`.
- Un carico precedente che contiene la funzione `indaytotime()` impostata come `'in_day_to_time'`, che determina se ciascuna delle transazioni avviene prima delle 9:00 del mattino.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
    Load
        *,
        indaytotime(date,'01/05/2022 9:00:00 AM',0) as in_day_to_time
        ;
Load
*
Inline
[
id,date,amount
8188,'01/04/2022 3:41:54 AM',25.66
8189,'01/04/2022 4:19:43 AM',87.21
8190,'01/04/2022 4:53:47 AM',53.80
8191,'01/04/2022 8:38:53 AM',69.98
8192,'01/04/2022 10:37:52 AM',57.42
8193,'01/04/2022 1:54:10 PM',45.89
8194,'01/04/2022 5:53:23 PM',82.77
8195,'01/04/2022 8:13:26 PM',36.23
8196,'01/04/2022 10:00:49 PM',76.11
8197,'01/05/2022 7:45:37 AM',82.06
8198,'01/05/2022 8:44:36 AM',17.17
8199,'01/05/2022 11:26:08 AM',40.39
8200,'01/05/2022 6:43:08 PM',37.23
8201,'01/05/2022 10:54:10 PM',88.27
8202,'01/05/2022 11:09:09 PM',95.93
];
```

Risultati

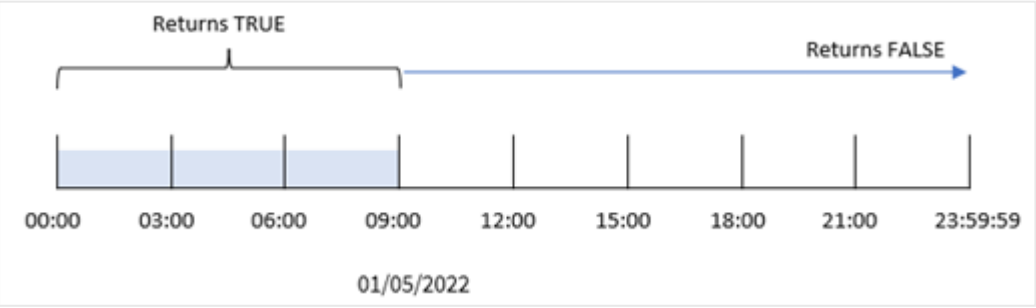
Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_day_to_time`

Tabella dei risultati

data	in_day_to_time
01/04/2022 3:41:54 AM	0
01/04/2022 4:19:43 AM	0
01/04/2022 04:53:47 AM	0
01/04/2022 8:38:53 AM	0
01/04/2022 10:37:52 AM	0
01/04/2022 1:54:10 PM	0
01/04/2022 5:53:23 PM	0
01/04/2022 8:13:26 PM	0
01/04/2022 10:00:49 PM	0
01/05/2022 7:45:37 AM	-1
01/05/2022 8:44:36 AM	-1
01/05/2022 11:26:08 AM	0
01/05/2022 6:43:08 PM	0
01/05/2022 10:54:10 PM	0
01/05/2022 11:09:09 PM	0

Schema 1 di esempio della funzione `indaytotime` con il limite delle 9:00 AM.



`in_day_to_time field` creato nell'istruzione di caricamento precedente mediante l'uso della funzione `indaytotime()` e trasferendo il campo data, un timestamp con hard coding per il 5 gennaio alle 9:00 e un offset di 0 come argomenti della funzione. Qualsiasi transazione che si verifichi tra la mezzanotte e le 9:00 del 5 gennaio restituisce TRUE.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario utilizzati nel primo esempio.

Tuttavia, in questo esempio, verrà calcolato se la data di transazione è occorsa un giorno prima delle 9:00 del 5 gennaio.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
    Load
        *,
        indaytotime(date,'01/05/2022 9:00:00 AM', -1) as in_day_to_time
        ;
Load
*
Inline
[
id,date,amount
8188,'01/04/2022 3:41:54 AM',25.66
8189,'01/04/2022 4:19:43 AM',87.21
8190,'01/04/2022 4:53:47 AM',53.80
8191,'01/04/2022 8:38:53 AM',69.98
8192,'01/04/2022 10:37:52 AM',57.42
8193,'01/04/2022 1:54:10 PM',45.89
8194,'01/04/2022 5:53:23 PM',82.77
8195,'01/04/2022 8:13:26 PM',36.23
8196,'01/04/2022 10:00:49 PM',76.11
8197,'01/05/2022 7:45:37 AM',82.06
8198,'01/05/2022 8:44:36 AM',17.17
8199,'01/05/2022 11:26:08 AM',40.39
8200,'01/05/2022 6:43:08 PM',37.23
8201,'01/05/2022 10:54:10 PM',88.27
8202,'01/05/2022 11:09:09 PM',95.93
];
```

Risultati

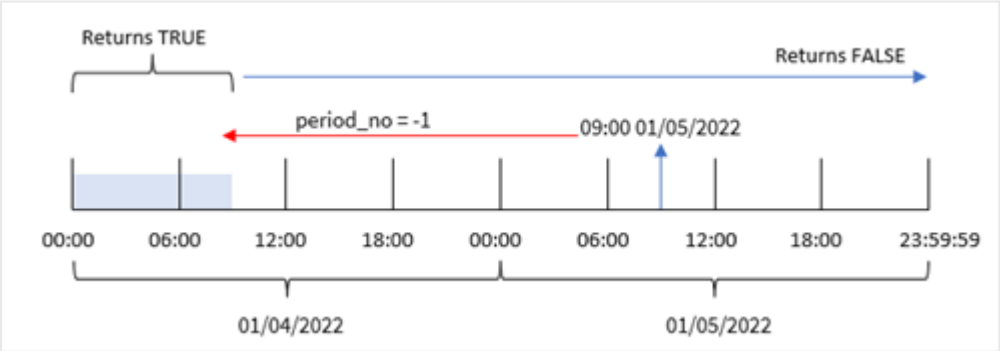
Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- date
- in_day_to_time

Tabella dei risultati

data	in_day_to_time
01/04/2022 3:41:54 AM	-1
01/04/2022 4:19:43 AM	-1
01/04/2022 04:53:47 AM	-1
01/04/2022 8:38:53 AM	-1
01/04/2022 10:37:52 AM	0
01/04/2022 1:54:10 PM	0
01/04/2022 5:53:23 PM	0
01/04/2022 8:13:26 PM	0
01/04/2022 10:00:49 PM	0
01/05/2022 7:45:37 AM	0
01/05/2022 8:44:36 AM	0
01/05/2022 11:26:08 AM	0
01/05/2022 6:43:08 PM	0
01/05/2022 10:54:10 PM	0
01/05/2022 11:09:09 PM	0

Schema esempio 2 della funzione `indaytotime` con transazioni dal 4 gennaio.



In questo esempio, dato che un offset di -1 è stato utilizzato come argomento di offset nella funzione `indaytotime()`, la funzione determina se ciascuna data di transazione ha avuto luogo prima delle 9:00 del 4 gennaio. Ciò è verificabile nella tabella di output dove una transazione restituisce un risultato booleano di TRUE.

Esempio 3 – day_start

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, la policy aziendale indica che la giornata lavorativa inizia e termina alle 8:00.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
    Load
        *,
        indaytotime(date,'01/05/2022 9:00:00 AM', 0,8/24) as in_day_to_time
        ;
Load
*
Inline
[
id,date,amount
8188,'01/04/2022 3:41:54 AM',25.66
8189,'01/04/2022 4:19:43 AM',87.21
8190,'01/04/2022 4:53:47 AM',53.80
8191,'01/04/2022 8:38:53 AM',69.98
8192,'01/04/2022 10:37:52 AM',57.42
8193,'01/04/2022 1:54:10 PM',45.89
8194,'01/04/2022 5:53:23 PM',82.77
8195,'01/04/2022 8:13:26 PM',36.23
8196,'01/04/2022 10:00:49 PM',76.11
8197,'01/05/2022 7:45:37 AM',82.06
8198,'01/05/2022 8:44:36 AM',17.17
8199,'01/05/2022 11:26:08 AM',40.39
8200,'01/05/2022 6:43:08 PM',37.23
8201,'01/05/2022 10:54:10 PM',88.27
8202,'01/05/2022 11:09:09 PM',95.93
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

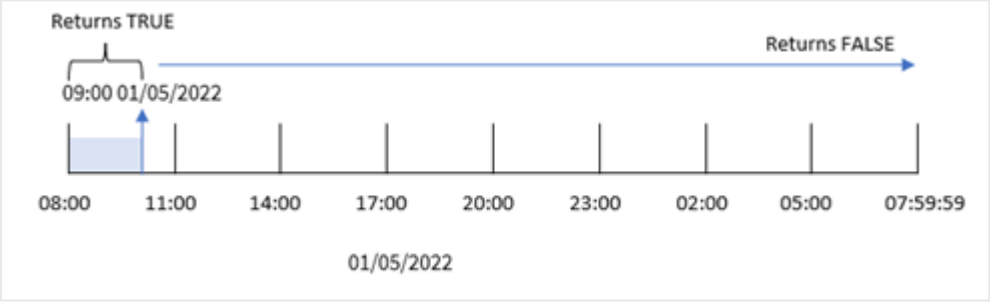
- `date`
- `in_day_to_time`

Tabella dei risultati

data	in_day_to_time
01/04/2022 3:41:54 AM	0
01/04/2022 4:19:43 AM	0
01/04/2022 04:53:47 AM	0

data	in_day_to_time
01/04/2022 8:38:53 AM	0
01/04/2022 10:37:52 AM	0
01/04/2022 1:54:10 PM	0
01/04/2022 5:53:23 PM	0
01/04/2022 8:13:26 PM	0
01/04/2022 10:00:49 PM	0
01/05/2022 7:45:37 AM	0
01/05/2022 8:44:36 AM	-1
01/05/2022 11:26:08 AM	0
01/05/2022 6:43:08 PM	0
01/05/2022 10:54:10 PM	0
01/05/2022 11:09:09 PM	0

Schema esempio 3 della funzione `indaytotime` con transazioni dalle 8:00 alle 9:00.



Poiché nella funzione `indaytotime()` viene utilizzato l'argomento `start_day` 8/24, che equivale alle 8:00 del mattino, ogni giorno inizia e finisce alle 8:00 del mattino. Pertanto, la funzione `indaytotime()` restituirà un risultato booleano di TRUE per qualsiasi transazione avvenuta tra le 8:00 e le 9:00 del 5 gennaio.

Esempio 4 - Oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Si calcolerà per determinare se una transazione è avvenuta il 5 gennaio prima delle 9:00 creando una misura in un oggetto grafico.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,'01/04/2022 3:41:54 AM',25.66
8189,'01/04/2022 4:19:43 AM',87.21
8190,'01/04/2022 4:53:47 AM',53.80
8191,'01/04/2022 8:38:53 AM',69.98
8192,'01/04/2022 10:37:52 AM',57.42
8193,'01/04/2022 1:54:10 PM',45.89
8194,'01/04/2022 5:53:23 PM',82.77
8195,'01/04/2022 8:13:26 PM',36.23
8196,'01/04/2022 10:00:49 PM',76.11
8197,'01/05/2022 7:45:37 AM',82.06
8198,'01/05/2022 8:44:36 AM',17.17
8199,'01/05/2022 11:26:08 AM',40.39
8200,'01/05/2022 6:43:08 PM',37.23
8201,'01/05/2022 10:54:10 PM',88.27
8202,'01/05/2022 11:09:09 PM',95.93
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

`date`.

Per determinare se una transazione è avvenuta il 5 gennaio prima delle 9:00, creare la seguente misura:

```
=indaytotime(date,'01/05/2022 9:00:00 AM',0)
```

Tabella dei risultati

data	=indaytotime(date,'01/05/2022 9:00:00 AM',0)
01/04/2022 3:41:54 AM	0
01/04/2022 4:19:43 AM	0
01/04/2022 04:53:47 AM	0
01/04/2022 8:38:53 AM	0
01/04/2022 10:37:52 AM	0
01/04/2022 1:54:10 PM	0
01/04/2022 5:53:23 PM	0
01/04/2022 8:13:26 PM	0
01/04/2022 10:00:49 PM	0

data	=indaytotime(date,'01/05/2022 9:00:00 AM',0)
01/05/2022 7:45:37 AM	-1
01/05/2022 8:44:36 AM	-1
01/05/2022 11:26:08 AM	0
01/05/2022 6:43:08 PM	0
01/05/2022 10:54:10 PM	0
01/05/2022 11:09:09 PM	0

La misura `in_day_to_time` viene creata nell'oggetto grafico utilizzando la funzione `indaytotime()` e passando come argomenti della funzione il campo data, un timestamp codificato per le 9:00 del 5 gennaio e un offset di `0`. Qualsiasi transazione che si verifichi tra la mezzanotte e le 9:00 del 5 gennaio restituisce TRUE. Questo viene convalidato nella tabella dei risultati.

Esempio 5 – Scenario

Script di caricamento e risultati

Panoramica

In questo esempio, un set di dati contenente le vendite di biglietti di un cinema locale viene caricato in una tabella chiamata Ticket_Sales. Oggi è il 3 maggio 2022 e sono le 11:00 del mattino.

L'utente desidera un oggetto grafico KPI che mostri i ricavi di tutti gli spettacoli che si sono svolti oggi.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Ticket_Sales:
Load
*
Inline
[
sale ID, show time, ticket price
1,05/01/2022 09:30:00 AM,10.50
2,05/03/2022 05:30:00 PM,21.00
3,05/03/2022 09:30:00 AM,10.50
4,05/03/2022 09:30:00 AM,31.50
5,05/03/2022 09:30:00 AM,10.50
6,05/03/2022 12:00:00 PM,42.00
7,05/03/2022 12:00:00 PM,10.50
8,05/03/2022 05:30:00 PM,42.00
9,05/03/2022 08:00:00 PM,31.50
10,05/04/2022 10:30:00 AM,31.50
11,05/04/2022 12:00:00 PM,10.50
12,05/04/2022 05:30:00 PM,10.50
13,05/05/2022 05:30:00 PM,21.00
14,05/06/2022 12:00:00 PM,21.00
15,05/07/2022 09:30:00 AM,42.00
```

```
16,05/07/2022 10:30:00 AM,42.00
17,05/07/2022 10:30:00 AM,10.50
18,05/07/2022 05:30:00 PM,10.50
19,05/08/2022 05:30:00 PM,21.00
20,05/11/2022 09:30:00 AM,10.50
];
```

Risultati

Procedere come segue:

1. Creare un oggetto KPI.
2. Creare una misura che mostri la somma di tutte le vendite di biglietti per gli spettacoli che si sono svolti oggi fino ad ora, utilizzando la funzione `indaytotime()`:

```
=sum(if(indaytotime([show time],'05/03/2022 11:00:00 AM',0),[ticket price],0))
```

3. Creare un'etichetta per l'oggetto KPI "Entrate correnti".
4. Impostare la **Formattazione numero** in misura su **Denaro**.

Il totale delle vendite dei biglietti fino alle 11:00 del 3 maggio 2022 è di 52,50 dollari.

La funzione `indaytotime ()` restituisce un valore booleano quando si confrontano gli orari degli spettacoli di ciascuna vendita di biglietti con l'ora corrente ('05/03/2022 11:00:00 AM'). Per qualsiasi spettacolo del 3 maggio prima delle 11:00, la funzione `indaytotime ()` restituisce il valore booleano TRUE e il prezzo del biglietto sarà incluso nella somma totale.

inlunarweek

Questa funzione determina se **timestamp** ricade all'interno della settimana lunare contenente **base_date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana, a parte l'ultima settimana dell'anno, ogni settimana conterrà esattamente sette giorni.

Sintassi:

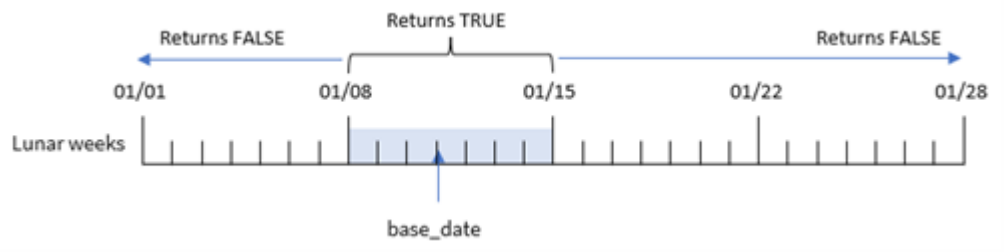
```
InLunarWeek (timestamp, base_date, period_no[, first_week_day])
```

Tipo di dati restituiti: Booleano

> *In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.*

La funzione `inlunarweek()` determina la settimana lunare in cui cade `base_date`. Restituisce quindi un risultato booleano dopo aver determinato se ogni valore di timestamp si verifica nella stessa settimana lunare del valore `base_date`.

Schema della funzione `inlunarweek()`.



Casi di utilizzo

La funzione `inlunarweek()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un'espressione IF. Questo restituisce un'aggregazione o un calcolo che dipende dal fatto che la data valutata si sia verificata durante la settimana lunare in questione.

Ad esempio, la funzione `inlunarweek()` può essere utilizzata per identificare tutte le apparecchiature prodotte in una particolare settimana lunare.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare la settimana lunare.
period_no	La settimana lunare può essere differita mediante **period_no**. period_no è un numero intero, in cui il valore 0 indica la settimana lunare che contiene **base_date**. I valori negativi di **period_no** indicano le settimane lunari precedenti, mentre i valori positivi indicano le settimane lunari successive.
first_week_day	Un differimento che può essere maggiore o minore di zero. Ciò modifica l'inizio dell'anno in base al numero specificato di giorni e/o frazioni di un giorno.

Esempi di funzioni

Esempio	Risultato
`inlunarweek ('01/12/2013', '01/14/2013', 0)`	Restituisce il valore `TRUE`, dato che il valore di `timestamp`, 01/12/2013, ricade nella settimana 01/08/2013 a 01/14/2013.
`inlunarweek ('01/12/2013', '01/07/2013', 0)`	Restituisce `FALSE`, dato che il valore di `base_date` 01/07/2013 è nella settimana lunare definita come 01/01/2013 a 01/07/2013.
`inlunarweek ('01/12/2013', '01/14/2013', -1)`	Restituisce `FALSE`. La specifica di un valore di period_no come -1 fa slittare la settimana a quella precedente, 01/01/2013 a 01/07/2013.
`inlunarweek ('01/07/2013', 01/14/2013', -1)`	Restituisce `TRUE`. In confronto con l'esempio precedente, il valore `timestamp` è nella settimana precedente, dopo aver considerato lo slittamento all'indietro.

Esempio	Risultato
`inlunarweek ('01/11/2006', '01/08/2006', 0, 3)`	Restituisce `FALSE`. La specifica di un valore di 3 per `first_week_day` significa che l'avvio dell'anno è calcolato da 01/04/2013. Pertanto il valore di `base_`date ricade nella prima settimana mentre il valore di `timestamp` ricade nella settimana da 01/11/2013 a 01/17/2013.

La funzione `inlunarweek()` viene spesso utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
lunarweekname (page 853)	Questa funzione viene utilizzata per determinare il numero della settimana lunare dell'anno in cui si verifica una data inserita.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di transazioni per il mese di gennaio, che viene caricato in una tabella chiamata `Transactions`.
- Il campo della data è stato fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

Crea un campo, `in_lunar_week`, che determina se le transazioni sono avvenute nella stessa settimana lunare del 10 gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inlunarweek(date,'01/10/2022', 0) as in_lunar_week
        ;
Load
*
Inline
[
id,date,amount
8183,'1/5/2022',42.32
8184,'1/6/2022',68.22
8185,'1/7/2022',15.25
8186,'1/8/2022',25.26
8187,'1/9/2022',37.23
8188,'1/10/2022',37.23
8189,'1/11/2022',17.17
8190,'1/12/2022',88.27
8191,'1/13/2022',57.42
8192,'1/14/2022',53.80
8193,'1/15/2022',82.06
8194,'1/16/2022',87.21
8195,'1/17/2022',95.93
8196,'1/18/2022',45.89
8197,'1/19/2022',36.23
8198,'1/20/2022',25.66
8199,'1/21/2022',82.77
8200,'1/22/2022',69.98
8201,'1/23/2022',76.11
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

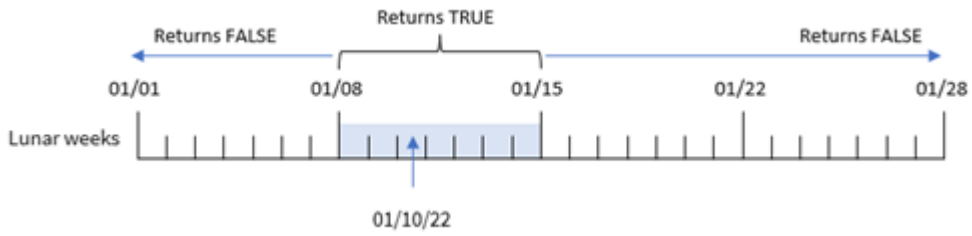
- `date`
- `in_lunar_week`

Tabella dei risultati

date	in_lunar_week
1/5/2022	0
1/6/2022	0
1/7/2022	0
1/8/2022	-1

date	in_lunar_week
1/9/2022	-1
1/10/2022	-1
1/11/2022	-1
1/12/2022	-1
1/13/2022	-1
1/14/2022	-1
1/15/2022	0
1/16/2022	0
1/17/2022	0
1/18/2022	0
1/19/2022	0
1/20/2022	0
1/21/2022	0
1/22/2022	0
1/23/2022	0

Funzione `inlunarweek()`, esempio base



Il campo `in_lunar_week` viene creato nella precedente istruzione LOAD utilizzando la funzione `inlunarweek()` e passando i seguenti argomenti alla funzione:

- Il campo `date`
- La data del 10 gennaio, codificata in modo rigido, è `base_date`
- A`period_no` di 0

Poiché le settimane lunari iniziano il 1° gennaio, il 10 gennaio cadrebbe nella settimana lunare che inizia l'8 gennaio e termina il 14 gennaio. Pertanto, tutte le transazioni che si verificano tra queste due date di gennaio restituiranno un valore booleano di `TRUE`. Questo viene convalidato nella tabella dei risultati.

Esempio 2 - period_no

Esempi e risultati:

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- Il campo della data è stato fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

Tuttavia, in questo esempio, il compito è quello di creare un campo, `2_lunar_weeks_later`, che determini se le transazioni sono avvenute o meno due settimane lunari dopo il 10 gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inlunarweek(date,'01/10/2022', 2) as [2_lunar_weeks_later]
        ;
Load
*
Inline
[
id,date,amount
8183,'1/5/2022',42.32
8184,'1/6/2022',68.22
8185,'1/7/2022',15.25
8186,'1/8/2022',25.26
8187,'1/9/2022',37.23
8188,'1/10/2022',37.23
8189,'1/11/2022',17.17
8190,'1/12/2022',88.27
8191,'1/13/2022',57.42
8192,'1/14/2022',53.80
8193,'1/15/2022',82.06
8194,'1/16/2022',87.21
8195,'1/17/2022',95.93
8196,'1/18/2022',45.89
8197,'1/19/2022',36.23
8198,'1/20/2022',25.66
8199,'1/21/2022',82.77
8200,'1/22/2022',69.98
8201,'1/23/2022',76.11
];
```

Risultati

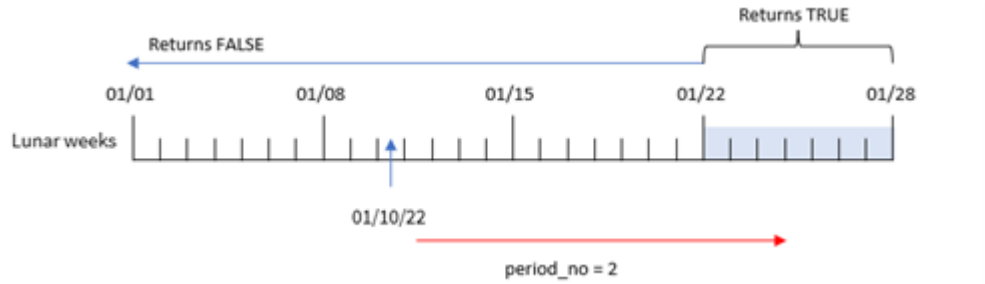
Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `2_lunar_weeks_later`

Tabella dei risultati

date	2_lunar_weeks_later
1/5/2022	0
1/6/2022	0
1/7/2022	0
1/8/2022	0
1/9/2022	0
1/10/2022	0
1/11/2022	0
1/12/2022	0
1/13/2022	0
1/14/2022	0
1/15/2022	0
1/16/2022	0
1/17/2022	0
1/18/2022	0
1/19/2022	0
1/20/2022	0
1/21/2022	0
1/22/2022	-1
1/23/2022	-1

Funzione `inlunarweek()`, esempio period_no



In questo caso, poiché è stato utilizzato un valore `period_no` di 2 come argomento di offset nella funzione `inlunarweek()`, la funzione definisce la settimana che inizia il 22 gennaio come settimana lunare rispetto alla quale convalidare le transazioni. Pertanto, qualsiasi transazione effettuata tra il 22 e il 28 gennaio restituirà un risultato booleano di `TRUE`.

Esempio 3 - first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, nell'esempio, abbiamo impostato le settimane lunari a partire dal 6 gennaio.

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- Un argomento `first_week_day` di 5. In questo modo le settimane lunari iniziano il 5 gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inlunarweek(date,'01/10/2022', 0,5) as in_lunar_week
        ;
Load
*
Inline
[
id,date,amount
8183,'1/5/2022',42.32
8184,'1/6/2022',68.22
8185,'1/7/2022',15.25
8186,'1/8/2022',25.26
8187,'1/9/2022',37.23
8188,'1/10/2022',37.23
```

```
8189,'1/11/2022',17.17
8190,'1/12/2022',88.27
8191,'1/13/2022',57.42
8192,'1/14/2022',53.80
8193,'1/15/2022',82.06
8194,'1/16/2022',87.21
8195,'1/17/2022',95.93
8196,'1/18/2022',45.89
8197,'1/19/2022',36.23
8198,'1/20/2022',25.66
8199,'1/21/2022',82.77
8200,'1/22/2022',69.98
8201,'1/23/2022',76.11
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

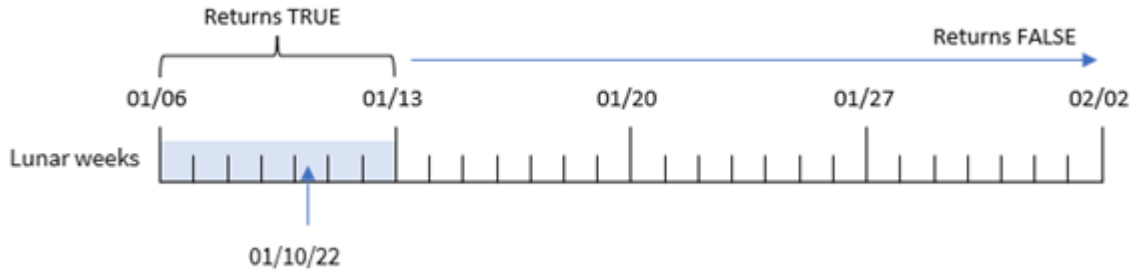
- `date`
- `in_lunar_week`

Tabella dei risultati

date	in_lunar_week
1/5/2022	0
1/6/2022	-1
1/7/2022	-1
1/8/2022	-1
1/9/2022	-1
1/10/2022	-1
1/11/2022	-1
1/12/2022	-1
1/13/2022	0
1/14/2022	0
1/15/2022	0
1/16/2022	0
1/17/2022	0
1/18/2022	0
1/19/2022	0
1/20/2022	0
1/21/2022	0

date	in_lunar_week
1/22/2022	0
1/23/2022	0

Funzione `inlunarweek()`, esempio first_week_day



In questo caso, poiché nella funzione `inlunarweek()` viene utilizzato l'argomento `first_week_date` 5, si sposta l'inizio della settimana lunare al 6 gennaio. Pertanto, il 10 gennaio cade nella settimana lunare che inizia il 6 gennaio e termina il 12 gennaio. Qualsiasi transazione che cade tra queste due date restituirà un valore booleano di `TRUE`.

Esempio 4 - Oggetto grafico

Script di caricamento ed espressione del grafico:

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- Il campo della data è stato fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che determina se le transazioni sono avvenute nella stessa settimana lunare del 10 gennaio viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8183,'1/5/2022',42.32
```

```
8184,'1/6/2022',68.22
8185,'1/7/2022',15.25
8186,'1/8/2022',25.26
8187,'1/9/2022',37.23
8188,'1/10/2022',37.23
8189,'1/11/2022',17.17
8190,'1/12/2022',88.27
8191,'1/13/2022',57.42
8192,'1/14/2022',53.80
8193,'1/15/2022',82.06
8194,'1/16/2022',87.21
8195,'1/17/2022',95.93
8196,'1/18/2022',45.89
8197,'1/19/2022',36.23
8198,'1/20/2022',25.66
8199,'1/21/2022',82.77
8200,'1/22/2022',69.98
8201,'1/23/2022',76.11
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Per calcolare se una transazione avviene nella settimana lunare che contiene il 10 gennaio, creare la seguente misura:

```
= inlunarweek(date,'01/10/2022', 0)
```

Tabella dei risultati

date	=inlunarweek(date,'01/10/2022', 0)
1/5/2022	0
1/6/2022	0
1/7/2022	0
1/8/2022	-1
1/9/2022	-1
1/10/2022	-1
1/11/2022	-1
1/12/2022	-1
1/13/2022	-1
1/14/2022	-1
1/15/2022	0

date	=inlunarweek(date,'01/10/2022', 0)
1/16/2022	0
1/17/2022	0
1/18/2022	0
1/19/2022	0
1/20/2022	0
1/21/2022	0
1/22/2022	0
1/23/2022	0

Esempio 5 - Scenario

Script di caricamento ed espressione del grafico:

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Products`.
- Informazioni composte da ID prodotto, data di produzione e prezzo di costo.

È stato individuato che, a causa di un errore delle apparecchiature, i prodotti fabbricati nella settimana lunare che comprendeva il 12 gennaio erano difettosi. L'utente finale desidera un oggetto grafico che visualizzi, per nome della settimana lunare, lo stato dei prodotti fabbricati "difettosi" o "non difettosi" e il costo dei prodotti fabbricati in quel mese.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
product_id,manufacture_date,cost_price
8183,'1/5/2022',42.32
8184,'1/6/2022',68.22
8185,'1/7/2022',15.25
8186,'1/8/2022',25.26
8187,'1/9/2022',37.23
8188,'1/10/2022',37.23
8189,'1/11/2022',17.17
8190,'1/12/2022',88.27
```

```
8191,'1/13/2022',57.42
8192,'1/14/2022',53.80
8193,'1/15/2022',82.06
8194,'1/16/2022',87.21
8195,'1/17/2022',95.93
8196,'1/18/2022',45.89
8197,'1/19/2022',36.23
8198,'1/20/2022',25.66
8199,'1/21/2022',82.77
8200,'1/22/2022',69.98
8201,'1/23/2022',76.11
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Creare una dimensione per mostrare i nomi dei mesi:
   `=lunarweekname(manufacture_date)`
3. Creare una misura per identificare quali prodotti sono difettosi e quali sono privi di difetti utilizzando la funzione `inlunarweek()`:
   `=if(only(inlunarweek(manufacture_date,makedate(2022,01,12),0)), 'Defective','Faultless')`
4. Creare una misura per sommare il valore `cost_price` dei prodotti:
   `=sum(cost_price)`
5. Impostare la **Formattazione numero** della misura su **Denaro**.
6. In **Aspetto**, disattivare **Totals**.

Tabella dei risultati

lunarweekname (manufacture_date)	=if(only(inlunarweek(manufacture_date,makedate (2022,01,12),0)), 'Defective','Faultless')	sum(cost_ price)
2022/01	Non difettoso	$125.79
2022/02	Difettoso	$316.38
2022/03	Non difettoso	$455.75
2022/04	Non difettoso	$146.09

La funzione `inlunarweek()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per qualsiasi prodotto fabbricato nella settimana lunare che contiene il 10 gennaio, la funzione `inlunarweek()` restituisce un valore booleano di `TRUE` e contrassegna i prodotti come "difettosi". Tutti i prodotti che restituiscono un valore pari a `FALSE`, e che quindi non sono stati prodotti in quella settimana, vengono contrassegnati come "non difettosi".

inlunarweektodate

Questa funzione stabilisce se **timestamp** ricade all'interno della parte della settimana lunare fino a includere l'ultimo millisecondo di **base_date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana e, a parte l'ultima settimana dell'anno, conterranno esattamente sette giorni.

Sintassi:

```
InLunarWeekToDate (timestamp, base_date, period_no [, first_week_day])
```

Tipo di dati restituiti: Booleano

> *In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.*

Schema esemplificativo della funzione `inlunarweektodate()`

La funzione `inlunarweektodate()` funge da punto finale della settimana lunare. In contrasto, la funzione `inlunarweek()` determina la settimana lunare in cui cade `base_date`. Ad esempio, se il `base_date` fosse il 5 gennaio, qualsiasi timestamp tra il 1 e il 5 gennaio restituirebbe un risultato booleano di `TRUE`, mentre le date del 6 e 7 gennaio e successive restituirebbero un risultato booleano di `FALSE`.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare la settimana lunare.
period_no	La settimana lunare può essere differita mediante **period_no**. period_no è un numero intero, in cui il valore 0 indica la settimana lunare che contiene **base_date**. I valori negativi di **period_no** indicano le settimane lunari precedenti, mentre i valori positivi indicano le settimane lunari successive.
first_week_day	Un differimento che può essere maggiore o minore di zero. Ciò modifica l'inizio dell'anno in base al numero specificato di giorni e/o frazioni di un giorno.

Casi di utilizzo

La funzione `inlunarweektodate()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un'espressione IF. La funzione `inlunarweektodate()` viene utilizzata quando l'utente desidera che il calcolo restituisca un'aggregazione o un calcolo, a seconda che la data valutata sia avvenuta in un particolare segmento della settimana in questione.

Ad esempio, la funzione `inlunarweektodate()` può essere utilizzata per identificare tutte le apparecchiature prodotte in una determinata settimana fino a una data specifica.

Esempi di funzioni

Esempio	Risultato
`inlunarweektodate ('01/12/2013', '01/13/2013', 0)`	Restituisce `TRUE`, dato che il valore di `timestamp`, 01/12/2013, ricade nella parte della settimana da 01/08/2013 a 01/13/2013.
`inlunarweektodate ('01/12/2013', '01/11/2013', 0)`	Restituisce `FALSE`, poiché il valore di `timestamp` è successivo al valore di `base_date`, anche se le due date si trovano nella stessa settimana lunare prima di 01/12/2012.
`inlunarweektodate ('01/12/2006', '01/05/2006', 1)`	Restituisce `TRUE`. Specificando un lavoro pari a 1 per `period_no`, `base_date` slitta in avanti di una settimana, pertanto il valore di`timestamp` ricade nella parte della settimana lunare.

La funzione `inlunarweektodate()` viene spesso utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
lunarweekname (page 853)	Questa funzione viene utilizzata per determinare il numero della settimana lunare dell'anno in cui si verifica una data inserita.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il mese di gennaio, caricato in una tabella denominata `Transactions`. È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- Creare un campo, `in_lunar_week_to_date`, che determina se le transazioni sono avvenute nella settimana lunare alla data del 10 gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inlunarweektodate(date,'01/10/2022', 0) as in_lunar_week_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'1/10/2022',37.23
8189,'1/17/2022',17.17
8190,'1/26/2022',88.27
8191,'1/12/2022',57.42
8192,'1/19/2022',53.80
8193,'1/21/2022',82.06
8194,'1/1/2022',40.39
8195,'1/27/2022',87.21
8196,'1/11/2022',95.93
8197,'1/29/2022',45.89
8198,'1/31/2022',36.23
8199,'1/18/2022',25.66
8200,'1/23/2022',82.77
8201,'1/15/2022',69.98
8202,'1/4/2022',76.11
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- date
- in_lunar_week_to_date

Tabella dei risultati

date	in_lunar_week_to_date
1/1/2022	0
1/4/2022	0
1/10/2022	-1
1/11/2022	0
1/12/2022	0
1/15/2022	0
1/17/2022	0
1/18/2022	0
1/19/2022	0
1/21/2022	0
1/23/2022	0
1/26/2022	0
1/27/2022	0
1/29/2022	0
1/31/2022	0

Funzione `inlunarweektodate()`, senza argomenti aggiuntivi

Il campo `in_lunar_week_to_date` viene creato nell'istruzione di caricamento precedente mediante l'uso della funzione `inlunarweektodate()` e trasferendo il campo `date`, una data con hard coding per il 10 gennaio come nostro `base_date`, e un offset di 0 come argomenti della funzione.

Poiché le settimane lunari iniziano il 1° gennaio, il 10 gennaio cadrebbe nella settimana lunare che inizia l'8 gennaio; e poiché stiamo utilizzando la funzione `inlunarweektodate()`, quella settimana lunare finirebbe il 10. Pertanto, tutte le transazioni che si verificano tra queste due date di gennaio restituiranno un valore booleano di `TRUE`. Questo viene convalidato nella tabella dei risultati.

Esempio 2 - period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il compito è quello di creare un campo, `2_lunar_weeks_later`, che determini se le transazioni sono avvenute o meno due settimane dopo la settimana lunare alla data del 1° gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
Transactions:
    Load
        *,
        inlunarweektodate(date,'01/10/2022', 2) as [2_lunar_weeks_later]
        ;
Load
*
Inline
[
id,date,amount
8188,'1/10/2022',37.23
8189,'1/17/2022',17.17
8190,'1/26/2022',88.27
8191,'1/12/2022',57.42
8192,'1/19/2022',53.80
8193,'1/21/2022',82.06
8194,'1/1/2022',40.39
8195,'1/27/2022',87.21
8196,'1/11/2022',95.93
8197,'1/29/2022',45.89
8198,'1/31/2022',36.23
8199,'1/18/2022',25.66
8200,'1/23/2022',82.77
8201,'1/15/2022',69.98
8202,'1/4/2022',76.11
];
```

Risultati

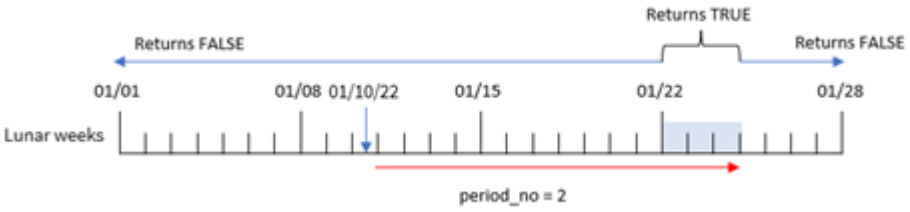
Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `2_lunar_weeks_later`

Tabella dei risultati

date	2_lunar_weeks_later
1/1/2022	0
1/4/2022	0
1/10/2022	0
1/11/2022	0
1/12/2022	0
1/15/2022	0
1/17/2022	0
1/18/2022	0
1/19/2022	0
1/21/2022	0
1/23/2022	-1
1/26/2022	0
1/27/2022	0
1/29/2022	0
1/31/2022	0

Funzione `inlunarweektodate()`, esempio period_no



In questo caso, la funzione `inlunarweektodate()` determina che la settimana lunare fino al 10 gennaio equivale a tre giorni (8, 9, 10 gennaio). Poiché è stato utilizzato un valore `period_no` di 2 come argomento di offset, questa settimana lunare è spostata di 14 giorni. Pertanto, la settimana lunare di tre giorni comprende il 22, 23 e 24 gennaio. Qualsiasi transazione effettuata tra il 22 e il 24 gennaio restituirà un risultato booleano di `TRUE`.

Esempio 3 - first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- È utilizzata la variabile di sistema predefinita `DateFormat` MM/GG/AAAA.
- Un argomento `first_week_date` di 3. In questo modo le settimane lunari iniziano il 3 gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inlunarweek(date,'01/10/2022', 0,3) as in_lunar_week_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'1/10/2022',37.23
8189,'1/17/2022',17.17
8190,'1/26/2022',88.27
8191,'1/12/2022',57.42
8192,'1/19/2022',53.80
8193,'1/21/2022',82.06
8194,'1/1/2022',40.39
8195,'1/27/2022',87.21
8196,'1/11/2022',95.93
8197,'1/29/2022',45.89
8198,'1/31/2022',36.23
8199,'1/18/2022',25.66
8200,'1/23/2022',82.77
8201,'1/15/2022',69.98
8202,'1/4/2022',76.11
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

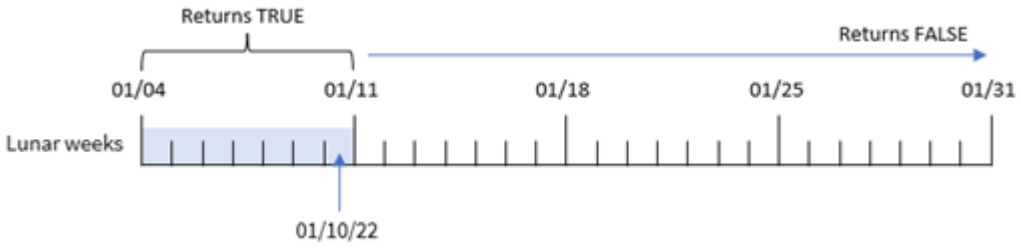
- `date`
- `in_lunar_week_to_date`

Tabella dei risultati

date	in_lunar_week_to_date
1/1/2022	0
1/4/2022	-1

date	in_lunar_week_to_date
1/10/2022	-1
1/11/2022	0
1/12/2022	0
1/15/2022	0
1/17/2022	0
1/18/2022	0
1/19/2022	0
1/21/2022	0
1/23/2022	0
1/26/2022	0
1/27/2022	0
1/29/2022	0
1/31/2022	0

Funzione `inlunarweektodate()`, esempio first_week_day



In questo caso, poiché nella funzione `inlunarweek()` viene utilizzato l'argomento 3 `the first_week_date`, la prima settimana lunare andrà dal 3 al 10 gennaio. Poiché il 10 gennaio è anche il `base_date`, qualsiasi transazione che cade tra queste due date restituirà un valore booleano di `TRUE`.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che determina se le transazioni sono avvenute nella stessa settimana lunare fino al 10 gennaio viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'1/10/2022',37.23
8189,'1/17/2022',17.17
8190,'1/26/2022',88.27
8191,'1/12/2022',57.42
8192,'1/19/2022',53.80
8193,'1/21/2022',82.06
8194,'1/1/2022',40.39
8195,'1/27/2022',87.21
8196,'1/11/2022',95.93
8197,'1/29/2022',45.89
8198,'1/31/2022',36.23
8199,'1/18/2022',25.66
8200,'1/23/2022',82.77
8201,'1/15/2022',69.98
8202,'1/4/2022',76.11
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Creare la seguente misura:

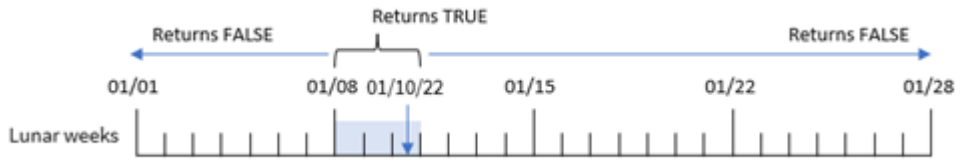
```
=inlunarweektodate(date,'01/10/2022', 0)
```

Tabella dei risultati

date	=inlunarweektodate(date,'01/10/2022', 0)
1/1/2022	0
1/4/2022	0
1/10/2022	-1
1/11/2022	0
1/12/2022	0

date	=inlunarweektodate(date,'01/10/2022', 0)
1/15/2022	0
1/17/2022	0
1/18/2022	0
1/19/2022	0
1/21/2022	0
1/23/2022	0
1/26/2022	0
1/27/2022	0
1/29/2022	0
1/31/2022	0

Funzione `inlunarweektodate()`, esempio di oggetto grafico



La misura `in_lunar_week_to_date` viene creata nell'oggetto grafico utilizzando la funzione `inlunarweektodate()` e passando al campo data, una data con hard-coding per il 10 gennaio come nostro `base_date`, e un offset di 0 come argomenti della funzione.

Poiché le settimane lunari iniziano il 1° gennaio, il 10 gennaio cadrebbe nella settimana lunare che inizia l'8 gennaio. Inoltre, dal momento che stiamo utilizzando la funzione `inlunarweektodate()`, la settimana lunare terminerebbe il 10. Pertanto, tutte le transazioni che si verificano tra queste due date di gennaio restituiranno un valore booleano di `TRUE`. Questo viene convalidato nella tabella dei risultati.

Esempio 5 - Scenario

Script di caricamento ed espressioni del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Products`.
- Informazioni composte da ID prodotto, data di produzione e prezzo di costo.

Si è determinato che, a causa di un errore delle apparecchiature, i prodotti fabbricati nella settimana lunare del 12 gennaio erano difettosi. Il problema è stato risolto il 13 gennaio. L'utente finale desidera un oggetto grafico che visualizzi, per settimana, se i prodotti fabbricati sono "difettosi" o "non difettosi" e il costo dei prodotti fabbricati quella settimana.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff]';

Products:
Load
*
Inline
[
product_id,manufacture_date,cost_price
8188,'01/02/2022 12:22:06',37.23
8189,'01/05/2022 01:02:30',17.17
8190,'01/06/2022 15:36:20',88.27
8191,'01/08/2022 10:58:35',57.42
8192,'01/09/2022 08:53:32',53.80
8193,'01/10/2022 21:13:01',82.06
8194,'01/11/2022 00:57:13',40.39
8195,'01/12/2022 09:26:02',87.21
8196,'01/13/2022 15:05:09',95.93
8197,'01/14/2022 18:44:57',45.89
8198,'01/15/2022 06:10:46',36.23
8199,'01/16/2022 06:39:27',25.66
8200,'01/17/2022 10:44:16',82.77
8201,'01/18/2022 18:48:17',69.98
8202,'01/26/2022 04:36:03',76.11
8203,'01/27/2022 08:07:49',25.12
8204,'01/28/2022 12:24:29',46.23
8205,'01/30/2022 11:56:56',84.21
8206,'01/30/2022 14:40:19',96.24
8207,'01/31/2022 05:28:21',67.67
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Creare una dimensione per visualizzare i nomi delle settimane:
   `=weekname(manufacture_date)`
3. Quindi, creare una dimensione che utilizzi la funzione `inlunarweektodate()` per identificare quali prodotti sono difettosi e quali non lo sono:
   `=if(inlunarweektodate(manufacture_date,makedate(2022,01,12),0),'Defective','Faultless')`
4. Creare una misura per sommare il valore `cost_price` dei prodotti:
   `=sum(cost_price)`
5. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

=lunarweekname (manufacture_date)	=if(InLunarWeekToDate(manufacture_date,makedate (2022,01,12),0),'Defective','Faultless')	=Sum(cost_ price)
2022/01	Non difettoso	$142.67
2022/02	Difettoso	$320.88
2022/02	Non difettoso	$141.82
2022/03	Non difettoso	$214.64
2022/04	Non difettoso	$147.46
2022/05	Non difettoso	$248.12

La funzione `inlunarweektodate()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per coloro che restituiscono il valore booleano `TRUE`, contrassegna i prodotti come `'Defective'`. Per qualsiasi prodotto che restituisce il valore `FALSE`, e quindi non è stato realizzato nella settimana lunare fino al 12 gennaio, contrassegna i prodotti come `'Faultless'`.

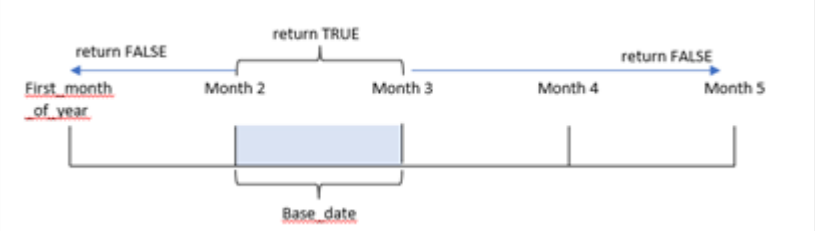
inmonth

Questa funzione restituisce True se **timestamp** ricade all'interno del mese contenente **base_ date**.

Sintassi:

```
InMonth (timestamp, base_date, period_no)
```

Schema della funzione `indaytotime`.



In pratica, la funzione `inmonth()` determina se un insieme di date rientra in questo mese e restituisce un valore booleano basato su un valore `base_date` che identifica il mese.

Casi di utilizzo

La funzione `inmonth()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. Restituisce un'aggregazione o un calcolo che dipende se una data si è verificata durante quel mese, unclusa la data in questione.

Ad esempio, la funzione `inmonth()` può essere utilizzata per identificare tutte le apparecchiature prodotte in un determinato giorno.

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Argomenti

Argomento	Descrizione
data e ora	La data da confrontare con `base_date`.
base_date	La data utilizzata per valutare il mese. È importante notare che `base_date` può essere qualsiasi giorno entro un mese.
period_no	Il mese può essere differito mediante `period_no`. `period_no` è un numero intero, in cui il valore 0 indica il mese che contiene `base_date`. I valori negativi di `period_no` indicano i mesi precedenti, mentre i valori positivi indicano i mesi successivi.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`inmonth ('25/01/2013', '01/01/2013', 0)`	Restituisce True
`inmonth('25/01/2013', '23/04/2013', 0)`	Restituisce False
`inmonth ('25/01/2013', '01/01/2013', -1)`	Restituisce False
`inmonth ('25/12/2012', '17/01/2013', -1)`	Restituisce True

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per la prima metà del 2022.
- Un'istruzione LOAD precedente con una variabile aggiuntiva, '`in_month`', che determina se le transazioni sono state effettuate nel mese di aprile.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
  Load
    *,
    inmonth(date,'04/01/2022', 0) as in_month
    ;
Load
*
Inline
[
id,date,amount
8188,'1/10/2022',37.23
8189,'1/14/2022',17.17
8190,'1/20/2022',88.27
8191,'1/22/2022',57.42
8192,'2/1/2022',53.80
8193,'2/2/2022',82.06
8194,'2/20/2022',40.39
8195,'4/11/2022',87.21
8196,'4/13/2022',95.93
8197,'4/15/2022',45.89
8198,'4/25/2022',36.23
8199,'5/20/2022',25.66
8200,'5/22/2022',82.77
8201,'6/19/2022',69.98
8202,'6/22/2022',76.11
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_month`

Esempi di funzioni

data	in_month
1/10/2022	0
1/14/2022	0
1/20/2022	0

data	in_month
1/22/2022	0
2/1/2022	0
2/2/2022	0
2/20/2022	0
4/11/2022	-1
4/13/2022	-1
4/15/2022	-1
4/25/2022	-1
5/20/2022	0
5/22/2022	0
6/19/2022	0
6/22/2022	0

Il campo '`in_month`' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `inmonth()` e trasferendo il campo della data, una data con hard coding per il primo aprile `base_date` e un `period_no` di `0` come argomenti della funzione.

`base_date` identifica il mese che restituirà un risultato booleano TRUE. Pertanto, tutte le transazioni avvenute ad aprile restituiscono il valore TRUE, che viene convalidato nella tabella dei risultati.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, bisogna creare un campo, "`2_months_prior`", che determina se le transazioni sono state effettuate due mesi prima di aprile.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
   *,
   inmonth(date,'04/01/2022', -2) as [2_months_prior]
Inline
[
id,date,amount
8188,'1/10/2022',37.23
8189,'1/14/2022',17.17
```

```
8190,'1/20/2022',88.27
8191,'1/22/2022',57.42
8192,'2/1/2022',53.80
8193,'2/2/2022',82.06
8194,'2/20/2022',40.39
8195,'4/11/2022',87.21
8196,'4/13/2022',95.93
8197,'4/15/2022',45.89
8198,'4/25/2022',36.23
8199,'5/20/2022',25.66
8200,'5/22/2022',82.77
8201,'6/19/2022',69.98
8202,'6/22/2022',76.11
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `2_months_prior`

Esempi di funzioni

data	2_months_prior
1/10/2022	0
1/14/2022	0
1/20/2022	0
1/22/2022	0
2/1/2022	-1
2/2/2022	-1
2/20/2022	-1
4/11/2022	0
4/13/2022	0
4/15/2022	0
4/25/2022	0
5/20/2022	0
5/22/2022	0
6/19/2022	0
6/22/2022	0

L'utilizzo di -2 come argomento `period_no` nella funzione `inmonth()` sposta il mese definito dall'argomento `base_date` due mesi prima. In questo esempio, il mese definito viene cambiato da aprile a febbraio.

Pertanto, qualsiasi transazione avvenuta a febbraio restituirà un risultato booleano TRUE.

Esempio 3 - Oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che determina se le transazioni sono avvenute tra ad aprile viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'1/10/2022',37.23
8189,'1/14/2022',17.17
8190,'1/20/2022',88.27
8191,'1/22/2022',57.42
8192,'2/1/2022',53.80
8193,'2/2/2022',82.06
8194,'2/20/2022',40.39
8195,'4/11/2022',87.21
8196,'4/13/2022',95.93
8197,'4/15/2022',45.89
8198,'4/25/2022',36.23
8199,'5/20/2022',25.66
8200,'5/22/2022',82.77
8201,'6/19/2022',69.98
8202,'6/22/2022',76.11
];
```

Oggetto grafico

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

```
date
```

Per calcolare se una transazione è avvenuta ad aprile, creare la seguente misura:

```
=inmonth(date,'04/01/2022', 0)
```

Risultati

	Esempi di funzioni
data	**=inmonth(date,'04/01/2022', 0)**
1/10/2022	0
1/14/2022	0
1/20/2022	0
1/22/2022	0
2/1/2022	0
2/2/2022	0
2/20/2022	0
4/11/2022	-1
4/13/2022	-1
4/15/2022	-1
4/25/2022	-1
5/20/2022	0
5/22/2022	0
6/19/2022	0
6/22/2022	0

Esempio 4 – Scenario

Script di caricamento e risultati

Panoramica

In questo esempio, un set di dati è caricato in una tabella denominata "`Products`". La tabella contiene i seguenti campi:

- Product ID
- Manufacture date
- Cost price

A causa di un errore dell'attrezzatura, i prodotti fabbricati nel mese di luglio 2022 erano difettosi. Il problema è stato risolto il 27 luglio 2022.

L'utente finale desidera un grafico che visualizzi, per mese, lo stato dei prodotti fabbricati "difettosi" (valore booleano TRUE) o "senza difetti" (valore booleano FALSE), oltre al il costo dei prodotti fabbricati quel mese.

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,manufacture_date,cost_price
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

```
=monthname(manufacture_date)
```

Creare le seguenti misure:

- `=sum(cost_price)`
- `=if(only(inmonth(manufacture_date,makedate(2022,07,01),0)),'Defective','Faultless')`

1. Impostare la misura **Number Formatting** su **Money**.
2. In **Aspetto**, disattivare **Totals**.

Tabella dei risultati

monthname (manufacture_date)	=if(only(inmonth(manufacture_date,makedate (2022,07,01),0)),'Defective','Faultless')	=sum(cost_ price)
Gen 2022	Non difettoso	$54.40
Feb 2022	Non difettoso	$145.69
Mar 2022	Non difettoso	$53.80

monthname (manufacture_date)	=if(only(inmonth(manufacture_date,makedate (2022,07,01),0)),'Defective','Faultless')	=sum(cost_ price)
Apr 2022	Non difettoso	$82.06
May 2022	Non difettoso	$127.60
Jun 2022	Non difettoso	$141.82
Jul 2022	Difettoso	$214.64
Aug 2022	Non difettoso	$147.46
Sep 2022	Non difettoso	$84.21
Oct 2022	Non difettoso	$163.91

La funzione `inmonth()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per qualsiasi prodotto fabbricato nel mese di luglio 2022, la funzione `inmonth()` restituisce un valore booleano TRUE e contrassegna i prodotti come "difettosi". I prodotti che restituiscono un valore FALSE, e che quindi non sono stati fabbricati a luglio, vengono contrassegnati come "non difettosi".

inmonths

Questa funzione consente di verificare se un indicatore temporale cade nello stesso mese, bimestre, trimestre, quadrimestre o semestre di una data base. È inoltre possibile stabilire se l'indicatore temporale ricade all'interno di un periodo di tempo precedente o successivo.

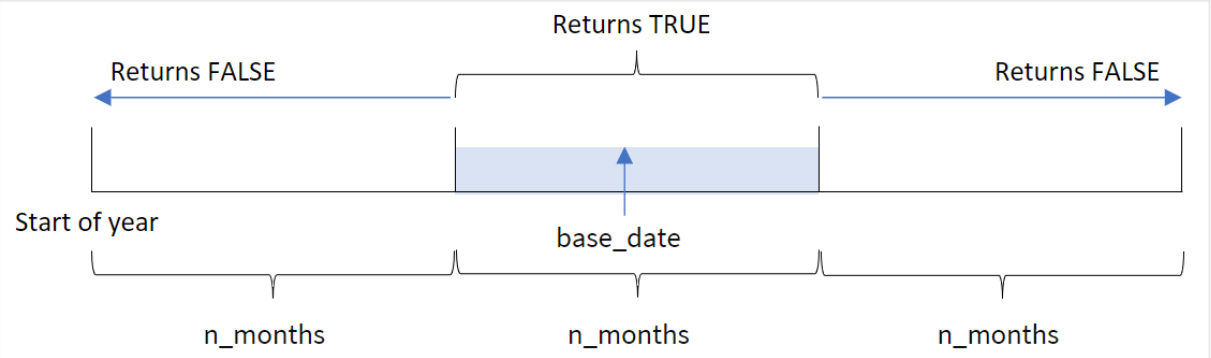
Sintassi:

```
InMonths(n_months, timestamp, base_date, period_no [, first_month_of_year])
```

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Schema della funzione `inmonths()`.



La funzione `inmonths()` divide l'anno in segmenti in base all'argomento `n_months` fornito. Quindi determina se ogni timestamp valutato rientra nello stesso segmento dell'argomento `base_date`. Se, invece, viene fornito un argomento `period_no`, la funzione determina se i timestamp cadono in un periodo precedente o successivo al periodo `base_date`.

I seguenti segmenti dell'anno sono disponibili nella funzione come argomenti `n_month`.

Argomenti `n_month`

Periodo	Numero di mesi
mese	1
bimestre	2
trimestre	3
quattro mesi	4
semestre	6

Casi di utilizzo

La funzione `inmonths()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. Utilizzando la funzione `inmonths()`, è possibile selezionare il periodo che si desidera valutare. Ad esempio, l'utente può identificare i prodotti fabbricati nel mese, trimestre o semestre di un determinato periodo.

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Argomenti

Argomento	Descrizione
n_months	Il numero di mesi che definisce il periodo. Un numero intero o un'espressione la cui risoluzione è un numero intero corrispondente a: 1 (equivalente alla funzione inmonth ()), 2 (bimestre), 3 (equivalente alla funzione inquarter()), 4 (quadrimestre) o 6 (semestre).
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare il periodo.
period_no	Il periodo può essere differito mediante **period_no**, un numero intero, o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica il periodo che contiene **base_date**. I valori negativi di **period_no** indicano i periodi precedenti, mentre i valori positivi indicano i periodi successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

È possibile utilizzare i seguenti valori per impostare il primo mese dell'anno nell'argomento `first_month_of_year`:

valori first_month_of_year

Month	Valore
Febbraio	2
March	3
April	4
May	5
June	6
July	7
August	8
September	9
October	10
Novembre	11
December	12

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`inmonths(4, '01/25/2013', '04/25/2013', 0)`	Restituisce TRUE. Perché il valore del timestamp, 25/01/2013, si trova nel periodo di quattro mesi dal 01/01/2013 al 30/04/2013, in cui si trova il valore di `base_date`, resta 25/04/2013.
`inmonths(4, '05/25/2013', '04/25/2013', 0)`	Restituisce FALSE. Perché il 25/05/2013 non rientra nello stesso periodo dell'esempio precedente.
`inmonths(4, '11/25/2012', '02/01/2013', -1)`	Restituisce TRUE. Poiché il valore di `period_no`, `-1`, sposta il periodo di ricerca indietro di un periodo di quattro mesi (il valore di n-mesi), il che rende il periodo di ricerca dal 01/09/2012 al 31/12/2012.

Esempio	Risultato
`inmonths(4, '05/25/2006', '03/01/2006', 0, 3)`	Restituisce TRUE. Perché il valore di `first_month_of_year` è impostato su `3`, che rende il periodo di ricerca dal 01/03/2006 al 30/07/2006 invece che dal 01/01/2006 al 30/04/2006.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- Un caricamento precedente con una variabile aggiuntiva, '`in_months`', che determina quali transazioni hanno avuto luogo nello stesso trimestre del 15 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       inmonths(3,date,'05/15/2022', 0) as in_months
       ;
Load
*
Inline
[
id,date,amount
8188,'2/19/2022',37.23
8189,'3/7/2022',17.17
8190,'3/30/2022',88.27
8191,'4/5/2022',57.42
8192,'4/16/2022',53.80
8193,'5/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/22/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
```

```
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

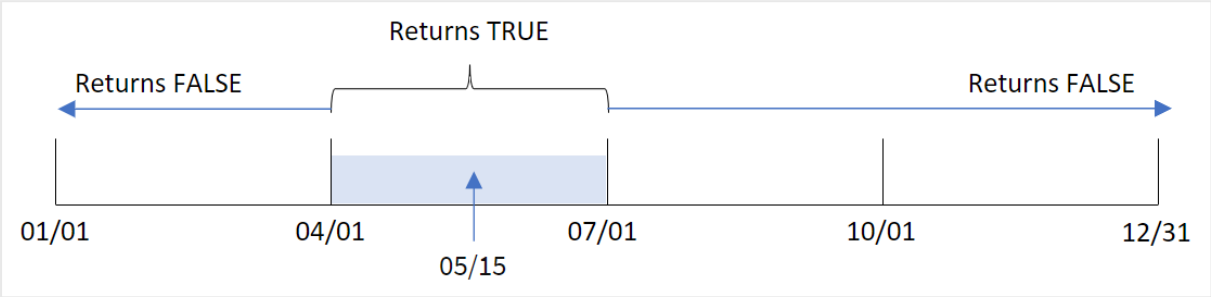
- `date`
- `in_months`

Tabella dei risultati

date	in_months
2/19/2022	0
3/7/2022	0
3/30/2022	0
4/5/2022	-1
4/16/2022	-1
5/1/2022	-1
5/7/2022	-1
5/22/2022	-1
6/15/2022	-1
6/26/2022	-1
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Il campo '`in_months`' viene creato nell'istruzione LOAD precedente utilizzando la funzione `inmonths()`. Il primo argomento fornito è `3`, che divide l'anno in segmenti di trimestre. Il secondo argomento identifica il campo da valutare, in questo esempio il campo data. Il terzo argomento è una data con hard coding per il 15 maggio, che è `base_date` e `period_no` di `0` è l'argomento finale.

Schema della funzione `inmonths()` con segmenti trimestrali



Maggio rientra nel secondo trimestre dell'anno. Pertanto, qualsiasi transazione effettuata tra il 1° aprile e il 30 giugno restituirà un risultato booleano pari a TRUE. Questo viene convalidato nella tabella dei risultati.

Esempio 2 - period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- Un caricamento precedente con una variabile aggiuntiva, '`previous_quarter`', che determina se le transazioni hanno avuto luogo nel trimestre precedente al 15 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       inmonths(3,date,'05/15/2022', -1) as previous_quarter
       ;
Load
*
Inline
[
id,date,amount
8188,'2/19/2022',37.23
8189,'3/7/2022',17.17
8190,'3/30/2022',88.27
```

```
8191,'4/5/2022',57.42
8192,'4/16/2022',53.80
8193,'5/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/22/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_quarter`

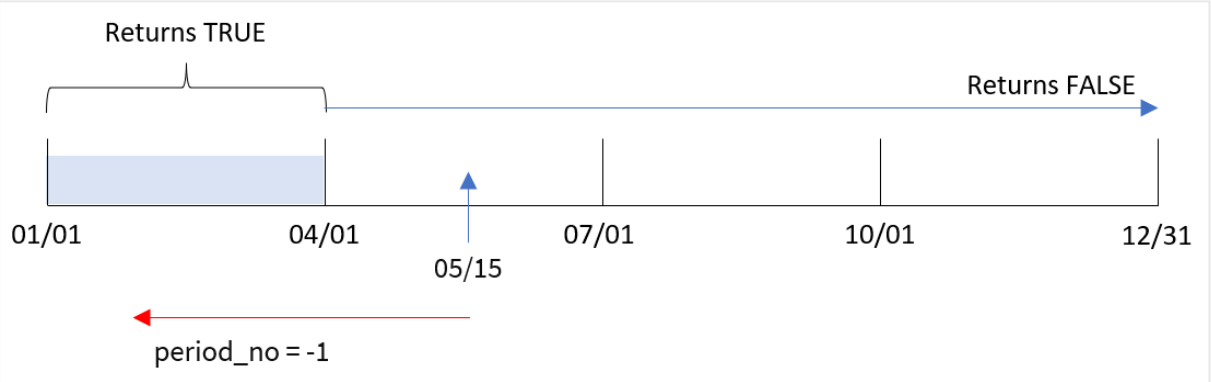
Tabella dei risultati

date	trimestre precedente
2/19/2022	-1
3/7/2022	-1
3/30/2022	-1
4/5/2022	0
4/16/2022	0
5/1/2022	0
5/7/2022	0
5/22/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0

date	trimestre precedente
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

La funzione valuta se le transazioni si sono verificate nel primo trimestre dell'anno utilizzando `-1` come argomento `period_no` della funzione `inmonths()`. Il 15 maggio è il `base_date` e ricade nel secondo trimestre dell'anno (aprile-giugno).

Schema della funzione `inmonths()` con segmenti trimestrali e `period_no` impostato su `-1`



Pertanto, qualsiasi transazione che si verifichi tra gennaio e marzo restituirà un risultato booleano di TRUE.

Esempio 3 - first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- Un caricamento precedente con una variabile aggiuntiva, '`in_months`', che determina quali transazioni hanno avuto luogo nello stesso trimestre del 15 maggio 2022.

In questo esempio, il criterio organizzativo prevede che marzo sia il primo mese dell'anno finanziario.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       inmonths(3,date,'05/15/2022', 0, 3) as in_months
       ;
Load
*
Inline
[
id,date,amount
8188,'2/19/2022',37.23
8189,'3/7/2022',17.17
8190,'3/30/2022',88.27
8191,'4/5/2022',57.42
8192,'4/16/2022',53.80
8193,'5/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/22/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_months`

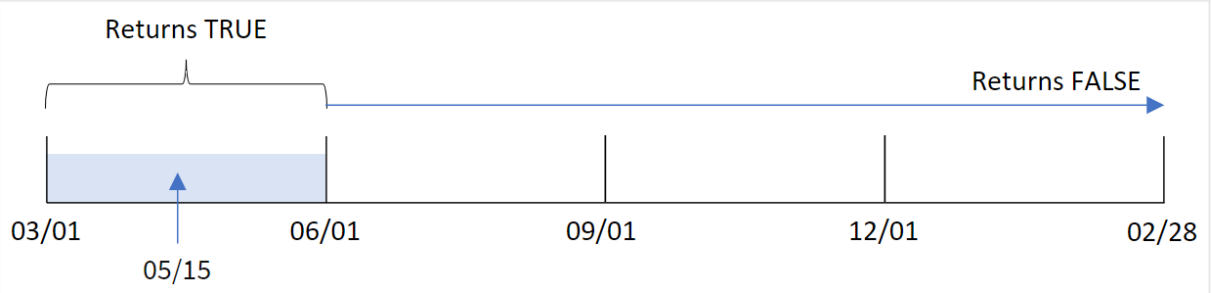
Tabella dei risultati

date	in_months
2/19/2022	0
3/7/2022	-1
3/30/2022	-1
4/5/2022	-1

date	in_months
4/16/2022	-1
5/1/2022	-1
5/7/2022	-1
5/22/2022	-1
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Utilizzando `3` come argomento `first_month_of_year` nella funzione `inmonths()`, la funzione inizia l'anno il 1° marzo. La funzione `inmonths()` divide quindi l'anno in trimestri: Mar-Mag, Giu-Ago, Set-Nov, Dic-Feb. Pertanto, il 15 maggio cade nel primo trimestre dell'anno (marzo-maggio).

Schema della funzione `inmonths()` con marzo impostato come primo mese dell'anno



Qualsiasi transazione che si verifica in questi mesi restituirà un risultato booleano pari a TRUE.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che determina se le transazioni sono avvenute nello stesso trimestre del 15 maggio 2022 viene creato come misura in un grafico dell'app.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'2/19/2022',37.23
8189,'3/7/2022',17.17
8190,'3/30/2022',88.27
8191,'4/5/2022',57.42
8192,'4/16/2022',53.80
8193,'5/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/22/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

- `date`

Per calcolare se le transazioni sono avvenute nello stesso trimestre del 15 maggio, creare la seguente misura:

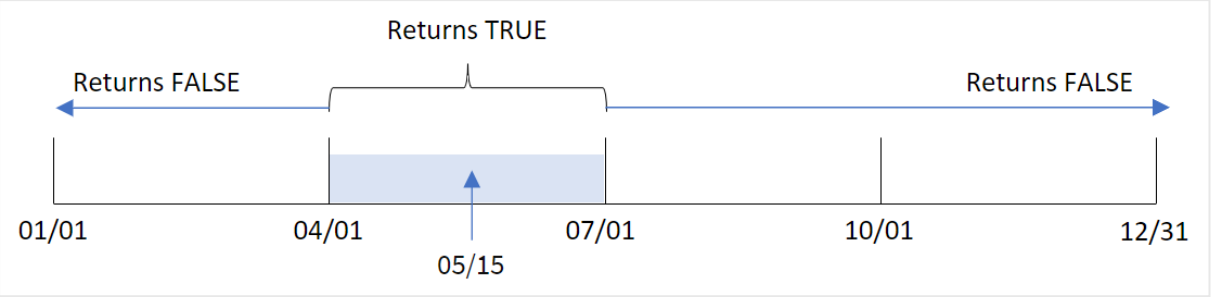
```
=inmonths(3,date,'05/15/2022', 0)
```

Tabella dei risultati

date	=inmonths(3,date,'05/15/2022', 0)
2/19/2022	0
3/7/2022	0
3/30/2022	0
4/5/2022	-1
4/16/2022	-1
5/1/2022	-1
5/7/2022	-1
5/22/2022	-1
6/15/2022	-1
6/26/2022	-1
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Il campo '`in_months`' viene creato nel grafico mediante l'utilizzo della funzione `inmonths()`. Il primo argomento fornito è `3`, che divide l'anno in segmenti di trimestre. Il secondo argomento identifica il campo da valutare, in questo esempio il campo data. Il terzo argomento è una data con hard coding per il 15 maggio, che è `base_date` e `period_no` di `0` è l'argomento finale.

Schema della funzione `inmonths()` con segmenti trimestrali



Maggio rientra nel secondo trimestre dell'anno. Pertanto, qualsiasi transazione effettuata tra il 1° aprile e il 30 giugno restituirà un risultato booleano pari a TRUE. Questo viene convalidato nella tabella dei risultati.

Esempio 5 - Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata '`Products`'.
- La tabella contiene i seguenti campi:
  - ID prodotto
  - tipo di prodotto
  - data di produzione
  - prezzo di costo

L'utente finale desidera un grafico che visualizzi, per tipologia di prodotto, il costo dei prodotti fabbricati nel primo segmento del 2021. L'utente vorrebbe poter definire la lunghezza di questo segmento.

Script di caricamento

```
SET vPeriod = 1;

Products:
Load
*
Inline
[
product_id,product_type,manufacture_date,cost_price
8188,product A,'2/19/2022',37.23
8189,product D,'3/7/2022',17.17
8190,product C,'3/30/2022',88.27
8191,product B,'4/5/2022',57.42
8192,product D,'4/16/2022',53.80
8193,product D,'5/1/2022',82.06
8194,product A,'5/7/2022',40.39
```

```
8195,product B,'5/22/2022',87.21
8196,product C,'6/15/2022',95.93
8197,product B,'6/26/2022',45.89
8198,product C,'7/9/2022',36.23
8199,product D,'7/22/2022',25.66
8200,product D,'7/23/2022',82.77
8201,product A,'7/27/2022',69.98
8202,product A,'8/2/2022',76.11
8203,product B,'8/8/2022',25.12
8204,product B,'8/19/2022',46.23
8205,product B,'9/26/2022',84.21
8206,product C,'10/14/2022',96.24
8207,product D,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio.

All'inizio dello script di caricamento, viene creata una variabile `vPeriod`, legata al controllo dell'input variabile.

Procedere come segue:

1. Nel pannello delle risorse, fare clic **su Oggetti personalizzati**.
2. Selezionare **Qlik Dashboard bundle** e creare un oggetto **Input variabile**.
3. Immettere un titolo per l'oggetto grafico.
4. In **Variabile**, selezionare **vPeriod** come nome e impostare l'oggetto in modo che venga visualizzato come **Elenco a discesa**.
5. Sotto **Valori**, fare clic sui valori **Dinamici**. Inserire quanto segue:
   `='1~month|2~bi-month|3~quarter|4~tertial|6~half-year'.`
6. Aggiungere una nuova tabella al foglio.
7. Sotto **Dati** nel pannello proprietà, aggiungere `product_type` come dimensione.
8. Aggiungere l'espressione seguente come misura:
   `=sum(if(inmonths($(vPeriod),manufacture_date,makedate(2022,01,01),0),cost_price,0))`
9. Impostare la misura **Formattazione numero** su **Denaro**.

Tabella dei risultati

product_type	=sum(if(inmonths($(vPeriod),manufacture_date,makedate(2022,01,01),0),cost_price,0))
prodotto A	$88.27
prodotto B	$37.23
prodotto C	$17.17
prodotto D	$0.00

La funzione `inmonths()` utilizza l'input dell'utente come argomento per definire la dimensione del segmento iniziale dell'anno. La funzione inserisce la data di produzione di ciascun prodotto come secondo argomento della funzione `inmonths()`. Utilizzando il 1° gennaio come terzo argomento della funzione `inmonths()`, i prodotti con date di produzione che cadono nel segmento iniziale dell'anno restituiranno un valore booleano di TRUE e quindi la funzione somma aggiungerà i costi di tali prodotti.

inmonthstodate

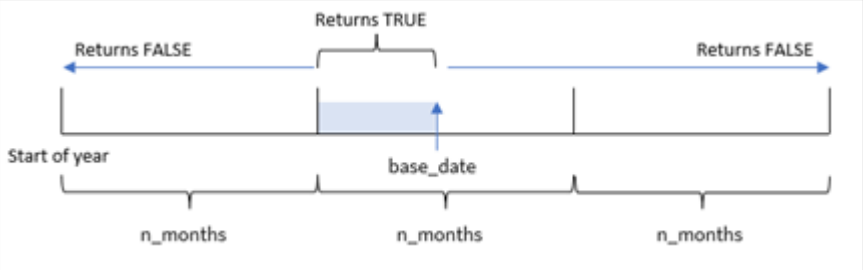
Questa funzione stabilisce se un indicatore temporale ricade all'interno della parte di un periodo di un mese, bimestre, trimestre, quadrimestre o semestre fino a includere l'ultimo millisecondo di `base_date`. È inoltre possibile stabilire se l'indicatore temporale ricade all'interno di un periodo di tempo precedente o successivo.

Sintassi:

```
InMonths (n_months, timestamp, base_date, period_no[, first_month_of_year ])
```

Tipo di dati restituiti: Booleano

Schema della funzione `inmonthstodate`.



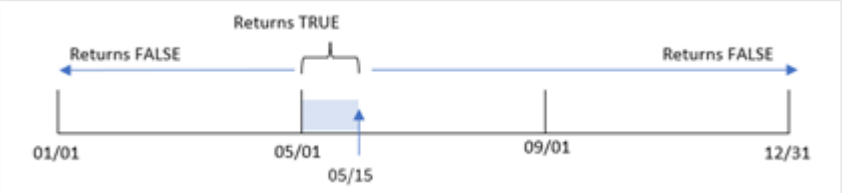
Argomenti

Argomento	Descrizione
n_months	Il numero di mesi che definisce il periodo. Un numero intero o un'espressione la cui risoluzione è un numero intero corrispondente a: 1 (equivalente alla funzione inmonth ()), 2 (bimestre), 3 (equivalente alla funzione inquarter()), 4 (quadrimestre) o 6 (semestre).
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare il periodo.
period_no	Il periodo può essere differito mediante **period_no**, un numero intero, o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica il periodo che contiene **base_date**. I valori negativi di **period_no** indicano i periodi precedenti, mentre i valori positivi indicano i periodi successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

Nella funzione `inmonthstodate()`, `base_date` agisce come punto finale del particolare segmento di anno di cui fa parte.

Ad esempio, se l'anno è stato suddiviso in segmenti terziari e la `base_date` era il 15 maggio, qualsiasi timestamp tra l'inizio di gennaio e la fine di aprile restituirà un risultato booleano di FALSE. Le date comprese tra il 1° e il 15 maggio restituirebbero TRUE. Il resto dell'anno restituirebbe FALSE.

Schema dell'intervallo di risultati booleani della funzione `inmonthstodate`.



I seguenti segmenti dell'anno sono disponibili nella funzione come argomenti `n_month`.

Argomenti `n_month`

Periodo	Numero di mesi
mese	1
bimestre	2
trimestre	3
quadrimestre	4
semestre	6

Casi di utilizzo

La funzione `inmonthstodate()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. Utilizzando la funzione `inmonthstodate()`, è possibile selezionare il periodo che si desidera valutare. Ad esempio, fornendo una variabile di input che consenta all'utente di identificare i prodotti fabbricati nel mese, trimestre o semestre di un periodo, fino a una certa data.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è

impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`inmonthstodate(4, '01/25/2013', '04/25/2013', 0)`	Restituisce il valore di True, perché il valore di timestamp, 01/25/2013 ricade entro il periodo di quattro mesi di 01/01/2013 fino alla fine di 04/25/2013, in cui ricade il valore di base_date, 04/25/2013.
`inmonthstodate(4, '04/26/2013', '04/25/2006', 0)`	Restituisce False, perché 04/26/2013 non ricade nello stesso periodo dell'esempio precedente.
`inmonthstodate(4, '09/25/2005', '02/01/2006', -1)`	Restituisce True, perché il valore di period_no, -1, fa slittare il periodo della ricerca indietro di un periodo di quattro mesi (il valore di n-months) e questo cambia il periodo di ricerca da 01/09/2005 a 02/01/2006.
`inmonthstodate(4, '04/25/2006', '06/01/2006', 0, 3)`	Restituisce True, perché il valore di first_month_of_year è impostato su 3 e ciò cambia il periodo di ricerca da 03/01/2006 a 06/01/2006 anziché da 05/01/2006 in 06/01/2006.

Esempio 1 – Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- Un campo data nel formato `(MM/DD/YYYY)` della variabile di sistema `DateFormat`.
- Un'istruzione LOAD precedente contenente:
    - La funzione `inmonthstodate()` impostata come campo, '`in_months_to_date`'. Questo determina quali transazioni sono state effettuate nel trimestre fino al 15 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    inmonthstodate(3,date,'05/15/2022', 0) as in_months_to_date
    ;
Load
*
```

```
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_months_to_date`

Tabella dei risultati

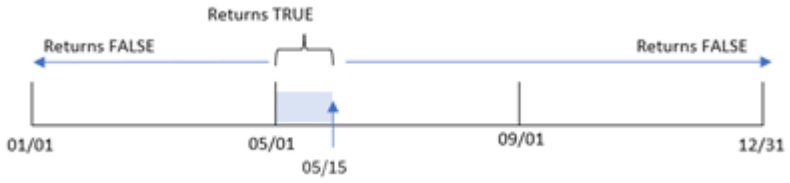
date	in_months_to_date
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	0
4/1/2022	-1
5/7/2022	-1
5/16/2022	0
6/15/2022	0
6/26/2022	0

date	in_months_to_date
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Il campo '`in_months_to_date`' viene creato nell'istruzione LOAD precedente utilizzando la funzione `inmonthstodate()`.

Il primo argomento fornito è `3`, che divide l'anno in segmenti di trimestre. Il secondo argomento identifica il campo da valutare. Il terzo argomento è una data codificata per il 15 maggio, ovvero il `base_date` che definisce il confine finale del segmento. Un `period_no` di `0` rappresenta l'argomento finale.

Schema della funzione `inmonthstodate` senza argomenti aggiuntivi.



Qualsiasi transazione avvenuta tra il 1° aprile e il 15 maggio restituisce il risultato booleano di TRUE. Le date di transazione al di fuori di tale periodo restituiscono FALSE.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il compito è quello di creare un campo, '`previous_qtr_to_date`' che determini se le transazioni hanno avuto luogo un trimestre prima del 15 maggio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    inmonthstodate(3,date,'05/15/2022', -1) as previous_qtr_to_date
    ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_qtr_to_date`

Tabella dei risultati

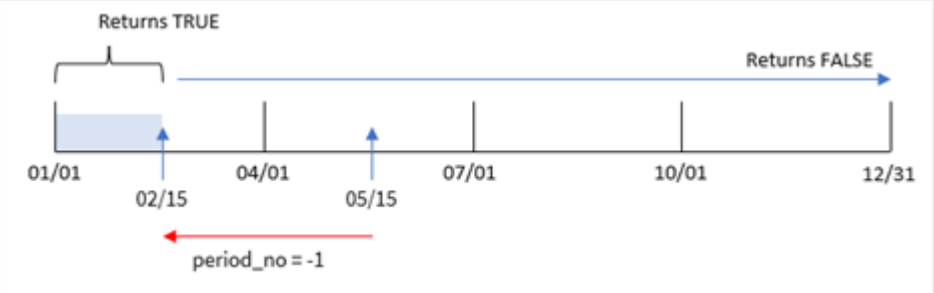
data	previous_qtr_to_date
1/7/2022	-1
1/19/2022	-1
2/5/2022	-1
2/28/2022	0

data	previous_qtr_to_date
3/16/2022	0
4/1/2022	0
5/7/2022	0
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Utilizzando `-1` come argomento `period_no` della funzione `inmonthstodate()`, la funzione sposta di un trimestre i confini del segmento dell'anno di confronto.

Il 15 maggio rientra nel secondo trimestre dell'anno, pertanto il segmento inizialmente corrisponde al periodo compreso tra il 1° aprile e il 15 maggio. L'argomento `period_no` compensa questo segmento con un valore negativo di tre mesi. I limiti di data diventano dal 1° gennaio al 15 febbraio.

Schema della funzione `inmonthstodate` con un valore period_no impostato a -1.



Pertanto, qualsiasi transazione che si verifichi tra il 1° gennaio e il 15 febbraio restituirà il risultato booleano di TRUE.

Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, il criterio organizzativo prevede che marzo sia il primo mese dell'anno finanziario.

Crea un campo, 'in_months_to_date', che determina quali transazioni sono avvenute nello stesso trimestre fino al 15 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    inmonthstodate(3,date,'05/15/2022', 0,3) as in_months_to_date
    ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- date
- in_months_to_date

Tabella dei risultati

data	previous_qtr_to_date
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	-1
4/1/2022	-1
5/7/2022	-1
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Utilizzando `3` come argomento `first_month_of_year` nella funzione `inmonthstodate()`, questa inizia l'anno il 1° marzo e poi divide l'anno in trimestri in base al primo argomento fornito. Pertanto, i segmenti del trimestre sono:

- Mag-Mag
- Giu-Ago
- Set-Nov
- Dic-Feb

Il `base_date` del 15 maggio segmenta quindi il trimestre da marzo a maggio impostando il limite finale al 15 maggio.

Schema della funzione `inmonthstodate` con marzo impostato come primo mese dell'anno.

Pertanto, qualsiasi transazione che si verifichi tra il 1° marzo e il 15 maggio restituirà un risultato booleano di TRUE, mentre le transazioni con date al di fuori di questi confini restituiranno un valore di FALSE.

Esempio 4 - Esempio grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, il set di dati è invariato e viene caricato nell'app. Il compito è quello di creare un calcolo che determini se le transazioni sono avvenute nello stesso trimestre del 15 maggio come misura in un grafico dell'app.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
```

```
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

```
date
```

Per calcolare se le transazioni sono avvenute nello stesso trimestre del 15 maggio, creare la seguente misura:

```
=inmonthstodate(3,date,'05/15/2022', 0)
```

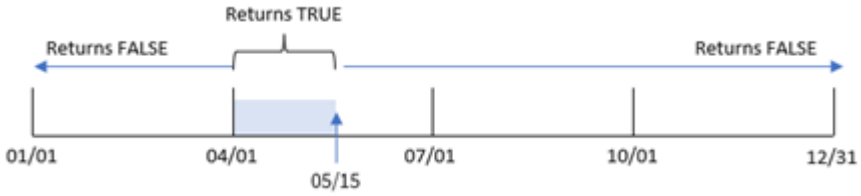
Tabella dei risultati

date	=inmonthstodate(3,date,'05/15/2022', 0)
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	0
4/1/2022	-1
5/7/2022	-1
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

La misura '`in_months_to_date`' viene creata nell'oggetto grafico utilizzando la funzione `inmonthstodate()`.

Il primo argomento fornito è `3`, che divide l'anno in segmenti di trimestre. Il secondo argomento identifica il campo da valutare. Il terzo argomento è una data codificata per il 15 maggio, ovvero il `base_date`, che definisce il confine finale del segmento. Un `period_no` di `0` rappresenta l'argomento finale.

Schema della funzione `inmonthstodate` con segmenti trimestrali.



Qualsiasi transazione avvenuta tra il 1° aprile e il 15 maggio restituirà il risultato booleano di TRUE. Le date di transazione al di fuori di tale segmento restituiscono FALSE.

Esempio 5 – Scenario

Script di caricamento e risultati

Panoramica

In questo esempio, un set di dati è caricato in una tabella denominata "`Sales`". La tabella contiene i seguenti campi:

- Product ID
- Tipo di prodotto
- Data di vendita
- Prezzo di vendita

L'utente finale desidera un grafico che visualizzi, per tipo di prodotto, le vendite dei prodotti venduti nel periodo precedente il 24 dicembre 2022. L'utente vorrebbe poter definire la lunghezza di questo periodo.

Script di caricamento

```
SET vPeriod = 1;

Products:
Load
*
Inline
[
product_id,product_type,sales_date,sales_price
8188,product A,'9/19/2022',37.23
8189,product D,'10/27/2022',17.17
8190,product C,'10/30/2022',88.27
8191,product B,'10/31/2022',57.42
8192,product D,'11/16/2022',53.80
8193,product D,'11/28/2022',82.06
8194,product A,'12/2/2022',40.39
8195,product B,'12/5/2022',87.21
```

```
8196,product C,'12/15/2022',95.93
8197,product B,'12/16/2022',45.89
8198,product C,'12/19/2022',36.23
8199,product D,'12/22/2022',25.66
8200,product D,'12/23/2022',82.77
8201,product A,'12/24/2022',69.98
8202,product A,'12/24/2022',76.11
8203,product B,'12/26/2022',25.12
8204,product B,'12/27/2022',46.23
8205,product B,'12/27/2022',84.21
8206,product C,'12/28/2022',96.24
8207,product D,'12/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio.

All'inizio dello script di caricamento, viene creata una variabile `vPeriod`, legata al controllo dell'input variabile.

Procedere come segue:

1. Nel pannello delle risorse, fare clic su **Oggetti personalizzati**.
2. Selezionare **Qlik Dashboard bundle** e aggiungere un **Input variabile** al proprio foglio.
3. Inserire un titolo per il grafico.
4. In **Variabile**, selezionare **vPeriod** come nome e impostare l'oggetto in modo che venga visualizzato come **Elenco a discesa**.
5. Sotto **Valori**, fare clic sui valori **Dinamici**. Inserire quanto segue:
   `='1~month|2~bi-month|3~quarter|4~tertial|6~half-year'.`
6. Aggiungere una nuova tabella al foglio.
7. Sotto **Dati** nel pannello proprietà, aggiungere `product_type` come dimensione.
8. Aggiungere l'espressione seguente come misura:
   `=sum(if(inmonthstodate($(vPeriod),sales_date,makedate(2022,12,24),0),sales_price,0))`
9. Impostare la misura **Formattazione numero** su **Denaro**.

Tabella dei risultati

product_type	=sum(if(inmonthstodate($(vPeriod),sales_date,makedate(2022,12,24),0),sales_price,0))
prodotto A	$186.48
prodotto B	$190.52
prodotto C	$220.43
prodotto D	$261.46

La funzione `inmonthstodate()` utilizza l'input dell'utente come argomento per definire la dimensione del segmento iniziale dell'anno.

La funzione inserisce la data di vendita di ciascun prodotto come secondo argomento della funzione `inmonthstodate()`. Utilizzando il 24 dicembre come terzo argomento della funzione `inmonthstodate()`, i prodotti con date di vendita che si verificano nel periodo definito fino al 24 dicembre incluso restituiscono un valore booleano di TRUE. La funzione somma aggiunge le vendite di questi prodotti.

inmonthtodate

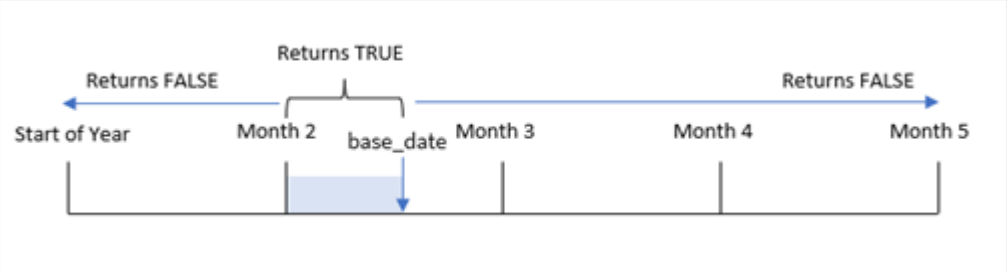
Restituisce True se **date** ricade nella parte di mese contenente **basedate** fino a includere l'ultimo millisecondo di **basedate**.

Sintassi:

```
InMonthToDate (timestamp, base_date, period_no)
```

Tipo di dati restituiti: Booleano

Schema della funzione `inmonthtodate`.



La funzione `inmonthtodate()` identifica un mese selezionato come un segmento. Il limite di partenza è l'inizio del mese. Il limite finale può essere impostato come una data successiva del mese. Determina quindi se un insieme di date rientra o meno in questo segmento, restituendo un valore booleano TRUE o FALSE.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare il mese.
period_no	Il mese può essere differito mediante **period_no**. **period_no** è un numero intero, in cui il valore 0 indica il mese che contiene **base_date**. I valori negativi di **period_no** indicano i mesi precedenti, mentre i valori positivi indicano i mesi successivi.

Casi di utilizzo

La funzione `inmonthtodate()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. La funzione `inmonthtodate()` restituisce un'aggregazione o un calcolo che dipende dal fatto che una data valutata si sia verificata in un determinato mese fino a includere la data in questione.

Ad esempio, la funzione `inmonthtodate()` può essere utilizzata per identificare tutte le apparecchiature prodotte in un mese fino a una data specifica.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`inmonthtodate ('01/25/2013', '25/01/2013', 0)`	Restituisce True
`inmonthtodate ('01/25/2013', '24/01/2013', 0)`	Restituisce False
`inmonthtodate ('01/25/2013', '28/02/2013', -1)`	Restituisce True

Esempio 1 – Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- Un campo data viene fornito nel formato (`MM/DD/YYYY`) della variabile di sistema `DateFormat`.
- Un'istruzione LOAD precedente contenente:
    - La funzione `inmonthtodate()` impostata come campo, '`in_month_to_date`'. Ciò determina quali transazioni sono state effettuate tra il 1° luglio e il 26 luglio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
```

```
    inmonthtodate(date,'07/26/2022', 0) as in_month_to_date
    ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_month_to_date`

Tabella dei risultati

date	in_month_to_date
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	0
4/1/2022	0
5/7/2022	0
5/16/2022	0

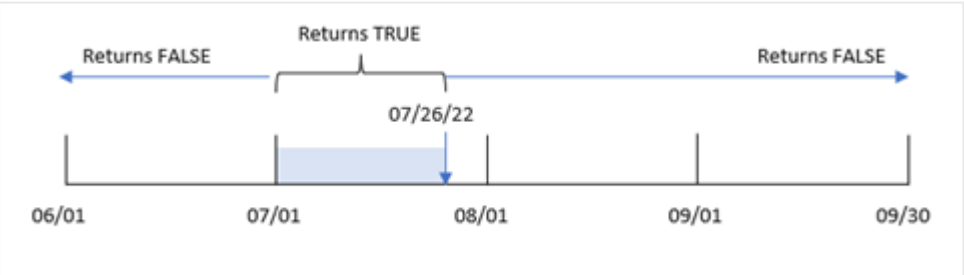
date	in_month_to_date
6/15/2022	0
6/26/2022	0
7/9/2022	-1
7/22/2022	-1
7/23/2022	-1
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Il campo '`in_month_to_date`' viene creato nell'istruzione LOAD precedente utilizzando la funzione `inmonthtodate()`.

Il primo argomento identifica il campo da valutare. Il secondo argomento è una data codificata, il 26 luglio, che è il `base_date`. Questo argomento `base_date` identifica quale mese è segmentato e il limite finale di tale segmento.

Un `period_no` di `0` è l'argomento finale, a significare che la funzione non confronta i mesi precedenti o successivi al mese segmentato.

Schema della funzione `inmonthtodate` senza argomenti aggiuntivi.



Di conseguenza, qualsiasi transazione effettuata tra il 1° e il 26 luglio restituisce un risultato booleano di TRUE. Qualsiasi transazione che si verifichi nel mese di luglio dopo il 26 luglio restituisce un risultato booleano di FALSE, così come qualsiasi transazione in qualsiasi altro mese dell'anno.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, il compito è quello di creare un campo, '`six_months_prior`', che determini quali transazioni hanno avuto luogo sei mesi prima del 1° luglio e del 26 luglio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    inmonthtodate(date,'07/26/2022', -6) as six_months_prior
    ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

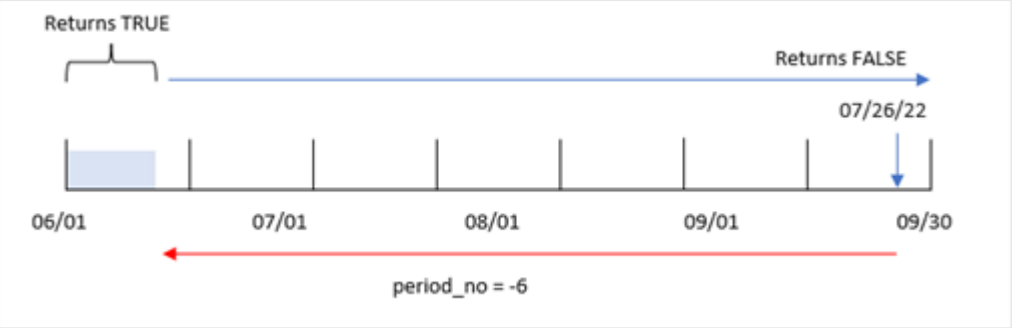
- `date`
- `six_months_prior`

Tabella dei risultati

date	six_months_prior
1/7/2022	-1
1/19/2022	-1
2/5/2022	0
2/28/2022	0
3/16/2022	0
4/1/2022	0
5/7/2022	0
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Utilizzando `-6` come argomento `period_no` nella funzione `inmonthtodate()`, i confini del segmento del mese di confronto si spostano di sei mesi. Inizialmente il segmento del mese corrisponde a un periodo compreso tra il 1° e il 26 luglio. `period_no` viene quindi compensato con un valore negativo di sei mesi e i confini della data vengono spostati e cadono tra il 1° e il 26 gennaio.

Schema della funzione `inmonthtodate` con un valore period_no impostato a -6.



Di conseguenza, qualsiasi transazione effettuata tra il 1° e il 26 gennaio restituirà un risultato booleano di TRUE.

Esempio 3 - Esempio di grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, il set di dati è invariato e caricato nell'app. Il compito è quello di creare un calcolo che determini se le transazioni sono avvenute tra il 1° e il 26 luglio come misura in un grafico dell'app.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
```

```
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

```
date
```

Per calcolare se le transazioni sono avvenute tra il 1° luglio e il 26 luglio, creare la seguente misura:

```
=inmonthtodate(date,'07/26/2022', 0)
```

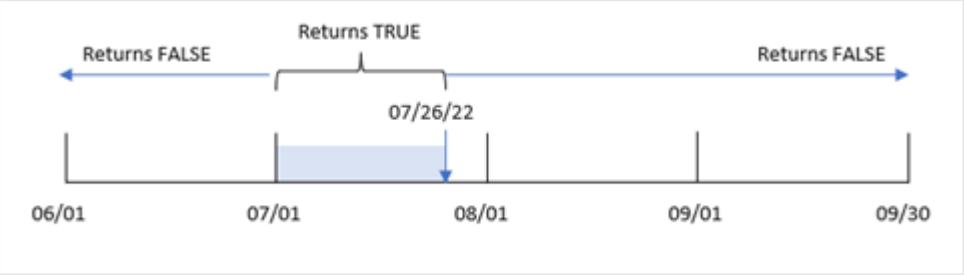
Tabella dei risultati

date	=inmonthtodate(date,'07/26/2022', 0)
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	0
4/1/2022	0
5/7/2022	0
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	-1
7/22/2022	-1
7/23/2022	-1
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

La misura del campo '`in_month_to_date`' viene creata nel grafico mediante l'utilizzo della funzione `inmonthtodate()`.

Il primo argomento identifica il campo da valutare. Il secondo argomento è una data codificata, il 26 luglio, che è il `base_date`. Questo argomento base_date identifica quale mese è segmentato e il limite finale di tale segmento. Un `period_no` di 0 rappresenta l'argomento finale. Ciò significa che la funzione non confronta i mesi precedenti o successivi al mese segmentato.

Schema della funzione `inmonthtodate` senza argomenti aggiuntivi.



Di conseguenza, qualsiasi transazione effettuata tra il 1° e il 26 luglio restituisce un risultato booleano di TRUE. Qualsiasi transazione che si verifichi nel mese di luglio dopo il 26 luglio restituisce un risultato booleano di FALSE, così come qualsiasi transazione in qualsiasi altro mese dell'anno.

Esempio 4 – Scenario

Script di caricamento e risultati

Panoramica

In questo esempio, un set di dati è caricato in una tabella denominata "`Products`". La tabella contiene i seguenti campi:

- Product ID
- Manufacture date
- Cost price

A causa di un errore dell'attrezzatura, i prodotti fabbricati nel mese di luglio 2022 erano difettosi. Il problema è stato risolto il 27 luglio 2022.

L'utente finale desidera un grafico che visualizzi, per mese, lo stato dei prodotti fabbricati "difettosi" (valore booleano TRUE) o "non difettosi" (valore booleano FALSE), oltre al il costo dei prodotti fabbricati quel mese.

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,manufacture_date,cost_price
```

```
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `=monthname(manufacture_date)`
- `=if(Inmonthtodate(manufacture_date,makedate(2022,07,26),0),'Defective','Faultless')`

Per calcolare la somma dei costi dei prodotti, creare questa misura:

```
=sum(cost_price)
```

Impostare la misura **Formattazione numero** su **Denaro**.

Tabella dei risultati

monthname (manufacture_date)	if(Inmonthtodate(manufacture_date,makedate (2022,07,26),0),'Defective','Faultless')	Sum(cost_ price)
Gen 2022	Non difettoso	$54.40
Feb 2022	Non difettoso	$145.69
Mar 2022	Non difettoso	$53.80
Apr 2022	Non difettoso	$82.06
May 2022	Non difettoso	$127.60
Jun 2022	Non difettoso	$141.82
Jul 2022	Difettoso	$144.66
Jul 2022	Non difettoso	$69.98

monthname (manufacture_date)	if(Inmonthtodate(manufacture_date,makedate (2022,07,26),0),'Defective','Faultless')	Sum(cost_ price)
Aug 2022	Non difettoso	$147.46
Sep 2022	Non difettoso	$84.21
Oct 2022	Non difettoso	$163.91

La funzione `inmonthtodate()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto.

Per le date che restituiscono un valore booleano di TRUE, il prodotto è contrassegnato come 'difettoso'. Tutti i prodotti che restituiscono un valore di FALSE, e che quindi non sono stati realizzati nel mese fino al 26 luglio compreso, vengono contrassegnati come 'non difettosi'.

inquarter

Questa funzione restituisce True se **timestamp** ricade all'interno del trimestre contenente **base_date**.

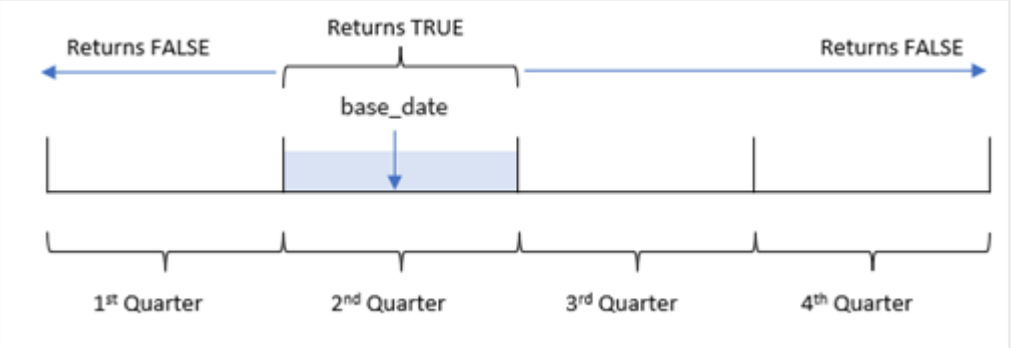
Sintassi:

```
InQuarter (timestamp, base_date, period_no[, first_month_of_year])
```

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Schema dell'intervallo della funzione `inquarter()`



In altre parole, la funzione `inquarter()` divide l'anno in quattro trimestri uguali tra il 1° gennaio e il 31 dicembre. È possibile utilizzare l'argomento `first_month_of_year` per modificare il mese che viene considerato il primo nella propria app e i trimestri cambieranno in base a tale argomento. `base_date`, la funzione identifica quale trimestre deve essere utilizzato come comparatore per la funzione. Infine, la funzione restituisce un risultato booleano quando si confrontano i valori della data con il segmento del trimestre.

Casi di utilizzo

La funzione `inquarter()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. Ciò restituisce un'aggregazione o un calcolo che dipende dalla presenza di una data nel trimestre selezionato.

Ad esempio, la funzione `inquarter()` può essere utilizzata per identificare tutte le apparecchiature prodotte in un segmento trimestrale in base alle date di produzione delle apparecchiature.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare il trimestre.
period_no	Il trimestre può essere differito mediante **period_no**. **period_no** è un numero intero, in cui il valore 0 indica il trimestre che contiene **base_date**. I valori negativi di **period_no** indicano i trimestri precedenti, mentre i valori positivi indicano i trimestri successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

È possibile utilizzare i seguenti valori per impostare il primo mese dell'anno nell'argomento `first_month_of_year`:

valori first_month_of_year

Month	Valore
Febbraio	2
March	3
April	4
May	5
June	6
July	7
August	8
September	9
October	10
Novembre	11
December	12

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
inquarter ('01/25/2013', '01/01/2013', 0)	Restituisce TRUE
inquarter ('01/25/2013', '04/01/2013', 0)	Restituisce FALSE
inquarter ('01/25/2013', '01/01/2013', -1)	Restituisce FALSE
inquarter ('12/25/2012', '01/01/2013', -1)	Restituisce TRUE
inquarter ('01/25/2013', '03/01/2013', 0, 3)	Restituisce FALSE
inquarter ('03/25/2013', '03/01/2013', 0, 3)	Restituisce TRUE

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- Un caricamento precedente che contiene la funzione `inquarter()` impostata come campo '`in_quarter`' e determina quali transazioni hanno avuto luogo nello stesso trimestre del 15 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
```

```
    Load
       *,
       inquarter (date,'05/15/2022', 0) as in_quarter
       ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_quarter`

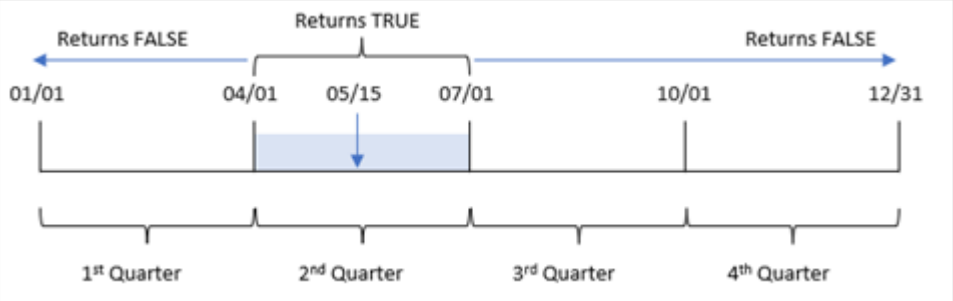
Tabella dei risultati

date	in_quarter
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	0
4/1/2022	-1
5/7/2022	-1

date	in_quarter
5/16/2022	-1
6/15/2022	-1
6/26/2022	-1
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Il campo '`in_quarter`' viene creato nell'istruzione LOAD precedente utilizzando la funzione `inquarter()`. Il primo argomento identifica il campo da valutare. Il secondo argomento è una data codificata per il 15 maggio che identifica il trimestre da definire come comparatore. Un `period_no` di `0` è l'argomento finale e garantisce che la funzione `inquarter()` non confronti i trimestri precedenti o successivi al trimestre segmentato.

Schema della funzione `inquarter()` con data base 15 maggio



Qualsiasi transazione che si verifica tra il 1° aprile e la fine del 30 giugno restituisce un risultato booleano pari a TRUE.

Esempio 2 - period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata 'Transactions'.
- Un caricamento precedente che contiene la funzione inquarter() impostata come campo 'previous_quarter' e determina quali transazioni hanno avuto luogo nello trimestre precedente quello del 15 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       inquarter (date,'05/15/2022', -1) as previous_qtr
       ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

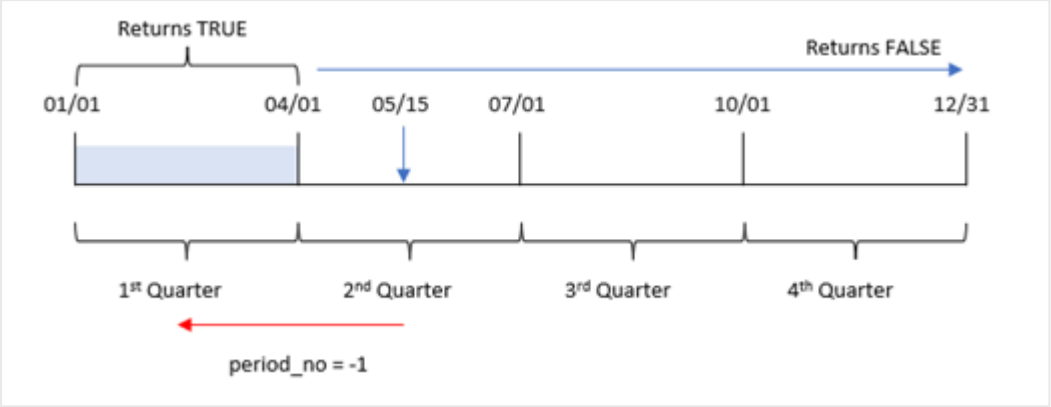
- date
- previous_qtr

Tabella dei risultati

date	previous_qtr
1/7/2022	-1
1/19/2022	-1
2/5/2022	-1
2/28/2022	-1
3/16/2022	-1
4/1/2022	0
5/7/2022	0
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

L'utilizzo di `-1` come argomento `period_no` nella funzione `inquarter()` sposta i confini del trimestre del comparatore indietro di un intero trimestre. Il 15 maggio cade nel secondo trimestre dell'anno e quindi il segmento equivale inizialmente al trimestre dal 1° aprile al 30 giugno. `period_no` compensa questo segmento con un valore negativo di tre mesi e fa sì che i confini della data diventino dal 1° gennaio al 30 marzo.

Schema della funzione `inquarter()` con data base 15 maggio



Pertanto, qualsiasi transazione che si verifichi tra il 1° gennaio e il 30 marzo restituirà un risultato booleano di TRUE.

Esempio 3 - first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- Un caricamento precedente che contiene la funzione `inquarter()` impostata come campo '`in_quarter`' e determina quali transazioni hanno avuto luogo nello stesso trimestre del 15 maggio 2022.

Tuttavia, in questo esempio, il criterio organizzativo prevede che marzo sia il primo mese dell'anno finanziario.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       inquarter (date,'05/15/2022', 0, 3) as in_quarter
       ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
```

```
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_qtr`

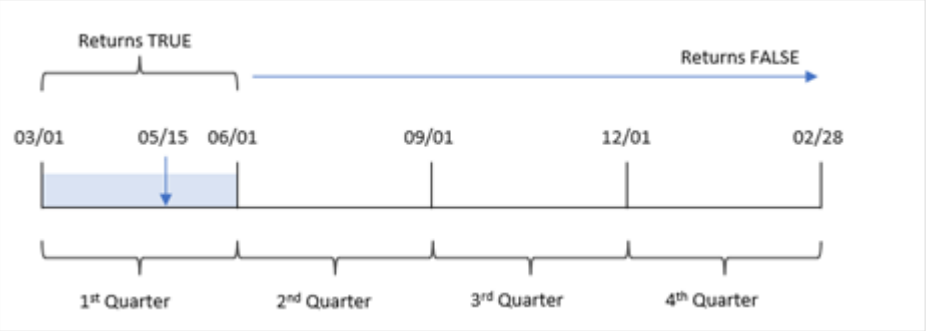
Tabella dei risultati

date	previous_qtr
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	-1
4/1/2022	-1
5/7/2022	-1
5/16/2022	-1
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0

date	previous_qtr
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Utilizzando `3` come argomento `first_month_of_year` nella funzione `inquarter()`, si imposta il 1° marzo come inizio dell'anno e poi si divide l'anno in trimestri. Pertanto, i segmenti trimestrali sono marzo-maggio, giugno-agosto, settembre-novembre, dicembre-febbraio. `base_date` del 15 maggio stabilisce il trimestre marzo-maggio come trimestre di confronto per la funzione.

Lo schema della funzione `inquarter()` con marzo impostato come primo mese dell'anno



Pertanto, qualsiasi transazione che si verifichi tra il 1° marzo e il 31 marzo restituirà un risultato booleano di TRUE.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- Un caricamento precedente che contiene la funzione `inquarter()` impostata come campo '`in_quarter`' e determina quali transazioni hanno avuto luogo nello stesso trimestre del 15 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

- `date`

Creare la seguente misura per calcolare se le transazioni sono avvenute nello stesso trimestre del 15 maggio:

```
=inquarter(date,'05/15/2022', 0)
```

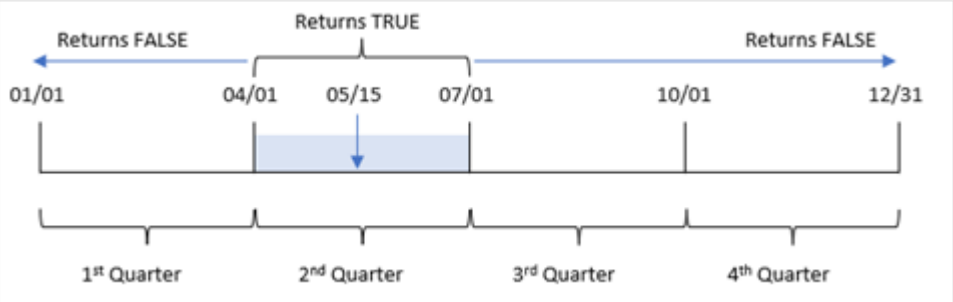
Tabella dei risultati

date	in_quarter
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0

date	in_quarter
3/16/2022	0
4/1/2022	-1
5/7/2022	-1
5/16/2022	-1
6/15/2022	-1
6/26/2022	-1
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

La misura '`in_quarter`' viene creata nell'oggetto grafico utilizzando la funzione `inquarter()`. Il primo argomento identifica il campo da valutare. Il secondo argomento è una data codificata per il 15 maggio che identifica il trimestre da definire come comparatore. Un `period_no` di `0` è l'argomento finale e garantisce che la funzione `inquarter()` non confronti i trimestri precedenti o successivi al trimestre segmentato.

Schema della funzione `inquarter()` con data base 15 maggio



Qualsiasi transazione che si verifica tra il 1° aprile e la fine del 30 giugno restituisce un risultato booleano pari a TRUE.

Esempio 5 - Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata '`Products`'.
- La tabella contiene i seguenti campi:
    - ID prodotto
    - tipo di prodotto
    - data di produzione
    - prezzo di costo

Si è determinato che, a causa di un errore delle apparecchiature, i prodotti fabbricati nella settimana nel trimestre del 15 maggio 2022 erano difettosi. L'utente finale vorrebbe un grafico che visualizzi, per nome del trimestre, lo stato dei prodotti fabbricati "difettosi" o "non difettosi" e il costo dei prodotti fabbricati in quel trimestre.

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,manufacture_date,cost_price
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

```
=quartername(manufacture_date)
```

Creare le seguenti misure:

- `=if(only(InQuarter(manufacture_date,makedate(2022,05,15),0)),'Defective','Faultless')`, per identificare quali prodotti sono difettosi e quali non lo sono utilizzando la funzione `inquarter()`.
- `=sum(cost_price)`, per mostrare la somma dei costi di ciascun prodotto.

Procedere come indicato di seguito:

1. Impostare la misura **Formattazione numero** su **Denaro**.
2. In **Aspetto**, disattivare **Totals**.

Tabella dei risultati

quartername (manufacture_date)	=if(only(InQuarter(manufacture_date,makedate (2022,05,15),0)),'Defective','Faultless')	Sum(cost_ price)
gen-mar 2022	Non difettoso	253.89
apr-giu 2022	Difettoso	351.48
lug-set 2022	Non difettoso	446.31
ott-dic 2022	Non difettoso	163.91

La funzione `inquarter()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per qualsiasi prodotto fabbricato nel trimestre che contiene il 15 maggio, la funzione `inquarter()` restituisce un valore booleano di TRUE e contrassegna i prodotti come 'difettosi'. Qualsiasi prodotto che restituisce un valore di FALSE, e che quindi non è stato fabbricato nel trimestre, viene contrassegnato come "non difettoso".

inquartertodate

Questa funzione restituisce True se **timestamp** ricade all'interno della parte del trimestre contenente **base_date** fino a includere l'ultimo millisecondo di **base_date**.

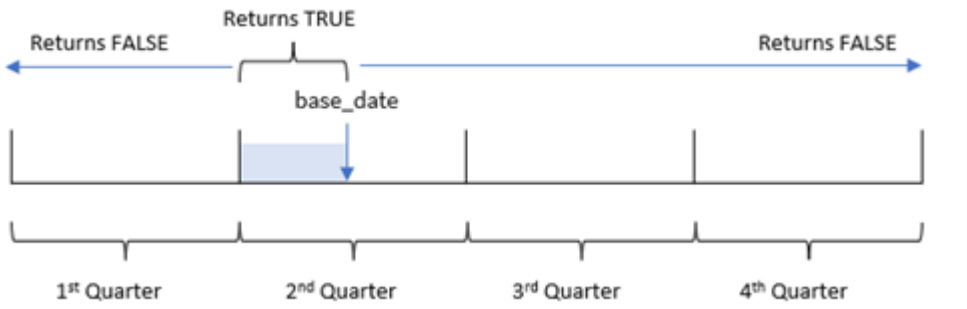
Sintassi:

```
InQuarterToDate (timestamp, base_date, period_no [, first_month_of_year])
```

Tipo di dati restituiti: Booleano

> *In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.*

Diagramma della funzione inquartertodate



La funzione `inquartertodate()` divide l'anno in quattro trimestri uguali tra il 1 gennaio e il 31 dicembre (o l'inizio dell'anno definito dall'utente e la data di fine corrispondente). Usando `base_date`, la funzione segmenterà quindi un particolare trimestre e `base_date` identificherà sia il trimestre che la data massima consentita per quel segmento di trimestre. Infine, la funzione restituisce un risultato booleano quando si confrontano i valori di data prescritti con quel segmento.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare il trimestre.
period_no	Il trimestre può essere differito mediante **period_no**. **period_no** è un numero intero, in cui il valore 0 indica il trimestre che contiene **base_date**. I valori negativi di **period_no** indicano i trimestri precedenti, mentre i valori positivi indicano i trimestri successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

Casi di utilizzo

La funzione `inquartertodate()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un'espressione `if`. La funzione `inquartertodate()` restituisce un'aggregazione o un calcolo che dipende dal fatto che una data valutata si sia verificata in un determinato trimestre fino a includere la data in questione.

Ad esempio, la funzione `inquartertodate()` può essere utilizzata per identificare tutte le apparecchiature prodotte in un in un determinato trimestre fino a una data specifica.

Esempi di funzioni

Esempio	Risultato
`inquartertodate ('01/25/2013', '03/25/2013', 0)`	Restituisce`TRUE`, poiché il valore di `timestamp`, 25/01/2013, rientra nel periodo di tre mesi dal giorno 01/01/2013 al giorno 25/03/2013, in cui il valore di `base_date`, 25/03/2013, è incluso.

Esempio	Risultato
`inquartertodate ('04/26/2013', '03/25/2013', 0)`	Restituisce `FALSE`, poiché la data 26/04/2013 non ricade nello stesso periodo dell'esempio precedente.
`inquartertodate ('02/25/2013', '06/09/2013', -1)`	Restituisce `TRUE`, poiché il valore di `period_no`, -1, sposta il periodo di ricerca indietro per un intervallo di tempo di tre mesi (un quarto dell'anno). Questo include il periodo di ricerca tra le date 01/01/2013 e 09/03/2013.
`inquartertodate ('03/25/2006', '04/15/2006', 0, 2)`	Restituisce `TRUE`, poiché il valore di `first_month_of_year` è impostato su 2, che fa rientrare il periodo di ricerca tra il 01/02/2006 e il 15/04/2006 anziché dal 01/04/2006 al 15/04/2006.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `in_quarter_to_date`, che determina quali transazioni sono avvenute nel trimestre fino al 15 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
```

```
    Load
        *,
        inquartertodate(date,'05/15/2022', 0) as in_quarter_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_quarter_to_date`

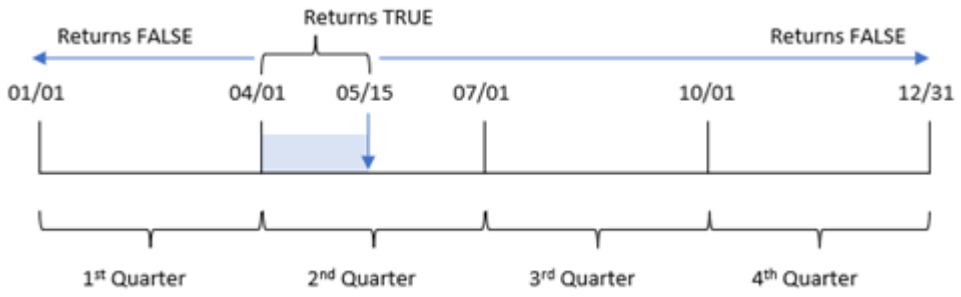
Tabella dei risultati

data	in_quarter_to_date
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	0
4/1/2022	-1
5/7/2022	-1

data	in_quarter_to_date
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Il campo `in_quarter_to_date` viene creato nell'istruzione LOAD precedente utilizzando la funzione `inquartertodate()`. Il primo argomento fornito identifica il campo da valutare. Il secondo argomento è una data codificata per il 15 maggio, che è il `base_date` che identifica il trimestre da segmentare e definisce il limite finale di quel segmento. Un `period_no` di 0 è l'argomento finale, il che significa che la funzione non confronta i trimestri precedenti o successivi al trimestre segmentato.

Diagramma della funzione inquartertodate, senza argomenti aggiuntivi



Qualsiasi transazione avvenuta tra il 1 aprile e il 15 maggio restituisce il risultato booleano TRUE. Le date delle transazioni del 16 maggio e successive restituiranno FALSE, così come tutte le transazioni precedenti al 1 aprile.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_qtr_to_date`, che determina quali transazioni sono avvenute nel trimestre precendente al segmento del trimestre che termina il 16 maggio 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inquartertodate(date,'05/15/2022', -1) as previous_qtr_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

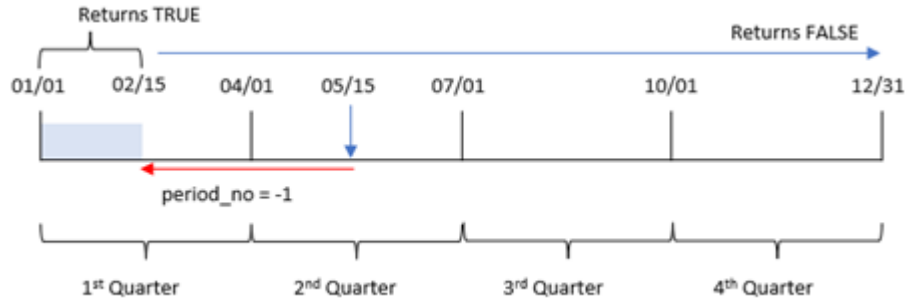
- `date`
- `previous_qtr_to_date`

Tabella dei risultati

data	previous_qtr_to_date
1/7/2022	-1
1/19/2022	-1
2/5/2022	-1
2/28/2022	0
3/16/2022	0
4/1/2022	0
5/7/2022	0
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Un valore `period_no` di -1 indica che la funzione `inquartertodate ()` confronta il segmento del trimestre di input con il trimestre precedente. Il 15 maggio rientra nel secondo trimestre dell'anno, quindi il segmento inizialmente corrisponde al periodo compreso tra il 1 aprile e il 15 maggio. `period_no` quindi compensa questo segmento di tre mesi prima, facendo sì che i limiti della data diventino dal 1 gennaio al 15 febbraio.

Diagramma della funzione inquartertodate, esempio di period_no



Pertanto, qualsiasi transazione che si verifichi tra il 1 gennaio e il 15 febbraio restituirà il risultato booleano `TRUE`.

FirstMonthOfYear

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `in_quarter_to_date`, che determina quali transazioni sono avvenute nello stesso trimestre fino al 15 maggio 2022.

In questo esempio, marzo è stato impostato come primo mese dell'anno fiscale.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inquartertodate(date,'05/15/2022', 0,3) as in_quarter_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
```

```
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_quarter_to_date`

Tabella dei risultati

data	in_quarter_to_date
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	-1
4/1/2022	-1
5/7/2022	-1
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0

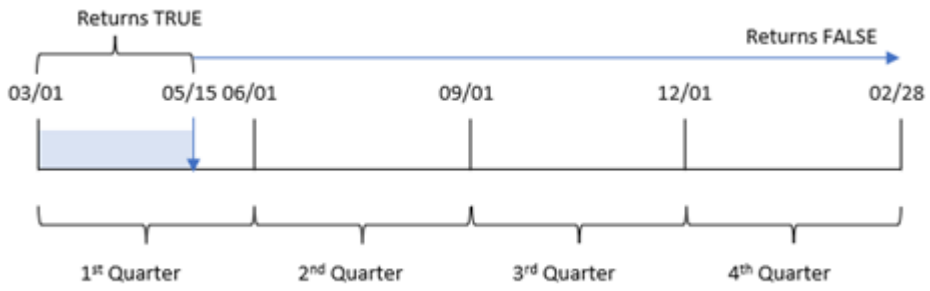
data	in_quarter_to_date
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

Utilizzando 3 come argomento `first_month_of_year` nella funzione `inquartertodate()`, la funzione inizia l'anno il 1 marzo, quindi divide l'anno in trimestri. Pertanto, i segmenti del trimestre sono:

- Da marzo a maggio
- Da giugno ad agosto
- Da settembre a novembre
- Da dicembre a febbraio

Il `base_date` del 15 maggio segmenta quindi il trimestre da marzo a maggio impostando il limite finale al 15 maggio.

Diagramma della funzione inquartertodate, esempio di first_month_of_year



Pertanto, qualsiasi transazione che si verifichi tra il 1 marzo e il 15 maggio restituirà un risultato booleano TRUE, mentre le transazioni con una data che non rientra in questi limiti restituiranno il valore FALSE.

Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che determina quali transazioni sono avvenute lo stesso trimestre del 15 maggi viene creato come misura in un oggetto grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`date`.

Creare la seguente misura:

```
=inquartertodate(date,'05/15/2022', 0)
```

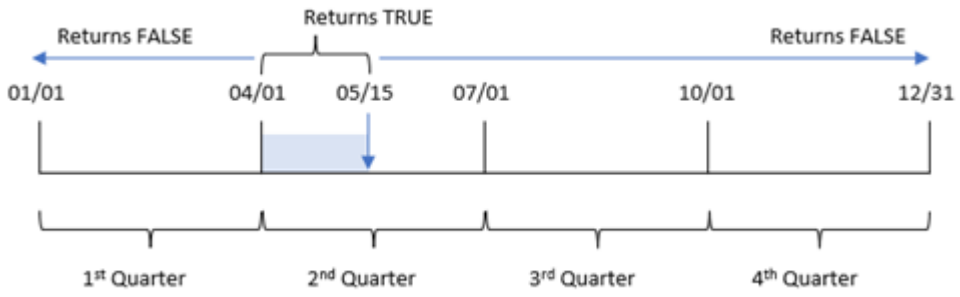
Tabella dei risultati

data	=inquartertodate(date,'15/05/2022', 0)
1/7/2022	0
1/19/2022	0
2/5/2022	0
2/28/2022	0
3/16/2022	0

data	=inquartertodate(date,'15/05/2022', 0)
4/1/2022	-1
5/7/2022	-1
5/16/2022	0
6/15/2022	0
6/26/2022	0
7/9/2022	0
7/22/2022	0
7/23/2022	0
7/27/2022	0
8/2/2022	0
8/8/2022	0
8/19/2022	0
9/26/2022	0
10/14/2022	0
10/29/2022	0

La misura `in_quarter_to_date` viene creata in un oggetto grafico utilizzando la funzione `inquartertodate()`. Il primo argomento è il campo della data da valutare. Il secondo argomento è una data codificata per il 15 maggio, che è il valore `base_date` che identifica il trimestre da segmentare e definisce il limite finale di quel segmento. Un `period_no` uguale a 0 è l'argomento finale, il che significa che la funzione non confronta i trimestri precedenti o successivi al trimestre segmentato.

Diagramma della funzione inquartertodate, esempio di oggetto grafico



Qualsiasi transazione avvenuta tra il 1 aprile e il 15 maggio restituisce il risultato booleano `TRUE`. Le transazioni del 16 maggio e successive restituiranno `FALSE`, così come tutte le transazioni precedenti al 1 aprile.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Products`.
- Informazioni relative all'ID prodotto, alla data di produzione e al prezzo di costo.

Il 15 maggio 2022, un errore dell'attrezzatura è stato identificato nel processo di produzione e risolto. I prodotti fabbricati in quel trimestre fino a questa data saranno difettosi. L'utente finale desidera un oggetto grafico che visualizzi, in base al nome del trimestre, se lo stato dei prodotti è "difettoso" o "senza difetti" e il costo dei prodotti fabbricati in quel trimestre fino a quella data.

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,manufacture_date,cost_price
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella. Creare una dimensione per mostrare i nomi dei trimestri:
   `=quartername(manufacture_date)`
2. Quindi, creare una dimensione per identificare quali prodotti sono difettosi e quali senza difetti:
   `=if(inquartertodate(manufacture_date,makedate(2022,05,15),0),'Defective','Faultless')`
3. Creare una misura per sommare il valore `cost_price` dei prodotti:
   `=sum(cost_price)`
4. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

quartername (manufacture_date)	if(inquartertodate(manufacture_date,makedate (2022,05,15),0),'Defective','Faultless')	=sum(cost_ price)
gen-mar 2022	Non difettoso	$253.89
A`p-giugno 2022	Non difettoso	$229.03
apr-giu 2022	Difettoso	$122.45
lug-set 2022	Non difettoso	$446.31
ott-dic 2022	Non difettoso	$163.91

La funzione `inquartertodate()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per coloro che restituiscono il valore booleano `TRUE`, contrassegna i prodotti come `'Defective'`. Per qualsiasi prodotto che restituisce il valore `FALSE`, e quindi non realizzato nel trimestre fino al 15 maggio incluso, contrassegna i prodotti come `'Faultless'`.

inweek

Questa funzione restituisce True se **timestamp** ricade all'interno della settimana contenente **base_date**.

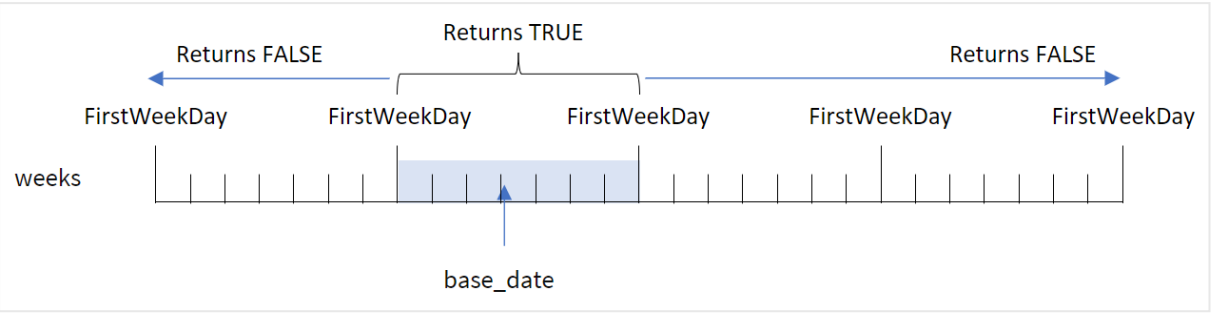
Sintassi:

```
InWeek (timestamp, base_date, period_no[, first_week_day])
```

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Schema dell'intervallo della funzione `inweek()`



La funzione `inweek()` utilizza l'argomento `base_date` per identificare in quale periodo di sette giorni cade la data. Il giorno iniziale della settimana si basa sulla variabile di sistema `FirstWeekDay`. Tuttavia, è possibile anche modificare il primo giorno della settimana utilizzando l'argomento `first_week_day` nella funzione `inweek()`.

Dopo aver definito la settimana selezionata, la funzione restituisce risultati booleani quando si confrontano i valori delle date prescritte con quel segmento di settimana.

Casi di utilizzo

La funzione `inweek()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. La funzione `inweek()` restituisce un'aggregazione o un calcolo che dipende dal fatto che una data valutata si sia verificata nella settimana con la data selezionata dell'argomento `base_date`.

Ad esempio, la funzione `inweek()` può essere utilizzata per identificare tutte le apparecchiature prodotte in una determinata settimana.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare la settimana.
period_no	La settimana può essere differita mediante **period_no**. **period_no** è un numero intero, in cui il valore 0 indica la settimana che contiene **base_date**. I valori negativi di **period_no** indicano le settimane precedenti, mentre i valori positivi indicano le settimane successive.
first_week_day	Per impostazione predefinita, il primo giorno della settimana è domenica (come determinato dalla variabile di sistema FirstWeekDay), a partire dalla mezzanotte tra sabato e domenica. Il parametro **first_week_day** sostituisce la variabile **FirstWeekDay**. Per indicare un altro giorno per l'inizio della settimana, specificare un contrassegno con valore da 0 a 6.

valori first_week_day

Giorno	Valore
Lunedì	0
Martedì	1
Mercoledì	2
Giovedì	3
Venerdì	4
Sabato	5
Domenica	6

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`inweek ('01/12/2006', '01/14/2006', 0)`	Restituisce TRUE
`inweek ('01/12/2006', '01/20/2006', 0 )`	Restituisce FALSE
`inweek ('01/12/2006', '01/14/2006', -1 )`	Restituisce FALSE
`inweek ('01/07/2006', '01/14/2006', -1)`	Restituisce TRUE

Esempio	Risultato
`inweek ('01/12/2006', '01/09/2006', 0, 3)`	Restituisce FALSE perché `first_week_day` è specificato come 3 (giovedì), il che rende il 12/01/2006 il primo giorno della settimana successiva a quella contenente il giorno 09/01/2006.

I seguenti argomenti possono aiutarti a lavorare con questa funzione:

Argomenti correlati

Argomento	Contrassegno predefinito / Valore	Descrizione
FirstWeekDay (page 224)	6 / Sunday	Definisce il giorno di inizio di ogni settimana.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il mese di gennaio 2022, caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `FirstWeekDay` che è impostata su `6` (domenica).
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
  - La funzione `inweek()`, impostata come campo '`in_week`' che determina quali transazioni hanno avuto luogo nella settimana del 14 gennaio 2022.
  - La funzione `weekday()`, impostata come campo '`week_day`' che mostra quale giorno della settimana corrisponde a ciascuna data.

Script di caricamento

```
SET FirstWeekDay=6;
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       weekday(date) as week_day,
       inweek(date,'01/14/2022', 0) as in_week
       ;
Load
*
Inline
[
```

```
id,date,amount
8188,'01/02/2022',37.23
8189,'01/05/2022',17.17
8190,'01/06/2022',88.27
8191,'01/08/2022',57.42
8192,'01/09/2022',53.80
8193,'01/10/2022',82.06
8194,'01/11/2022',40.39
8195,'01/12/2022',87.21
8196,'01/13/2022',95.93
8197,'01/14/2022',45.89
8198,'01/15/2022',36.23
8199,'01/16/2022',25.66
8200,'01/17/2022',82.77
8201,'01/18/2022',69.98
8202,'01/26/2022',76.11
8203,'01/27/2022',25.12
8204,'01/28/2022',46.23
8205,'01/29/2022',84.21
8206,'01/30/2022',96.24
8207,'01/31/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week_day`
- `in_week`

Tabella dei risultati

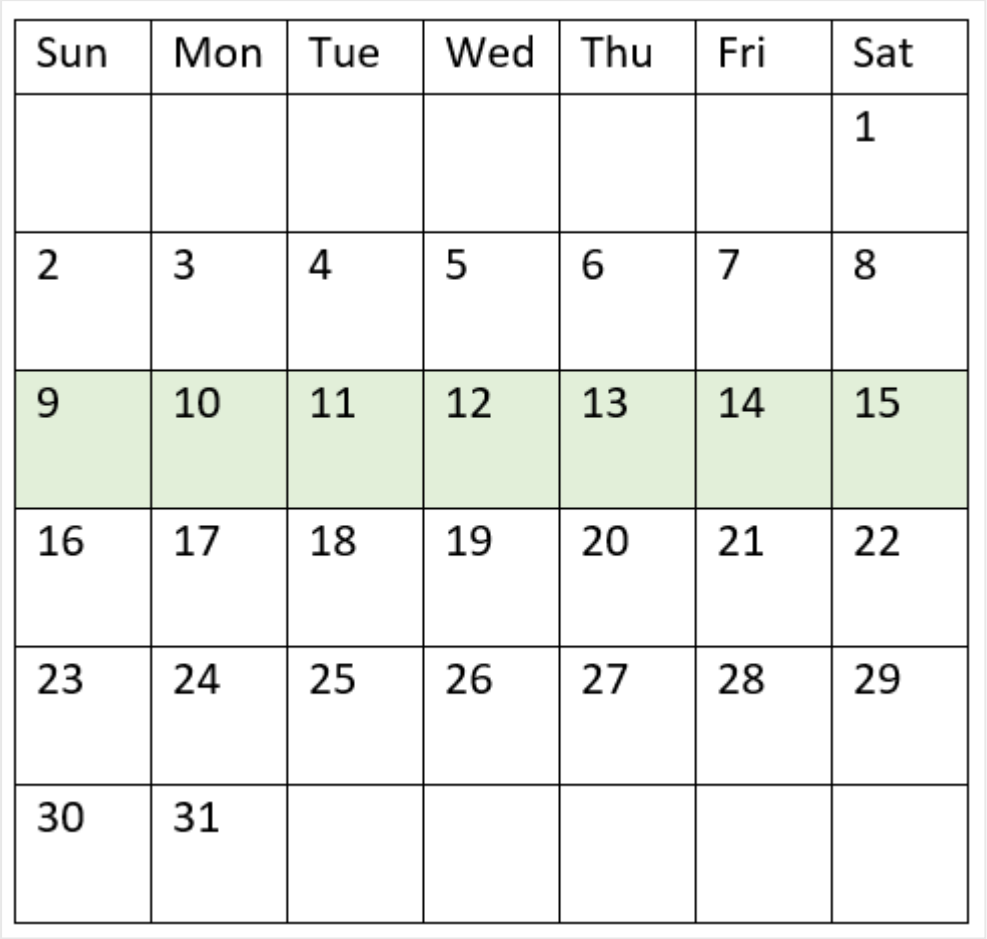
data	week_day	in_week
01/02/2022	Sun	0
01/05/2022	Wed	0
01/06/2022	Thu	0
01/08/2022	Sat	0
01/09/2022	Sun	-1
01/10/2022	Mon	-1
01/11/2022	Tue	-1
01/12/2022	Wed	-1
01/13/2022	Thu	-1
01/14/2022	Fri	-1
01/15/2022	Sat	-1

data	week_day	in_week
01/16/2022	Sun	0
01/17/2022	Mon	0
01/18/2022	Tue	0
01/26/2022	Wed	0
01/27/2022	Thu	0
01/28/2022	Fri	0
01/29/2022	Sat	0
01/30/2022	Sun	0
01/31/2022	Mon	0

Il campo '`in_week`' viene creato nell'istruzione LOAD precedente utilizzando la funzione `inweek()`. Il primo argomento identifica il campo da valutare. Il secondo argomento è una data codificata, il 14 gennaio, che è il `base_date`. L'argomento `base_date` collabora con la variabile di sistema `FirstWeekDay` per identificare la settimana del comparatore. Un `period_no` di `0` — significa che la funzione non sta confrontando le settimane precedenti o successive alla settimana segmentata – è l'argomento finale.

La variabile di sistema determina che le settimane iniziano di domenica e terminano di sabato.FirstWeekDay Pertanto, gennaio viene suddiviso in settimane secondo lo schema seguente, con le date comprese tra il 9 e il 15 gennaio che forniscono il periodo valido per il calcolo `inweek()`:

Schema del calendario con evidenziato l'intervallo della funzione `inweek()`



Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Qualsiasi transazione avvenuta tra il 9 e il 15 gennaio restituisce il risultato booleano di `TRUE`.

Esempio 2 - period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `FirstWeekDay` che è impostata su `6` (domenica).
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
  - La funzione `inweek ()`, impostata come campo '`prev_week`' che determina quali transazioni hanno avuto luogo in un'intera settimana prima della settimana del 14 gennaio 2022.

- La funzione `weekday()`, impostata come campo '`week_day`' che mostra quale giorno della settimana corrisponde a ciascuna data.

Script di caricamento

```
SET FirstWeekDay=6;
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       weekday(date) as week_day,
       inweek(date,'01/14/2022', -1) as prev_week
       ;
Load
*
Inline
[
id,date,amount
8188,'01/02/2022',37.23
8189,'01/05/2022',17.17
8190,'01/06/2022',88.27
8191,'01/08/2022',57.42
8192,'01/09/2022',53.80
8193,'01/10/2022',82.06
8194,'01/11/2022',40.39
8195,'01/12/2022',87.21
8196,'01/13/2022',95.93
8197,'01/14/2022',45.89
8198,'01/15/2022',36.23
8199,'01/16/2022',25.66
8200,'01/17/2022',82.77
8201,'01/18/2022',69.98
8202,'01/26/2022',76.11
8203,'01/27/2022',25.12
8204,'01/28/2022',46.23
8205,'01/29/2022',84.21
8206,'01/30/2022',96.24
8207,'01/31/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

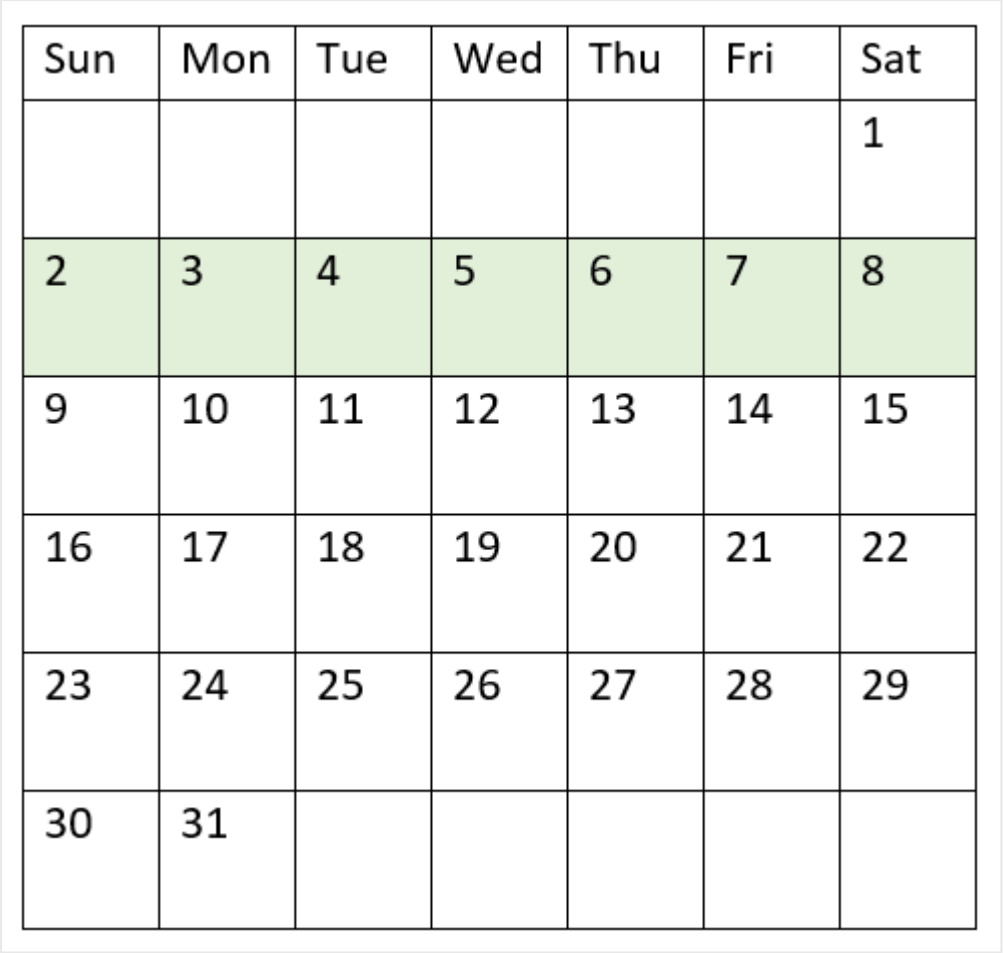
- `date`
- `week_day`
- `prev_week`

Tabella dei risultati

data	week_day	prev_week
01/02/2022	Sun	-1
01/05/2022	Wed	-1
01/06/2022	Thu	-1
01/08/2022	Sat	-1
01/09/2022	Sun	0
01/10/2022	Mon	0
01/11/2022	Tue	0
01/12/2022	Wed	0
01/13/2022	Thu	0
01/14/2022	Fri	0
01/15/2022	Sat	0
01/16/2022	Sun	0
01/17/2022	Mon	0
01/18/2022	Tue	0
01/26/2022	Wed	0
01/27/2022	Thu	0
01/28/2022	Fri	0
01/29/2022	Sat	0
01/30/2022	Sun	0
01/31/2022	Mon	0

L'utilizzo di `-1` come argomento `period_no` nella funzione `inweek()` sposta i confini della settimana del comparatore indietro di sette giorni interi. Con un `period_no` di `0` la settimana sarebbe compresa tra il 9 e il 15 gennaio. Eppure in questo esempio, il `period_no` di `-1` sposta il confine iniziale e finale di questo segmento indietro di una settimana. I limiti di data diventano dal 2 gennaio all'8 gennaio.

Schema del calendario con evidenziato l'intervallo della funzione `inweek()`



Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Pertanto, qualsiasi transazione che si verifichi tra il 2 gennaio e l'8 gennaio restituirà il risultato booleano di TRUE.

Esempio 3 - first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `FirstWeekDay` che è impostata su `6` (domenica).
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
    - La funzione `inweek()`, impostata come campo '`in_week`' che determina quali transazioni hanno avuto luogo nella settimana del 14 gennaio 2022.

- La funzione `weekday()`, impostata come campo '`week_day`' che mostra quale giorno della settimana corrisponde a ciascuna data.

Script di caricamento

```
SET FirstWeekDay=6;
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       weekday(date) as week_day,
       inweek(date,'01/14/2022', 0, 0) as in_week
       ;
Load
*
Inline
[
id,date,amount
8188,'01/02/2022',37.23
8189,'01/05/2022',17.17
8190,'01/06/2022',88.27
8191,'01/08/2022',57.42
8192,'01/09/2022',53.80
8193,'01/10/2022',82.06
8194,'01/11/2022',40.39
8195,'01/12/2022',87.21
8196,'01/13/2022',95.93
8197,'01/14/2022',45.89
8198,'01/15/2022',36.23
8199,'01/16/2022',25.66
8200,'01/17/2022',82.77
8201,'01/18/2022',69.98
8202,'01/26/2022',76.11
8203,'01/27/2022',25.12
8204,'01/28/2022',46.23
8205,'01/29/2022',84.21
8206,'01/30/2022',96.24
8207,'01/31/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week_day`
- `in_week`

Tabella dei risultati

data	week_day	in_week
01/02/2022	Sun	0
01/05/2022	Wed	0
01/06/2022	Thu	0
01/08/2022	Sat	0
01/09/2022	Sun	0
01/10/2022	Mon	-1
01/11/2022	Tue	-1
01/12/2022	Wed	-1
01/13/2022	Thu	-1
01/14/2022	Fri	-1
01/15/2022	Sat	-1
01/16/2022	Sun	-1
01/17/2022	Mon	0
01/18/2022	Tue	0
01/26/2022	Wed	0
01/27/2022	Thu	0
01/28/2022	Fri	0
01/29/2022	Sat	0
01/30/2022	Sun	0
01/31/2022	Mon	0

Utilizzando `0` come argomento `first_week_day` nella funzione `inweek()` si sostituisce la variabile di sistema `FirstWeekDay` e si imposta lunedì come primo giorno della settimana.

Schema del calendario con evidenziato l'intervallo della funzione `inweek()`

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Pertanto, qualsiasi transazione che si verifichi tra il 10 e il 16 gennaio restituirà un risultato booleano di TRUE.

Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Creare una misura nella tabella dei risultati per determinare quali transazioni sono state effettuate nella settimana del 14 gennaio 2022.

Script di caricamento

```
SET FirstWeekDay=6;
SET DateFormat='MM/DD/YYYY';

Transactions:
```

```
Load
*
Inline
[
id,date,amount
8188,'01/02/2022',37.23
8189,'01/05/2022',17.17
8190,'01/06/2022',88.27
8191,'01/08/2022',57.42
8192,'01/09/2022',53.80
8193,'01/10/2022',82.06
8194,'01/11/2022',40.39
8195,'01/12/2022',87.21
8196,'01/13/2022',95.93
8197,'01/14/2022',45.89
8198,'01/15/2022',36.23
8199,'01/16/2022',25.66
8200,'01/17/2022',82.77
8201,'01/18/2022',69.98
8202,'01/26/2022',76.11
8203,'01/27/2022',25.12
8204,'01/28/2022',46.23
8205,'01/29/2022',84.21
8206,'01/30/2022',96.24
8207,'01/31/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

- `date`

Creare le seguenti misure:

- `=inweek (date,'01/14/2022',0)`, per calcolare se le transazioni sono avvenute nella stessa settimana del 14 gennaio.
- `=weekday(date)`, per mostrare quale giorno della settimana corrisponde a ciascuna data.

Tabella dei risultati

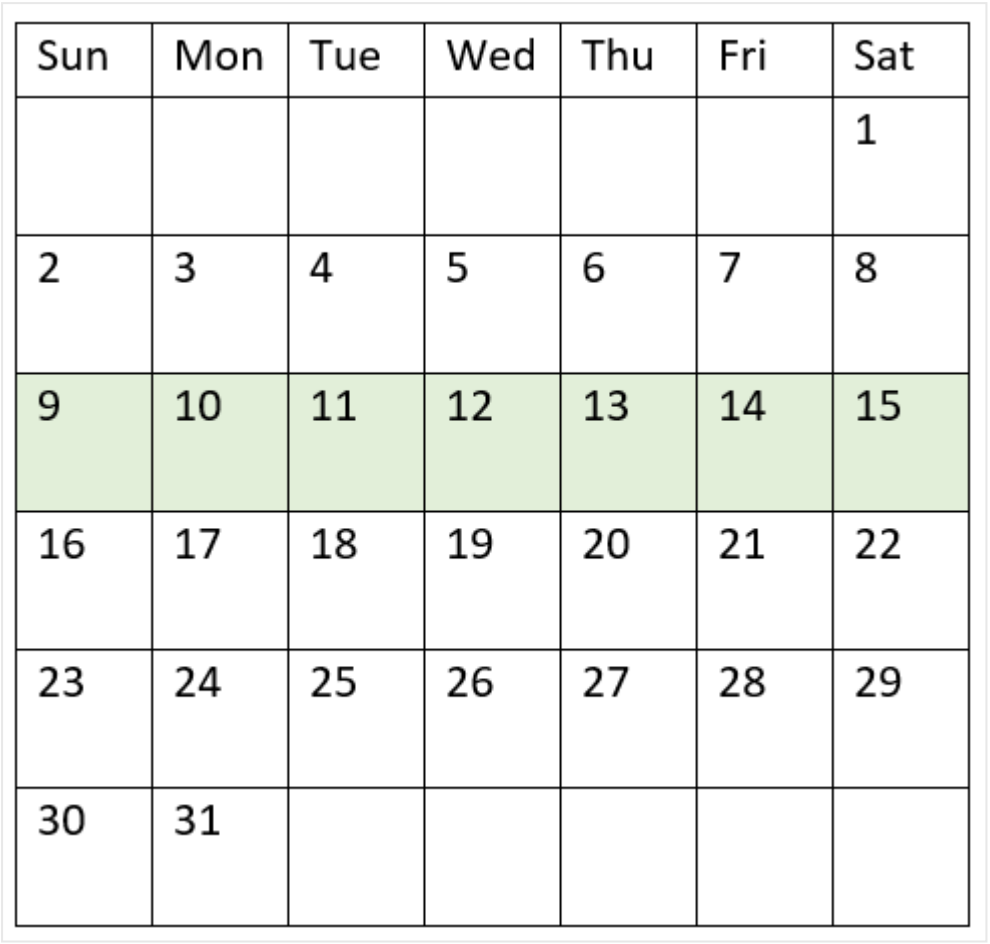
data	week_day	=inweek (date,'01/14/2022',0)
01/02/2022	Sun	0
01/05/2022	Wed	0
01/06/2022	Thu	0
01/08/2022	Sat	0
01/09/2022	Sun	-1
01/10/2022	Mon	-1

data	week_day	=inweek (date,'01/14/2022',0)
01/11/2022	Tue	-1
01/12/2022	Wed	-1
01/13/2022	Thu	-1
01/14/2022	Fri	-1
01/15/2022	Sat	-1
01/16/2022	Sun	0
01/17/2022	Mon	0
01/18/2022	Tue	0
01/26/2022	Wed	0
01/27/2022	Thu	0
01/28/2022	Fri	0
01/29/2022	Sat	0
01/30/2022	Sun	0
01/31/2022	Mon	0

La misura '`in_week`' viene creata nel grafico utilizzando la funzione `inweek()`. Il primo argomento identifica il campo da valutare. Il secondo argomento è una data codificata, il 14 gennaio, che è il `base_date`. L'argomento `base_date` collabora con la variabile di sistema `FirstWeekDay` per identificare la settimana del comparatore. Un `period_no` di `0` rappresenta l'argomento finale.

La variabile di sistema determina che le settimane iniziano di domenica e terminano di sabato.FirstWeekDay Pertanto, gennaio viene suddiviso in settimane secondo lo schema seguente, con le date comprese tra il 9 e il 15 gennaio che forniscono il periodo valido per il calcolo `inweek()`:

Schema del calendario con evidenziato l'intervallo della funzione `inweek()`



Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Qualsiasi transazione avvenuta tra il 9 e il 15 gennaio restituisce il risultato booleano di TRUE.

Esempio 5 - Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata '`Products`'.
- La tabella contiene i seguenti campi:
  - ID prodotto
  - tipo di prodotto
  - data di produzione
  - prezzo di costo

Si è determinato che, a causa di un errore delle apparecchiature, i prodotti fabbricati nella settimana del 12 gennaio erano difettosi. L'utente finale vorrebbe un grafico che visualizzi, per settimana, lo stato dei prodotti fabbricati "difettosi" o "non difettosi" e il costo dei prodotti fabbricati in quella settimana.

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,manufacture_date,cost_price
8188,'01/02/2022',37.23
8189,'01/05/2022',17.17
8190,'01/06/2022',88.27
8191,'01/08/2022',57.42
8192,'01/09/2022',53.80
8193,'01/10/2022',82.06
8194,'01/11/2022',40.39
8195,'01/12/2022',87.21
8196,'01/13/2022',95.93
8197,'01/14/2022',45.89
8198,'01/15/2022',36.23
8199,'01/16/2022',25.66
8200,'01/17/2022',82.77
8201,'01/18/2022',69.98
8202,'01/26/2022',76.11
8203,'01/27/2022',25.12
8204,'01/28/2022',46.23
8205,'01/29/2022',84.21
8206,'01/30/2022',96.24
8207,'01/31/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

- `=weekname(manufacture_date)`

Creare le seguenti misure:

- `=if(only(inweek(manufacture_date,makedate(2022,01,12),0)),'Defective','Faultless')`, per identificare quali prodotti sono difettosi e quali non lo sono utilizzando la funzione `inweek()`.
- `=sum(cost_price)`, per mostrare la somma dei costi di ciascun prodotto.

Procedere come indicato di seguito:

1. Impostare la misura **Formattazione numero** su **Denaro**.
2. In **Aspetto**, disattivare **Totals**.

Tabella dei risultati

weekname (manufacture_date)	=if(only(inweek(manufacture_date,makedate (2022,01,12),0)), 'Defective','Faultless')	=sum(cost_ price)
2022/02	Non difettoso	200.09
2022/03	Difettoso	441.51
2022/04	Non difettoso	178.41
2022/05	Non difettoso	231.67
2022/06	Non difettoso	163.91

La funzione `inweek()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per qualsiasi prodotto fabbricato nella settimana del 12 gennaio, la funzione `inweek()` restituisce un valore booleano TRUE e contrassegna i prodotti come "difettosi". Per tutti i prodotti che restituiscono un valore pari a FALSE, e che quindi non sono stati prodotti in quella settimana, vengono contrassegnati come "non difettosi".

inweektodate

Questa funzione restituisce True se **timestamp** ricade all'interno della parte della settimana contenente **base_date** fino a includere l'ultimo millisecondo di **base_date**.

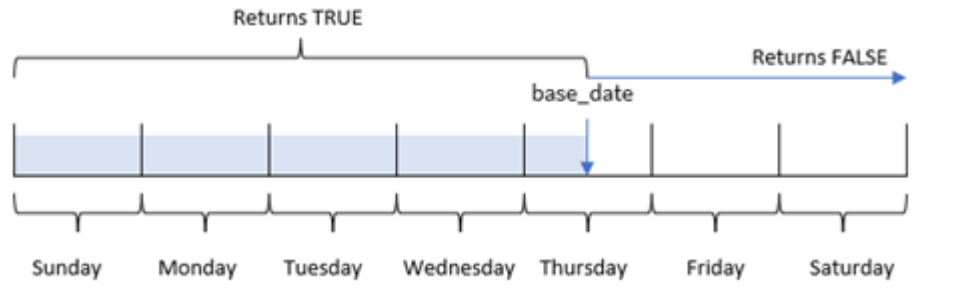
Sintassi:

```
InWeekToDate (timestamp, base_date, period_no [, first_week_day])
```

Tipo di dati restituiti: Booleano

> *In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.*

Diagramma della funzione inweektodate



La funzione `inweektodate()` utilizza il parametro `base_date` per identificare una data limite massima di un segmento settimanale, nonché la data corrispondente per l'inizio della settimana, che si basa sulla

variabile di sistema `FirstWeekDay` (o sul parametro `first_week_day` definito dall'utente). Una volta che questo segmento di settimana viene definito, la funzione restituisce un risultato booleano quando si confrontano i valori di data prescritti con quel segmento.

Casi di utilizzo

La funzione `inweektodate()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un'espressione `if`. Questa restituisce un'aggregazione o un calcolo che dipende dal fatto che una data valutata si sia verificata nella settimana in questione fino a includere una data specifica.

Ad esempio, la funzione `inweektodate()` può essere utilizzata per calcolare tutte le vendite effettuate durante una settimana specificata fino a una data particolare.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare la settimana.
period_no	La settimana può essere differita mediante **period_no**. **period_no** è un numero intero, in cui il valore 0 indica la settimana che contiene **base_date**. I valori negativi di **period_no** indicano le settimane precedenti, mentre i valori positivi indicano le settimane successive.
first_week_day	Per impostazione predefinita, il primo giorno della settimana è domenica (come determinato dalla variabile di sistema FirstWeekDay), a partire dalla mezzanotte tra sabato e domenica. Il parametro **first_week_day** sostituisce la variabile **FirstWeekDay**. Per indicare un altro giorno per l'inizio della settimana, specificare un contrassegno con valore da 0 a 6. Per una settimana che inizia il lunedì e termina la domenica, usare il contrassegno 0 per lunedì, 1 per martedì, 2 per mercoledì, 3 per giovedì, 4 per venerdì, 5 per sabato e 6 per domenica.

Esempi di funzioni

Esempio	Interazione
`inweektodate ('01/12/2006', '01/12/2006', 0)`	Restituisce `TRUE`.
`inweektodate ('01/12/2006', '01/11/2006', 0)`	Restituisce `FALSE`.
`inweektodate ('01/12/2006', '01/18/2006', -1)`	Restituisce `FALSE`. Perché `period_no` è specificato come -1, la data effettiva con cui viene misurato `timestamp` è 01/11/2006.

Esempio	Interazione
`inweektodate ('01/11/2006', '01/12/2006', 0, 3 )`	Restituisce `FALSE`, poiché il valore `first_week_day` è specificato come `3` (giovedì), rendendo 01/12/2006 il primo giorno della settimana successiva alla settimana che include 01/12/2006.

I seguenti argomenti possono aiutarti a lavorare con questa funzione:

Argomenti correlati

Argomento	Contrassegno predefinito / Valore	Descrizione
FirstWeekDay (page 224)	6 / Sunday	Definisce il giorno di inizio di ogni settimana.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il mese di gennaio 2022, caricato in una tabella denominata `Transactions`.
- Il campo dati fornito nel formato `TimestampFormat='M/D/YYYY h:mm:ss[.fff]'`.
- La creazione di un campo, `in_week_to_date`, che determina quali transazioni sono avvenute nella settimana fino al 14 gennaio 2022.

- La creazione di un campo aggiuntivo, denominato `weekday`, utilizzando la funzione `weekday()`. Questo nuovo campo viene creato per mostrare quale giorno della settimana corrisponde a ciascuna data.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff]';
SET FirstWeekDay=6;
Transactions:
    Load
        *,
        weekday(date) as week_day,
        inweektodate(date,'01/14/2022', 0) as in_week_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'2022-01-02 12:22:06',37.23
8189,'2022-01-05 01:02:30',17.17
8190,'2022-01-06 15:36:20',88.27
8191,'2022-01-08 10:58:35',57.42
8192,'2022-01-09 08:53:32',53.80
8193,'2022-01-10 21:13:01',82.06
8194,'2022-01-11 00:57:13',40.39
8195,'2022-01-12 09:26:02',87.21
8196,'2022-01-13 15:05:09',95.93
8197,'2022-01-14 18:44:57',45.89
8198,'2022-01-15 06:10:46',36.23
8199,'2022-01-16 06:39:27',25.66
8200,'2022-01-17 10:44:16',82.77
8201,'2022-01-18 18:48:17',69.98
8202,'2022-01-26 04:36:03',76.11
8203,'2022-01-27 08:07:49',25.12
8204,'2022-01-28 12:24:29',46.23
8205,'2022-01-30 11:56:56',84.21
8206,'2022-01-30 14:40:19',96.24
8207,'2022-01-31 05:28:21',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week_day`
- `in_week_to_date`

Tabella dei risultati

data	week_day	in_week_to_date
2022-01-02 12:22:06	Sun	0
2022-01-05 01:02:30	Wed	0
2022-01-06 15:36:20	Thu	0
2022-01-08 10:58:35	Sat	0
2022-01-09 08:53:32	Sun	-1
2022-01-10 21:13:01	Mon	-1
2022-01-11 00:57:13	Tue	-1
2022-01-12 09:26:02	Wed	-1
2022-01-13 15:05:09	Thu	-1
2022-01-14 18:44:57	Fri	-1
2022-01-15 06:10:46	Sat	0
2022-01-16 06:39:27	Sun	0
2022-01-17 10:44:16	Mon	0
2022-01-18 18:48:17	Tue	0
2022-01-26 04:36:03	Wed	0
2022-01-27 08:07:49	Thu	0
2022-01-28 12:24:29	Fri	0
2022-01-30 11:56:56	Sun	0
2022-01-30 14:40:19	Sun	0
2022-01-31 05:28:21	Mon	0

Il campo `in_week_to_date` viene creato nell'istruzione LOAD precedente utilizzando la funzione `inweektodate()`. Il primo argomento fornito identifica il campo da valutare. Il secondo argomento è una data codificata per il 14 gennaio, che è il valore `base_date` che identifica la settimana da segmentare e definisce il limite finale di quel segmento. Un valore `period_no` uguale a 0 è l'argomento finale, il che significa che la funzione non confronta le settimane precedenti o successive alla settimana segmentata.

La variabile di sistema `FirstWeekDay` determina che le settimane iniziano di domenica e terminano di sabato. Pertanto, gennaio viene suddiviso in settimane secondo il diagramma seguente, con le date comprese tra il 9 e il 14 gennaio che forniscono il periodo valido per il calcolo `inweekdodate()`:

Diagramma del calendario che mostra le date della transazione che restituiscono un risultato booleano TRUE

Sun	Mon	Tue	Wed	Thur	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Qualsiasi transazione avvenuta tra il 9 e il 14 gennaio restituisce un risultato booleano TRUE. Le transazioni prima e dopo quelle date restituiscono il risultato booleano FALSE.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `prev_week_to_date`, che determina quali transazioni sono avvenute nella settimana precedente al segmento della settimana che termina il 14 gennaio 2022.
- La creazione di un campo aggiuntivo, denominato `weekday`, utilizzando la funzione `weekday()`. Questo mostra quale giorno della settimana corrisponde a ciascuna data.

Script di caricamento

```
SET FirstWeekDay=6;
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff]';
Transactions:
    Load
        *,
        weekday(date) as week_day,
        inweektodate(date,'01/14/2022', -1) as prev_week_to_date
        ;
Load
*
Inline
[
id,date,amount
```

```
8188,'2022-01-02 12:22:06',37.23
8189,'2022-01-05 01:02:30',17.17
8190,'2022-01-06 15:36:20',88.27
8191,'2022-01-08 10:58:35',57.42
8192,'2022-01-09 08:53:32',53.80
8193,'2022-01-10 21:13:01',82.06
8194,'2022-01-11 00:57:13',40.39
8195,'2022-01-12 09:26:02',87.21
8196,'2022-01-13 15:05:09',95.93
8197,'2022-01-14 18:44:57',45.89
8198,'2022-01-15 06:10:46',36.23
8199,'2022-01-16 06:39:27',25.66
8200,'2022-01-17 10:44:16',82.77
8201,'2022-01-18 18:48:17',69.98
8202,'2022-01-26 04:36:03',76.11
8203,'2022-01-27 08:07:49',25.12
8204,'2022-01-28 12:24:29',46.23
8205,'2022-01-30 11:56:56',84.21
8206,'2022-01-30 14:40:19',96.24
8207,'2022-01-31 05:28:21',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week_day`
- `prev_week_to_date`

Tabella dei risultati

data	week_day	prev_week_to_date
2022-01-02 12:22:06	Sun	-1
2022-01-05 01:02:30	Wed	-1
2022-01-06 15:36:20	Thu	-1
2022-01-08 10:58:35	Sat	0
2022-01-09 08:53:32	Sun	0
2022-01-10 21:13:01	Mon	0
2022-01-11 00:57:13	Tue	0
2022-01-12 09:26:02	Wed	0
2022-01-13 15:05:09	Thu	0
2022-01-14 18:44:57	Fri	0
2022-01-15 06:10:46	Sat	0

data	week_day	prev_week_to_date
2022-01-16 06:39:27	Sun	0
2022-01-17 10:44:16	Mon	0
2022-01-18 18:48:17	Tue	0
2022-01-26 04:36:03	Wed	0
2022-01-27 08:07:49	Thu	0
2022-01-28 12:24:29	Fri	0
2022-01-30 11:56:56	Sun	0
2022-01-30 14:40:19	Sun	0
2022-01-31 05:28:21	Mon	0

Il valore `period_no` di -1 indica che la funzione `inweektodate ()` confronta il segmento del trimestre di input con la settimana precedente. Il segmento della settimana inizialmente corrisponde a una data compresa tra il 9 e il 14 gennaio. Il valore `period_no` quindi sposta sia il limite di inizio che di fine di questo segmento a una settimana prima, facendo sì che i limiti della data siano definiti tra il 2 e il 7 gennaio.

Diagramma del calendario che mostra le date della transazione che restituiscono un risultato booleano TRUE

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Pertanto, qualsiasi transazione che si verifichi tra il 2 e l'8 gennaio (non includendo lo stesso 8 gennaio) restituirà il risultato booleano `TRUE`.

Esempio 3 – first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `in_week_to_date`, che determina quali transazioni sono avvenute nella settimana fino al 14 gennaio 2022.
- La creazione di un campo aggiuntivo, denominato `weekday`, utilizzando la funzione `weekday()`. Questo mostra quale giorno della settimana corrisponde a ciascuna data.

In questo esempio viene utilizzato il domenica come primo giorno della settimana.

Script di caricamento

```
SET FirstWeekDay=6;
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff]';

Transactions:
    Load
        *,
        weekday(date) as week_day,
        inweektodate(date,'01/14/2022', 0, 0) as in_week_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'2022-01-02 12:22:06',37.23
8189,'2022-01-05 01:02:30',17.17
8190,'2022-01-06 15:36:20',88.27
8191,'2022-01-08 10:58:35',57.42
8192,'2022-01-09 08:53:32',53.80
8193,'2022-01-10 21:13:01',82.06
8194,'2022-01-11 00:57:13',40.39
8195,'2022-01-12 09:26:02',87.21
8196,'2022-01-13 15:05:09',95.93
8197,'2022-01-14 18:44:57',45.89
8198,'2022-01-15 06:10:46',36.23
8199,'2022-01-16 06:39:27',25.66
8200,'2022-01-17 10:44:16',82.77
8201,'2022-01-18 18:48:17',69.98
8202,'2022-01-26 04:36:03',76.11
8203,'2022-01-27 08:07:49',25.12
8204,'2022-01-28 12:24:29',46.23
8205,'2022-01-30 11:56:56',84.21
8206,'2022-01-30 14:40:19',96.24
8207,'2022-01-31 05:28:21',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `week_day`

- in_week_to_date

Tabella dei risultati

data	week_day	in_week_to_date
2022-01-02 12:22:06	Sun	0
2022-01-05 01:02:30	Wed	0
2022-01-06 15:36:20	Thu	0
2022-01-08 10:58:35	Sat	0
2022-01-09 08:53:32	Sun	0
2022-01-10 21:13:01	Mon	-1
2022-01-11 00:57:13	Tue	-1
2022-01-12 09:26:02	Wed	-1
2022-01-13 15:05:09	Thu	-1
2022-01-14 18:44:57	Fri	-1
2022-01-15 06:10:46	Sat	0
2022-01-16 06:39:27	Sun	0
2022-01-17 10:44:16	Mon	0
2022-01-18 18:48:17	Tue	0
2022-01-26 04:36:03	Wed	0
2022-01-27 08:07:49	Thu	0
2022-01-28 12:24:29	Fri	0
2022-01-30 11:56:56	Sun	0
2022-01-30 14:40:19	Sun	0
2022-01-31 05:28:21	Mon	0

Utilizzando 0 come argomento `first_week_day` nella funzione `inweektodate()`, l'argomento della funzione sostituisce la variabile di sistema `FirstWeekDay` e imposta il lunedì come primo giorno della settimana.

Diagramma del calendario che mostra le date della transazione che restituiscono un risultato booleano TRUE

Mon	Tue	Wed	Thu	Fri	Sat	Sun
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	17
24	25	26	27	28	29	30
31						

Pertanto, qualsiasi transazione che si verifichi tra il 10 e 15 gennaio restituirà un risultato booleano TRUE, mentre le transazioni con una data che non rientra in questi limiti restituiranno il valore FALSE.

Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che determina quali transazioni sono avvenute nella settimana fino al 14 gennaio viene creato come misura in un oggetto grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'2022-01-02 12:22:06',37.23
8189,'2022-01-05 01:02:30',17.17
8190,'2022-01-06 15:36:20',88.27
8191,'2022-01-08 10:58:35',57.42
8192,'2022-01-09 08:53:32',53.80
8193,'2022-01-10 21:13:01',82.06
8194,'2022-01-11 00:57:13',40.39
```

```
8195,'2022-01-12 09:26:02',87.21
8196,'2022-01-13 15:05:09',95.93
8197,'2022-01-14 18:44:57',45.89
8198,'2022-01-15 06:10:46',36.23
8199,'2022-01-16 06:39:27',25.66
8200,'2022-01-17 10:44:16',82.77
8201,'2022-01-18 18:48:17',69.98
8202,'2022-01-26 04:36:03',76.11
8203,'2022-01-27 08:07:49',25.12
8204,'2022-01-28 12:24:29',46.23
8205,'2022-01-30 11:56:56',84.21
8206,'2022-01-30 14:40:19',96.24
8207,'2022-01-31 05:28:21',67.67
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.
2. Per calcolare se le transazioni sono avvenute nella stessa settimana fino al 14 gennaio, creare la misura seguente:
   `=inweektodate(date,'01/14/2022',0)`
3. Per mostrare quale giorno della settimana corrisponde a ciascuna data, creare una misura aggiuntiva:
   `=weekday(date)`

Tabella dei risultati

data	week_day	in_week_to_date
2022-01-02 12:22:06	Sun	0
2022-01-05 01:02:30	Wed	0
2022-01-06 15:36:20	Thu	0
2022-01-08 10:58:35	Sat	0
2022-01-09 08:53:32	Sun	-1
2022-01-10 21:13:01	Mon	-1
2022-01-11 00:57:13	Tue	-1
2022-01-12 09:26:02	Wed	-1
2022-01-13 15:05:09	Thu	-1
2022-01-14 18:44:57	Fri	-1
2022-01-15 06:10:46	Sat	0
2022-01-16 06:39:27	Sun	0

data	week_day	in_week_to_date
2022-01-17 10:44:16	Mon	0
2022-01-18 18:48:17	Tue	0
2022-01-26 04:36:03	Wed	0
2022-01-27 08:07:49	Thu	0
2022-01-28 12:24:29	Fri	0
2022-01-30 11:56:56	Sun	0
2022-01-30 14:40:19	Sun	0
2022-01-31 05:28:21	Mon	0

Il campo `in_week_to_date` viene creato come misura nell'oggetto grafico utilizzando la funzione `inweektodate()`. Il primo argomento fornito identifica il campo da valutare. Il secondo argomento è una data codificata per il 14 gennaio, che è il valore `base_date` che identifica la settimana da segmentare e definisce il limite finale di quel segmento. Un valore `period_no` uguale a 0 è l'argomento finale, il che significa che la funzione non confronta le settimane precedenti o successive alla settimana segmentata.

La variabile di sistema determina che le settimane iniziano di domenica e terminano di sabato.FirstWeekDay Pertanto, gennaio viene suddiviso in settimane secondo il diagramma seguente, con le date comprese tra il 9 e il 14 gennaio che forniscono il periodo valido per il calcolo `inweekdodate()`:

Diagramma del calendario che mostra le date della transazione che restituiscono un risultato booleano TRUE

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Qualsiasi transazione avvenuta tra il 9 e il 14 gennaio restituisce il risultato booleano `TRUE`. Le transazioni prima e dopo quelle date restituiscono il risultato booleano `FALSE`.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Products`.
- Informazioni relative all'ID prodotto, alla data di produzione e al prezzo di costo.

Si è determinato che, a causa di un errore delle apparecchiature, i prodotti fabbricati nella settimana del 12 gennaio erano difettosi. Il problema è stato risolto il 13 gennaio. L'utente finale desidera un oggetto grafico che visualizzi, per settimana, se i prodotti fabbricati sono "difettosi" o "senza difetti" e il costo dei prodotti fabbricati quella settimana.

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,manufacture_date,cost_price
8188,'2022-01-02 12:22:06',37.23
8189,'2022-01-05 01:02:30',17.17
8190,'2022-01-06 15:36:20',88.27
8191,'2022-01-08 10:58:35',57.42
8192,'2022-01-09 08:53:32',53.80
8193,'2022-01-10 21:13:01',82.06
8194,'2022-01-11 00:57:13',40.39
8195,'2022-01-12 09:26:02',87.21
8196,'2022-01-13 15:05:09',95.93
8197,'2022-01-14 18:44:57',45.89
8198,'2022-01-15 06:10:46',36.23
8199,'2022-01-16 06:39:27',25.66
8200,'2022-01-17 10:44:16',82.77
8201,'2022-01-18 18:48:17',69.98
8202,'2022-01-26 04:36:03',76.11
8203,'2022-01-27 08:07:49',25.12
8204,'2022-01-28 12:24:29',46.23
8205,'2022-01-30 11:56:56',84.21
8206,'2022-01-30 14:40:19',96.24
8207,'2022-01-31 05:28:21',67.67
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella. Creare una dimensione per visualizzare i nomi delle settimane:
   `=weekname(manufacture_date)`
2. Quindi, creare una dimensione per identificare quali prodotti sono difettosi e quali senza difetti:
   `=if(inWeektodate(manufacture_date,makedate(2022,01,12),0),'Defective','Faultless')`
3. Creare una misura per sommare il valore `cost_price` dei prodotti:
   `=sum(cost_price)`
4. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

weekname(manufacture_date)	if(inweektodate(manufacture_date,makedate (2022,01,12),0),'Defective','Faultless')	=sum(cost_price)
2022/02	Non difettoso	$200.09
2022/03	Difettoso	$263.46
2022/03	Non difettoso	$178.05
2022/04	Non difettoso	$178.41
2022/05	Non difettoso	$147.46
2022/06	Non difettoso	$248.12

La funzione `inweektodate()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per coloro che restituiscono il valore booleano `TRUE`, contrassegna i prodotti come `'Defective'`. Per qualsiasi prodotto che restituisce il valore `FALSE`, e quindi non realizzato nella settimana fino al 12 gennaio, contrassegna i prodotti come `'Faultless'`.

inyear

Questa funzione restituisce True se **timestamp** ricade all'interno dell'anno contenente **base_date**.

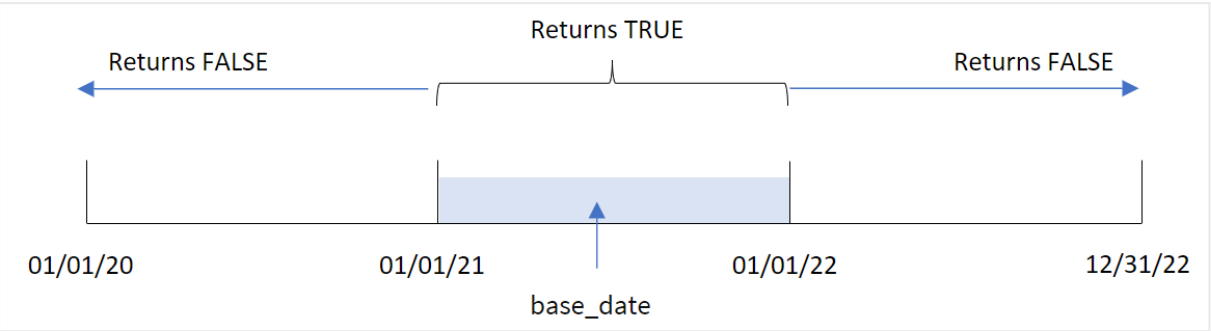
Sintassi:

```
InYear (timestamp, base_date, period_no [, first_month_of_year])
```

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Schema dell'intervallo della funzione `inyear()`



La funzione `inyear()` restituisce un risultato booleano quando si confrontano i valori della data selezionata con un anno definito dall'opzione `base_date`.

Casi di utilizzo

La funzione `inyear()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un file `if expression`. Restituisce un'aggregazione o un calcolo che dipende dal fatto che una data valutata si sia verificata nell'anno in questione. Ad esempio, la funzione `inyear()` può essere utilizzata per identificare tutte le vendite avvenute in un determinato anno.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare l'anno.
period_no	L'anno può essere differito mediante **period_no**. **period_no** è un numero intero, in cui il valore 0 indica l'anno che contiene **base_date**. I valori negativi di **period_no** indicano gli anni precedenti, mentre i valori positivi indicano gli anni successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

È possibile utilizzare i seguenti valori per impostare il primo mese dell'anno nell'argomento `first_month_of_year`:

valori first_month_of_year

Month	Valore
Febbraio	2
March	3
April	4
May	5
June	6

Month	Valore
July	7
August	8
September	9
October	10
Novembre	11
December	12

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`inyear ('01/25/2013', '01/01/2013', 0 )`	Restituisce TRUE
`inyear ('01/25/2012', '01/01/2013', 0)`	Restituisce FALSE
`inyear ('01/25/2013', '01/01/2013', -1)`	Restituisce FALSE
`inyear ('01/25/2012', '01/01/2013', -1 )`	Restituisce TRUE
`inyear ('01/25/2013', '01/01/2013', 0, 3)`	Restituisce TRUE Il valore di `base_date` e `first_month_of_year` specifica che il timestamp deve essere compreso tra il 03/01/2012 e il 28/02/2013.
`inyear ('03/25/2013', '07/01/2013', 0, 3 )`	Restituisce TRUE

Esempio 1 - Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Un caricamento precedente che contiene la funzione `inyear()` impostata come campo '`in_year`' e che determina quali transazioni hanno avuto luogo nello stesso anno del 26 luglio 2021.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
Transactions:
    Load
        *,
        inyear(date,'07/26/2021', 0) as in_year
        ;
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

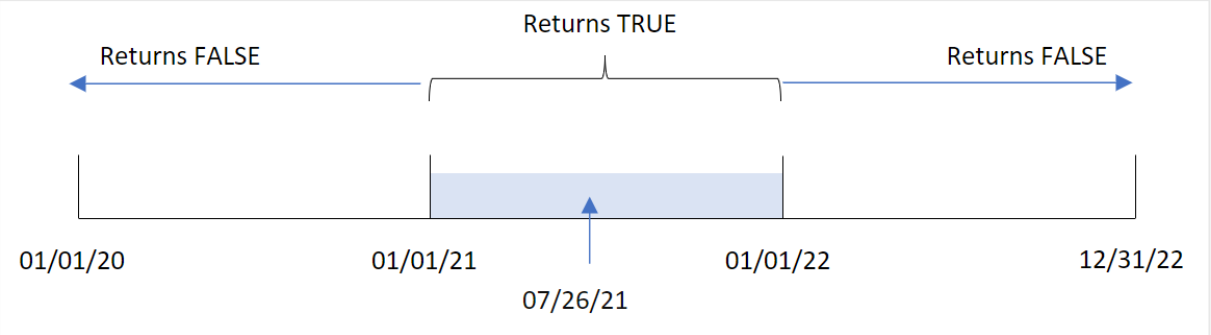
- `date`
- `in_year`

Tabella dei risultati

date	in_year
01/13/2020	0
02/26/2020	0
03/27/2020	0
04/16/2020	0
05/21/2020	0
08/14/2020	0
10/07/2020	0
12/05/2020	0
01/22/2021	-1
02/03/2021	-1
03/17/2021	-1
04/23/2021	-1
05/04/2021	-1
06/30/2021	-1
07/26/2021	-1
12/27/2021	-1
06/06/2022	0
07/18/2022	0
11/14/2022	0
12/12/2022	0

Il campo '`in_year`' viene creato nell'istruzione LOAD precedente utilizzando la funzione `inyear()`. Il primo argomento identifica il campo da valutare. Il secondo argomento è una data codificata per il 26 luglio 2021, ovvero il `base_date` che determina l'anno di confronto. Un `period_no` di `0` è l'argomento finale, il che significa che la funzione `inyear()` non confronta gli anni precedenti o successivi all'anno.

Schema dell'intervallo della funzione `inyear()` con il 26 luglio come data base



Qualsiasi transazione che si verifica nel 2021 restituisce un risultato booleano pari a TRUE.

Esempio 2 - period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Un caricamento precedente che contiene la funzione `inyear()` impostata come campo '`previous_year`' e determina quali transazioni hanno avuto luogo nell'anno prima dell'anno con 26 luglio 2021.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
Transactions:
    Load
        *,
        inyear(date,'07/26/2021', -1) as previous_year
        ;
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
```

```
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_year`

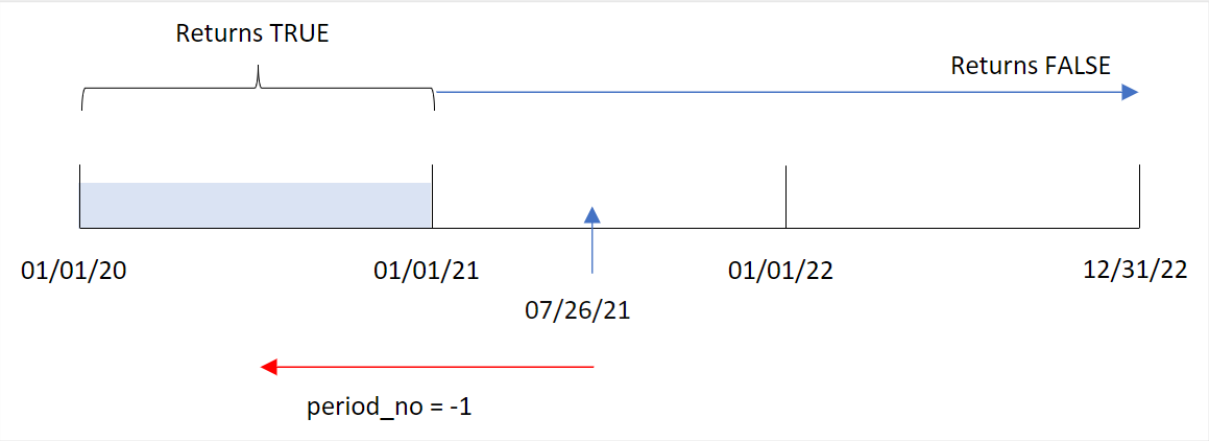
Tabella dei risultati

date	previous_year
01/13/2020	-1
02/26/2020	-1
03/27/2020	-1
04/16/2020	-1
05/21/2020	-1
08/14/2020	-1
10/07/2020	-1
12/05/2020	-1
01/22/2021	0
02/03/2021	0
03/17/2021	0
04/23/2021	0
05/04/2021	0
06/30/2021	0
07/26/2021	0
12/27/2021	0
06/06/2022	0
07/18/2022	0

date	previous_year
11/14/2022	0
12/12/2022	0

L'utilizzo di `-1` come argomento `period_no` della funzione `inyear()` sposta i confini dell'anno di confronto indietro di un intero anno. Il 2021 viene inizialmente identificato come anno di confronto. `period_no` compensa l'anno comparatore di un anno, rendendo il 2020 l'anno di confronto.

Schema dell'intervallo della funzione `inyear()` con l'argomento period_no impostato su `-1`



Pertanto, qualsiasi transazione che si verifica nel 2020 restituisce un risultato booleano di TRUE.

Esempio 3 - first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Un caricamento precedente che contiene la funzione `inyear()` impostata come campo '`in_year`' e che determina quali transazioni hanno avuto luogo nello stesso anno del 26 luglio 2021.

Tuttavia, in questo esempio, il criterio organizzativo prevede che marzo sia il primo mese dell'anno finanziario.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
Transactions:
    Load
        *,
```

```
        inyear(date,'07/26/2021', 0, 3) as in_year
        ;
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_year`

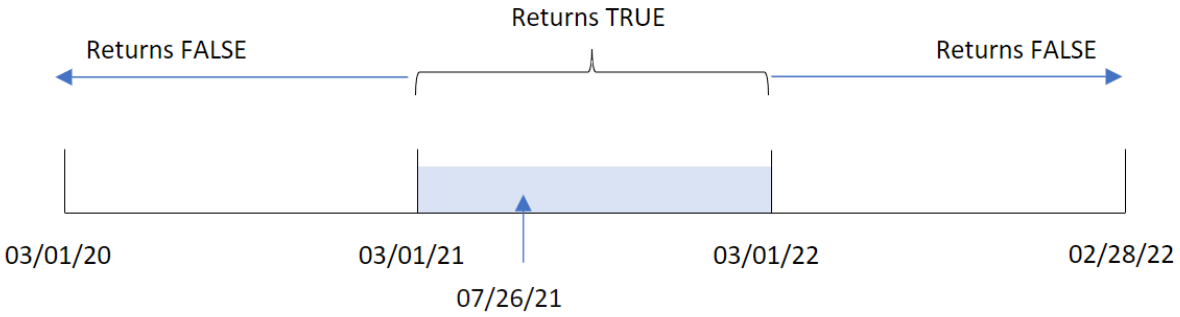
Tabella dei risultati

date	in_year
01/13/2020	0
02/26/2020	0
03/27/2020	0
04/16/2020	0
05/21/2020	0
08/14/2020	0
10/07/2020	0
12/05/2020	0

date	in_year
01/22/2021	0
02/03/2021	0
03/17/2021	-1
04/23/2021	-1
05/04/2021	-1
06/30/2021	-1
07/26/2021	-1
12/27/2021	-1
06/06/2022	0
07/18/2022	0
11/14/2022	0
12/12/2022	0

Utilizzando `3` come argomento `first_month_of_year` nella funzione `inyear()`, l'anno inizia il 1° marzo e termina alla fine di febbraio.

Schema dell'intervallo della funzione `inyear()` con marzo impostato come primo mese dell'anno



Pertanto, qualsiasi transazione che si verifichi tra il 1° marzo 2021 e il 1° marzo 2022 restituirà un risultato booleano di TRUE.

Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che determina se le transazioni sono avvenute nello stesso anno del 26 luglio 2021 viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
Transactions:
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

- `date`

Per calcolare se le transazioni sono avvenute nello stesso anno del 26 luglio 2021, creare la seguente misura:

- `=inyear(date,'07/26/2021', 0)`

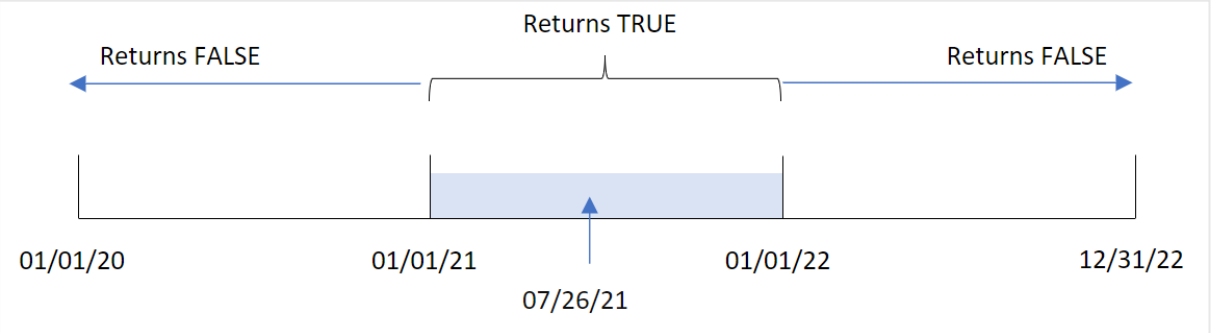
Tabella dei risultati

date	=inyear(date,'07/26/2021',0)
01/13/2020	0
02/26/2020	0

date	=inyear(date,'07/26/2021',0)
03/27/2020	0
04/16/2020	0
05/21/2020	0
08/14/2020	0
10/07/2020	0
12/05/2020	0
01/22/2021	-1
02/03/2021	-1
03/17/2021	-1
04/23/2021	-1
05/04/2021	-1
06/30/2021	-1
07/26/2021	-1
12/27/2021	-1
06/06/2022	0
07/18/2022	0
11/14/2022	0
12/12/2022	0

Il campo 'in_year' viene creato nel grafico usando la funzione `inyear()`. Il primo argomento identifica il campo da valutare. Il secondo argomento è una data codificata per il 26 luglio 2021, ovvero il `base_date` che determina l'anno di confronto. Un `period_no` di `0` è l'argomento finale, il che significa che la funzione `inyear()` non confronta gli anni precedenti o successivi all'anno.

Schema dell'intervallo della funzione `inyear()` con il 27 luglio come data base



Qualsiasi transazione che si verifica nel 2021 restituisce un risultato booleano pari a TRUE.

Esempio 5 - Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata '`Products`'.
- La tabella contiene i seguenti campi:
    - ID prodotto
    - tipo di prodotto
    - data di produzione
    - prezzo di costo

L'utente finale vuole un oggetto grafico che mostri, per tipo di prodotto, il costo dei prodotti fabbricati nel 2021.

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,product_type,manufacture_date,cost_price
8188,product A,'01/13/2020',37.23
8189,product B,'02/26/2020',17.17
8190,product B,'03/27/2020',88.27
8191,product C,'04/16/2020',57.42
8192,product D,'05/21/2020',53.80
8193,product D,'08/14/2020',82.06
8194,product C,'10/07/2020',40.39
8195,product B,'12/05/2020',87.21
8196,product A,'01/22/2021',95.93
8197,product B,'02/03/2021',45.89
8198,product C,'03/17/2021',36.23
8199,product C,'04/23/2021',25.66
8200,product B,'05/04/2021',82.77
8201,product D,'06/30/2021',69.98
8202,product D,'07/26/2021',76.11
8203,product D,'12/27/2021',25.12
8204,product C,'06/06/2022',46.23
8205,product C,'07/18/2022',84.21
8206,product A,'11/14/2022',96.24
8207,product B,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

- `product_type`

Per calcolare la somma di ogni prodotto fabbricato nel 2021, creare la seguente misura:

- `=sum(if(InYear(manufacture_date,makedate(2021,01,01),0),cost_price,0))`

Procedere come indicato di seguito:

1. Impostare la misura **Number Formatting** su **Money**.
2. In **Aspetto**, disattivare **Totals**.

Tabella dei risultati

product_type	=sum(if(InYear(manufacture_date,makedate(2021,01,01),0),cost_price,0))
prodotto A	$95.93
prodotto B	$128.66
prodotto C	$61.89
prodotto D	$171.21

La funzione `inyear()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per qualsiasi prodotto fabbricato nel 2021, la funzione `inyear()` restituisce il valore booleano TRUE e mostra la somma di `cost_price`.

inyeartodate

Questa funzione restituisce True se **timestamp** ricade all'interno della parte dell'anno contenente **base_date** fino a includere l'ultimo millisecondo di **base_date**.

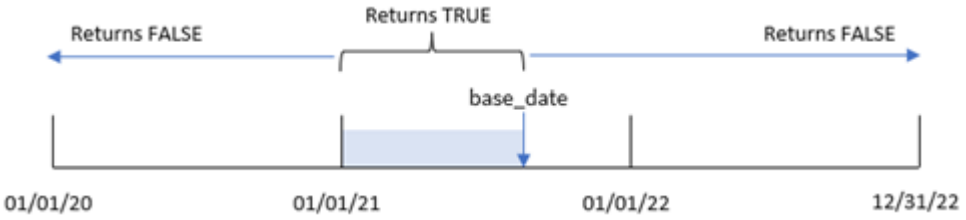
Sintassi:

```
InYearToDate (timestamp, base_date, period_no[, first_month_of_year])
```

Tipo di dati restituiti: Booleano

> *In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.*

Diagramma della funzione inyeartodate



La funzione `inyeartodate()` segmenterà una particolare parte dell'anno con `base_date`, identificando la data massima consentita per quel segmento di anno. La funzione valuta quindi se un campo data o un valore rientrano in tale segmento e restituisce un risultato booleano.

Argomenti

Argomento	Descrizione
timestamp	La data da confrontare con **base_date**.
base_date	La data utilizzata per valutare l'anno.
period_no	L'anno può essere differito mediante **period_no**. **period_no** è un numero intero, in cui il valore 0 indica l'anno che contiene **base_date**. I valori negativi di **period_no** indicano gli anni precedenti, mentre i valori positivi indicano gli anni successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

Casi di utilizzo

La funzione `inyeartodate()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un'espressione `if`. Questa restituisce un'aggregazione o un calcolo che dipende dal fatto che una data valutata si sia verificata in quell'anno fino a includere la data in questione.

Ad esempio, la funzione `inyeartodate()` può essere utilizzata per identificare tutte le apparecchiature prodotte in un in un determinato anno fino a una data specifica.

In questi esempi viene utilizzato il formato della data (MM/GG/AAAA). Il formato della data viene specificato nell'istruzione `SET DateFormat` nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di funzioni

Esempio	Risultato
`inyeartodate ('01/25/2013', '02/01/2013', 0)`	Restituisce `TRUE`.
`inyeartodate ('01/25/2012', '01/01/2013', 0)`	Restituisce `FALSE`.

Esempio	Risultato
`inyeartodate ('01/25/2012', '02/01/2013', -1)`	Restituisce `TRUE`.
`inyeartodate ('11/25/2012', '01/31/2013', 0, 4)`	Restituisce `TRUE`. Il valore di `timestamp` ricade nell'anno fiscale che inizia nel quarto mese e prima del valore di `base_date`.
`inyeartodate ('3/31/2013', '01/31/2013', 0, 4 )`	Restituisce `FALSE`. Rispetto all'esempio precedente, il valore di `timestamp` ricade ancora nell'anno fiscale, ma viene dopo il valore di`base_date`, pertanto ricade fuori dalla parte dell'anno.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `in_year_to_date`, che determina quali transazioni sono avvenute durante l'anno fino al 26 luglio 2021.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
```

```
Transactions:
    Load
        *,
        inyeartodate(date,'07/26/2021', 0) as in_year_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'06/14/2020',82.06
8194,'08/07/2020',40.39
8195,'09/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'07/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_year_to_date`

Tabella dei risultati

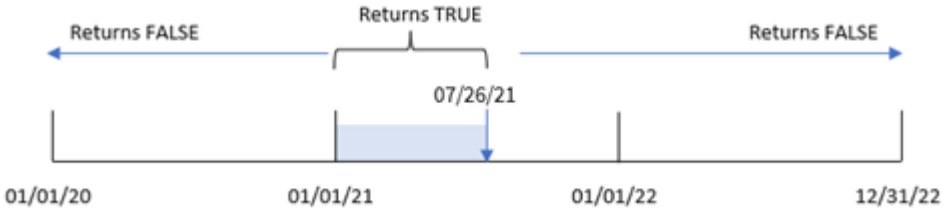
data	in_year_to_date
01/13/2020	0
02/26/2020	0
03/27/2020	0
04/16/2020	0
05/21/2020	0
06/14/2020	0

data	in_year_to_date
08/07/2020	0
09/05/2020	0
01/22/2021	-1
02/03/2021	-1
03/17/2021	-1
04/23/2021	-1
05/04/2021	-1
06/30/2021	-1
07/26/2021	-1
07/27/2021	0
06/06/2022	0
07/18/2022	0
11/14/2022	0
12/12/2022	0

Il campo `in_year_to_date` viene creato nell'istruzione LOAD precedente utilizzando la funzione `inyeartodate()`. Il primo argomento fornito identifica il campo da valutare.

Il secondo argomento è una data codificata per il 26 luglio 2021, che è il valore `base_date` che identifica il limite finale del segmento di anno. Un valore `period_no` uguale a 0 è l'argomento finale, il che significa che la funzione non confronta gli anni precedenti o successivi all'anno segmentato.

Diagramma della funzione inyeartodate, senza argomenti aggiuntivi



Qualsiasi transazione avvenuta tra il 1 gennaio e il 26 luglio restituisce il risultato booleano TRUE. Le date delle transazioni anteriori al 2021 e successive al 26 luglio 2021 restituiscono il valore FALSE.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_year_to_date`, che determina quali transazioni sono avvenute un anno prima del segmento di anno che termina il 26 luglio 2021.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inyeartodate(date,'07/26/2021', -1) as previous_year_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'06/14/2020',82.06
8194,'08/07/2020',40.39
8195,'09/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'07/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

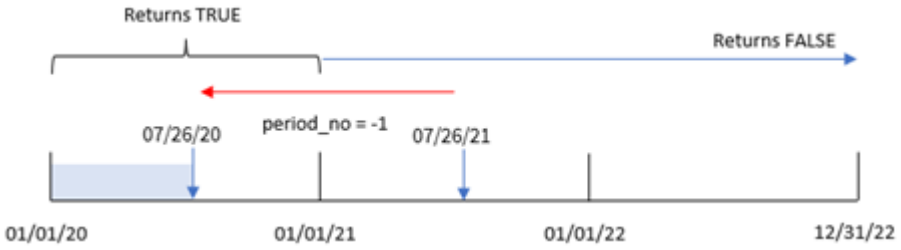
- `date`
- `previous_year_to_date`

Tabella dei risultati

data	previous_year_to_date
01/13/2020	-1
02/26/2020	-1
03/27/2020	-1
04/16/2020	-1
05/21/2020	-1
06/14/2020	-1
08/07/2020	0
09/05/2020	0
01/22/2021	0
02/03/2021	0
03/17/2021	0
04/23/2021	0
05/04/2021	0
06/30/2021	0
07/26/2021	0
07/27/2021	0
06/06/2022	0
07/18/2022	0
11/14/2022	0
12/12/2022	0

Il valore `period_no` di -1 indica che la funzione `inyeartodate ()` confronta il segmento del trimestre di input con l'anno precedente. Con la data di input del 26 luglio 2021, il segmento dal 1 gennaio 2021 al 26 luglio 2021 è stato inizialmente identificato come year-to-date. `period_no` quindi compensa questo segmento facendolo precedere di un intero anno, facendo sì che i limiti della data diventino dal 1 gennaio al 26 luglio 2020.

Diagramma della funzione inyeartodate, esempio di period_no



Pertanto, qualsiasi transazione che si verifichi tra il 1 gennaio e il 26 luglio 2020 restituirà il risultato booleano `TRUE`.

Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `in_year_to_date`, che determina quali transazioni sono avvenute nello stesso anno fino al 26 luglio 2021.

In questo esempio, marzo è stato impostato come primo mese dell'anno fiscale.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        inyeartodate(date,'07/26/2021', 0,3) as in_year_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'06/14/2020',82.06
8194,'08/07/2020',40.39
8195,'09/05/2020',87.21
8196,'01/22/2021',95.93
```

```
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'07/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `in_year_to_date`

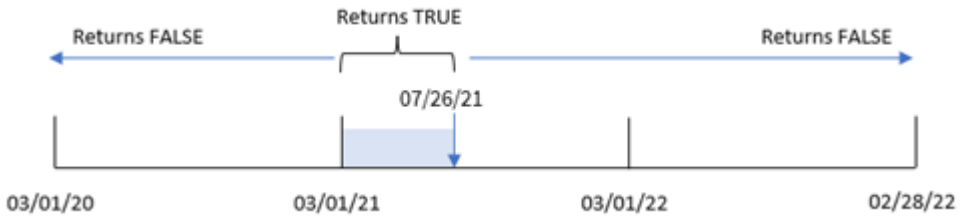
Tabella dei risultati

data	in_year_to_date
01/13/2020	0
02/26/2020	0
03/27/2020	0
04/16/2020	0
05/21/2020	0
06/14/2020	0
08/07/2020	0
09/05/2020	0
01/22/2021	0
02/03/2021	0
03/17/2021	-1
04/23/2021	-1
05/04/2021	-1
06/30/2021	-1
07/26/2021	-1
07/27/2021	0
06/06/2022	0

data	in_year_to_date
07/18/2022	0
11/14/2022	0
12/12/2022	0

Utilizzando 3 come argomento `first_month_of_year` nella funzione `inyeartodate()`, la funzione fa iniziare l'anno il 1 marzo. Il valore `base_date` del 26 luglio 2021 imposta quindi la data di fine per quel segmento di anno.

Diagramma della funzione inyeartodate, esempio di first_month_of_year



Pertanto, qualsiasi transazione che si verifichi tra il 1 marzo e il 26 luglio 2001 restituirà un risultato booleano `TRUE`, mentre le transazioni con una data che non rientra in questi limiti restituiranno il valore `FALSE`.

Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che determina quali transazioni sono avvenute nello stesso anno fino al 26 luglio 2021 viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
```

```
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'06/14/2020',82.06
8194,'08/07/2020',40.39
8195,'09/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'07/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`date`.

Creare la seguente misura:

```
=inyeartodate(date,'07/26/2021', 0)
```

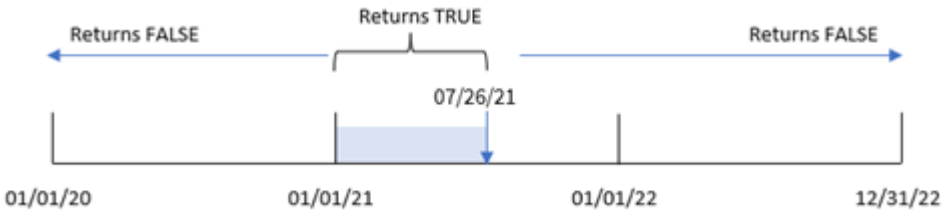
Tabella dei risultati

data	=inyeartodate(date,'07/26/2021', 0)
01/13/2020	0
02/26/2020	0
03/27/2020	0
04/16/2020	0
05/21/2020	0
06/14/2020	0
08/07/2020	0
09/05/2020	0
01/22/2021	-1
02/03/2021	-1
03/17/2021	-1
04/23/2021	-1

data	=inyeartodate(date,'07/26/2021', 0)
05/04/2021	-1
06/30/2021	-1
07/26/2021	-1
07/27/2021	0
06/06/2022	0
07/18/2022	0
11/14/2022	0
12/12/2022	0

La misura `in_year_to_date` viene creata nell'oggetto grafico utilizzando la funzione `inyeartodate()`. Il primo argomento fornito identifica il campo da valutare. Il secondo argomento è una data codificata per il 26 luglio 2021, che è il valore `base_date` che identifica il limite finale del segmento di anno di confronto. Un valore `period_no` uguale a 0 è l'argomento finale, il che significa che la funzione non confronta gli anni precedenti o successivi all'anno segmentato.

Diagramma della funzione inyeartodate, esempio di oggetto grafico



Qualsiasi transazione avvenuta tra il 1 gennaio e il 26 luglio 2021 restituisce il risultato booleano `TRUE`. Le date delle transazioni anteriori al 2021 e successive al 26 luglio 2021 restituiscono il valore `FALSE`.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Products`.
- Informazioni relative all'ID prodotto, al tipo di prodotto, alla data di produzione e al prezzo di costo.

L'utente finale vuole un oggetto grafico che mostri, per tipo di prodotto, il costo dei prodotti fabbricati nel 2021 fino al 26 luglio.

Script di caricamento

```
Products:
Load
*
Inline
[
product_id,product_type,manufacture_date,cost_price
8188,product A,'01/13/2020',37.23
8189,product B,'02/26/2020',17.17
8190,product B,'03/27/2020',88.27
8191,product C,'04/16/2020',57.42
8192,product D,'05/21/2020',53.80
8193,product D,'08/14/2020',82.06
8194,product C,'10/07/2020',40.39
8195,product B,'12/05/2020',87.21
8196,product A,'01/22/2021',95.93
8197,product B,'02/03/2021',45.89
8198,product C,'03/17/2021',36.23
8199,product C,'04/23/2021',25.66
8200,product B,'05/04/2021',82.77
8201,product D,'06/30/2021',69.98
8202,product D,'07/26/2021',76.11
8203,product D,'12/27/2021',25.12
8204,product C,'06/06/2022',46.23
8205,product C,'07/18/2022',84.21
8206,product A,'11/14/2022',96.24
8207,product B,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`product_type`.

Creare una misura che calcoli la somma di ogni prodotto che è stato prodotto nel 2021 prima del 27 luglio:

```
=sum(if(inyeartodate(manufacture_date,makedate(2021,07,26),0),cost_price,0))
```

Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

product_type	=sum(if(inyeartodate(manufacture_date,makedate (2021,07,26),0),cost_price,0))
prodotto A	$95.93
prodotto B	$128.66
prodotto C	$61.89
prodotto D	$146.09

La funzione `inyeartodate()` restituisce un valore booleano quando valuta le date di produzione di ciascun prodotto. Per qualsiasi prodotto fabbricato nel 2021 prima del 27 luglio, la funzione `inyeartodate()` restituisce un valore booleano `TRUE` e somma il valore `cost_price`.

Il prodotto D è l'unico prodotto fabbricato anche dopo il 26 luglio 2021. La voce con `product_ID` 8203 è stata prodotta il 27 dicembre e costa $25.12. Pertanto, questo costo non è stato incluso nel totale del Prodotto D nell'oggetto grafico.

lastworkdate

La funzione **lastworkdate** restituisce la data di fine più prossima per ottenere **no_of_ workdays** (dal lunedì al venerdì) se si inizia dalla data **start_date** tenendo in considerazione tutte le festività **holiday** eventualmente in calendario. **start_date** e **holiday** devono essere date o indicatori temporali validi.

Sintassi:

```
lastworkdate(start_date, no_of_workdays {, holiday})
```

Tipo di dati restituiti: numero intero

Un calendario che mostra come viene utilizzata la funzione `lastworkdate()`

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10 start_date	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26 end_date	27
28	29	30	31			

Limitazioni

Non esiste un metodo per modificare la funzione `lastworkdate()` per le regioni o gli scenari che prevedono qualcosa di diverso da una settimana lavorativa che inizia il lunedì e termina il venerdì.

Il parametro vacanza deve essere una costante stringa. Non accetta un'espressione.

Casi di utilizzo

La funzione `lastworkdate()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera calcolare la data di fine proposta di un progetto o di un incarico, in base alla data di inizio del progetto e alle vacanze che si verificheranno in quel periodo.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Argomenti

Argomento	Descrizione
start_date	La data di inizio da valutare.
no_of_ workdays	Il numero di giorni lavorativi da raggiungere.
holiday	Periodi di vacanza da escludere dai giorni lavorativi. Una vacanza è definita come una stringa di data costante. È possibile specificare più periodi di vacanza, separati da virgole. **Esempio:** `'12/25/2013', '12/26/2013', '12/31/2013', '01/01/2014'`

Esempio 1 - Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente gli ID dei progetti, le date di inizio e l'impegno stimato, in giorni, richiesto per i progetti. Il set di dati viene caricato in una tabella chiamata '`Projects`'.

- Un caricamento precedente che contiene la funzione `lastworkdate()` impostata come campo '`end_date`' e che identifica quando è prevista la fine di ogni progetto.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Projects:
    Load
       *,
       LastWorkDate(start_date,effort) as end_date
       ;
Load
id,
start_date,
effort
Inline
[
id,start_date,effort
1,01/01/2022,14
2,02/10/2022,17
3,05/17/2022,5
4,06/01/2022,12
5,08/10/2022,26
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `start_date`
- `effort`
- `end_date`

Tabella dei risultati

id	start_date	effort	end_date
1	01/01/2022	14	01/20/2022
2	02/10/2022	17	03/04/2022
3	05/17/2022	5	05/23/2022
4	06/01/2022	12	06/16/2022
5	08/10/2022	26	09/14/2022

Poiché non ci sono giorni festivi programmati, la funzione aggiunge il numero definito di giorni lavorativi, dal lunedì al venerdì, alla data di inizio per trovare la prima data di fine possibile.

Il calendario seguente mostra la data di inizio e di fine del progetto 3, con i giorni lavorativi evidenziati in verde.

Un calendario che mostra la data di inizio e di fine del progetto 3

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17 Start Date	18	19	20	21
22	23 End Date	24	25	26	27	28
29	30	31				

Esempio 2 - Singolo giorno festivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente gli ID dei progetti, le date di inizio e l'impegno stimato, in giorni, richiesto per i progetti. Il set di dati viene caricato in una tabella chiamata '`Projects`'.
- Un caricamento precedente che contiene la funzione `lastworkdate()` impostata come campo '`end_date`' e che identifica quando è prevista la fine di ogni progetto.

Tuttavia, è prevista una festività il 18 maggio 2022. La funzione `lastworkdate()` nel caricamento precedente include il giorno festivo nel suo terzo argomento per identificare quando è prevista la fine di ogni progetto.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Projects:
    Load
       *,
       LastWorkDate(start_date,effort, '05/18/2022') as end_date
       ;
Load
id,
start_date,
effort
Inline
[
id,start_date,effort
1,01/01/2022,14
2,02/10/2022,17
3,05/17/2022,5
4,06/01/2022,12
5,08/10/2022,26
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `start_date`
- `effort`
- `end_date`

Tabella dei risultati

id	start_date	effort	end_date
1	01/01/2022	14	01/20/2022
2	02/10/2022	17	03/04/2022
3	05/17/2022	5	05/24/2022
4	06/01/2022	12	06/16/2022
5	08/10/2022	26	09/14/2022

La singola vacanza programmata viene inserita come terzo argomento della funzione `lastworkdate()`. Di conseguenza, la data di fine del progetto 3 viene spostata di un giorno perché la vacanza si svolge in uno dei giorni lavorativi precedenti la data di fine.

Il calendario seguente mostra la data di inizio e di fine del progetto 3 e mostra che la vacanza modifica la data di fine del progetto di un giorno.

Un calendario che mostra la data di inizio e fine del progetto 3 con una vacanza il 18 maggio.

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17 Start Date	18 Holiday	19	20	21
22	23	24 End Date	25	26	27	28
29	30	31				

Esempio 3 - Vacanze multiple

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente gli ID dei progetti, le date di inizio e l'impegno stimato, in giorni, richiesto per i progetti. Il set di dati viene caricato in una tabella chiamata '`Projects`'.
- Un caricamento precedente che contiene la funzione `lastworkdate()` impostata come campo '`end_date`' e che identifica quando è prevista la fine di ogni progetto.

Tuttavia, sono previste quattro festività: il 19, 20, 21 e 22 maggio. La funzione `lastworkdate()` nel caricamento precedente include ciascuno dei giorni festivi nel suo terzo argomento per identificare quando è prevista la fine di ogni progetto.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Projects:
    Load
       *,
       LastWorkDate(start_date,effort, '05/19/2022','05/20/2022','05/21/2022','05/22/2022') as
end_date
       ;
Load
id,
start_date,
effort
Inline
[
id,start_date,effort
1,01/01/2022,14
2,02/10/2022,17
3,05/17/2022,5
4,06/01/2022,12
5,08/10/2022,26
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `start_date`
- `effort`
- `end_date`

Tabella dei risultati

id	start_date	effort	end_date
1	01/01/2022	14	01/20/2022
2	02/10/2022	17	03/04/2022
3	05/17/2022	5	05/25/2022
4	06/01/2022	12	06/16/2022
5	08/10/2022	26	09/14/2022

I quattro giorni festivi vengono inseriti come elenco di argomenti nella funzione `lastworkdate()` dopo la data di inizio e il numero di giorni lavorativi.

Il calendario seguente mostra la data di inizio e di fine del progetto 3 e mostra che le vacanze modificano la data di fine del progetto di tre giorni.

Un calendario che mostra la data di inizio e fine del progetto 3 con le vacanze dal 19 al 22 maggio.

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17 Start Date	18	19 Holiday	20 Holiday	21 Holiday
22 Holiday	23	24	25 End Date	26	27	28
29	30	31				

Esempio 4 - Vacanza singola (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e caricato nell'app. Il campo `end_date` viene calcolato come misura in un grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Projects:
Load
id,
start_date,
```

```
effort
Inline
[
id,start_date,effort
1,01/01/2022,14
2,02/10/2022,17
3,05/17/2022,5
4,06/01/2022,12
5,08/10/2022,26
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `start_date`
- `effort`

Per calcolare la end_date, creare la seguente misura:

- `=LastWorkDate(start_date,effort,'05/18/2022')`

Tabella dei risultati

id	start_date	effort	=LastWorkDate(start_date,effort,'05/18/2022')
1	01/01/2022	14	01/20/2022
2	02/10/2022	17	03/04/2022
3	05/17/2022	5	05/23/2022
4	06/01/2022	12	06/16/2022
5	08/10/2022	26	09/14/2022

La singola festività programmata viene inserita come misura nel grafico. Di conseguenza, la data di fine del progetto 3 viene spostata di un giorno perché la vacanza si svolge in uno dei giorni lavorativi precedenti la data di fine.

Il calendario seguente mostra la data di inizio e di fine del progetto 3 e mostra che la vacanza modifica la data di fine del progetto di un giorno.

Un calendario che mostra la data di inizio e fine del progetto 3 con una vacanza il 18 maggio.

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17 Start Date	18 Holiday	19	20	21
22	23	24 End Date	25	26	27	28
29	30	31				

localtime

Questa funzione restituisce un indicatore temporale dell'ora attuale fornita per il fuso orario specificato.

Sintassi:

```
LocalTime([timezone [, ignoreDST ]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
timezone	Il valore **timezone** viene specificato come stringa contenente una qualsiasi delle località geografiche elencate in **Time Zone** nel **Windows Control Panel** per **Date and Time** o come stringa nel formato 'GMT+hh:mm'. Se non è specificato alcun fuso orario, viene restituita l'ora locale.
ignoreDST	Se **ignoreDST** è -1(True) il passaggio automatico all'ora legale verrà ignorato.

Esempi e risultati:

Gli esempi riportati di seguito sono basati sulla funzione chiamata il 2014-10-22 12:54:47 ora locale e con il fuso orario locale GMT+01:00.

Esempi di script

Esempio	Risultato
`localtime ()`	Restituisce l'ora locale 2014-10-22 12:54:47.
`localtime ('London')`	Restituisce l'ora locale di Londra, 2014-10-22 11:54:47.
`localtime ('GMT+02:00')`	Restituisce l'ora locale nel fuso orario di GMT+02:00, 2014-10-22 13:54:47.
`localtime ('Paris','-1')`	Restituisce l'ora locale di Parigi e l'ora legale viene ignorata, 2014-10-22 11:54:47.

lunarweekend

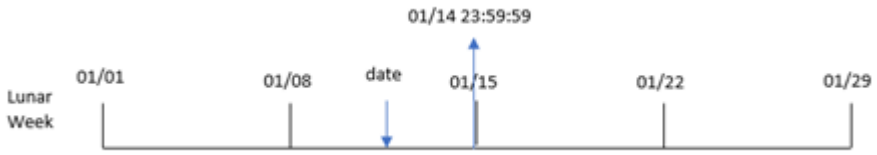
Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo dell'ultimo giorno della settimana lunare contenente **date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana e, a parte l'ultima settimana dell'anno, conterranno esattamente sette giorni.

Sintassi:

```
LunarweekEnd(date[, period_no[, first_week_day]])
```

Tipo di dati restituiti: duale

Schema esemplificativo della funzione `lunarweekend()`



La funzione `lunarweekend()` determina la settimana lunare in cui cade `date`. Quindi, restituisce un timestamp, nel formato data, per l'ultimo millisecondo di quella settimana.

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica la settimana lunare che contiene il valore **date**. I valori negativi di **period_no** indicano le settimane lunari precedenti, mentre i valori positivi indicano le settimane lunari successive.
first_week_day	Un differimento che può essere maggiore o minore di zero. Ciò modifica l'inizio dell'anno in base al numero specificato di giorni e/o frazioni di un giorno.

Casi di utilizzo

La funzione `lunarweekend()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione di settimana non ancora trascorsa. A differenza della funzione `weekend()`, l'ultima settimana lunare di ogni anno solare terminerà il 31 dicembre. Ad esempio, la funzione `lunarweekend()` può essere utilizzata per calcolare gli interessi non ancora maturati durante la settimana.

Esempi di funzioni

Esempio	Risultato
`lunarweekend('01/12/2013')`	Restituisce 01/14/2013 23:59:59.
`lunarweekend('01/12/2013', -1)`	Restituisce 01/07/2013 23:59:59.
`lunarweekend('01/12/2013', 0, 1)`	Restituisce 01/15/2013 23:59:59.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `end_of_week`, che restituisce un timestamp per la fine della settimana lunare in cui sono avvenute le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        lunarweekend(date) as end_of_week,
        timestamp(lunarweekend(date)) as end_of_week_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
```

```
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `end_of_week`
- `end_of_week_timestamp`

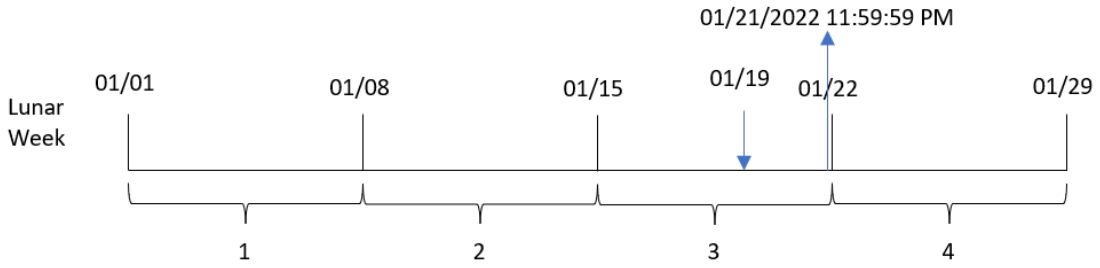
Tabella dei risultati

date	end_of_week	end_of_week_timestamp
1/7/2022	01/07/2022	1/7/2022 11:59:59 PM
1/19/2022	01/21/2022	1/21/2022 11:59:59 PM
2/5/2022	02/11/2022	2/11/2022 11:59:59 PM
2/28/2022	03/04/2022	3/4/2022 11:59:59 PM
3/16/2022	03/18/2022	3/18/2022 11:59:59 PM
4/1/2022	04/01/2022	4/1/2022 11:59:59 PM
5/7/2022	05/13/2022	5/13/2022 11:59:59 PM
5/16/2022	05/20/2022	5/20/2022 11:59:59 PM
6/15/2022	06/17/2022	6/17/2022 11:59:59 PM
6/26/2022	07/01/2022	7/1/2022 11:59:59 PM
7/9/2022	07/15/2022	7/15/2022 11:59:59 PM
7/22/2022	07/22/2022	7/22/2022 11:59:59 PM
7/23/2022	07/29/2022	7/29/2022 11:59:59 PM
7/27/2022	07/29/2022	7/29/2022 11:59:59 PM
8/2/2022	08/05/2022	8/5/2022 11:59:59 PM
8/8/2022	08/12/2022	8/12/2022 11:59:59 PM
8/19/2022	08/19/2022	8/19/2022 11:59:59 PM
9/26/2022	09/30/2022	9/30/2022 11:59:59 PM
10/14/2022	10/14/2022	10/14/2022 11:59:59 PM
10/29/2022	11/04/2022	11/4/2022 11:59:59 PM

Il campo `end_of_week` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `lunarweekend()` e trasferendo il campo `date` come argomento della funzione.

La funzione `lunarweekend()` identifica in quale settimana lunare cade il valore della data, restituendo un timestamp per l'ultimo millisecondo di quella settimana.

Schema della funzione `lunarweekend()`, esempio senza argomenti aggiuntivi



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekend()` identifica che la settimana lunare inizia il 15 gennaio. Pertanto, il valore `end_of_week` per questa transazione restituisce l'ultimo millisecondo della settimana lunare, ovvero il 21 gennaio alle 23:59:59.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_lunar_week_end`, che restituisce il timestamp per la fine della settimana lunare prima che avvenga la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        lunarweekend(date,-1) as previous_lunar_week_end,
        timestamp(lunarweekend(date,-1)) as previous_lunar_week_end_timestamp
        ;
Load
*
Inline
[
id,date,amount
```

```
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_lunar_week_end`
- `previous_lunar_week_end_timestamp`

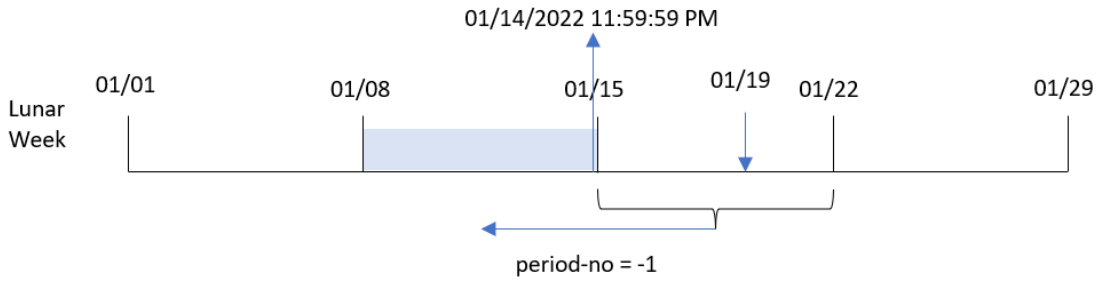
Tabella dei risultati

date	previous_lunar_week_end	previous_lunar_week_end_timestamp
1/7/2022	12/31/2021	12/31/2021 11:59:59 PM
1/19/2022	01/14/2022	14/01/2022 23:59:59
2/5/2022	02/04/2022	2/4/2022 11:59:59 PM
2/28/2022	02/25/2022	2/25/2022 11:59:59 PM
3/16/2022	03/11/2022	3/18/2022 11:59:59 PM
4/1/2022	03/25/2022	3/25/2022 11:59:59 PM
5/7/2022	05/06/2022	5/6/2022 11:59:59 PM
5/16/2022	05/13/2022	5/13/2022 11:59:59 PM
6/15/2022	06/10/2022	6/10/2022 11:59:59 PM
6/26/2022	06/24/2022	6/24/2022 11:59:59 PM
7/9/2022	07/08/2022	7/8/2022 11:59:59 PM

date	previous_lunar_week_end	previous_lunar_week_end_timestamp
7/22/2022	07/15/2022	7/15/2022 11:59:59 PM
7/23/2022	07/22/2022	7/22/2022 11:59:59 PM
7/27/2022	07/22/2022	7/22/2022 11:59:59 PM
8/2/2022	07/29/2022	7/29/2022 11:59:59 PM
8/8/2022	08/05/2022	8/5/2022 11:59:59 PM
8/19/2022	08/12/2022	8/12/2022 11:59:59 PM
9/26/2022	09/23/2022	9/23/2022 11:59:59 PM
10/14/2022	10/07/2022	10/7/2022 11:59:59 PM
10/29/2022	10/28/2022	10/28/2022 11:59:59 PM

In questo caso, poiché è stato utilizzato un valore `period_no` di -1 come argomento di offset nella funzione `lunarweekend()`, la funzione identifica innanzitutto la settimana lunare in cui sono avvenute le transazioni. Si sposta poi una settimana prima e identifica il millisecondo finale di quella settimana lunare.

Schema della funzione `lunarweekend()`, esempio di period_no



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekend()` identifica che la settimana lunare inizia il 15 gennaio. Pertanto, la settimana lunare precedente è iniziata l'8 gennaio ed è terminata il 14 gennaio alle 23:59:59; questo è il valore restituito per il campo `previous_lunar_week_end`.

Esempio 3 – first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. In questo esempio, abbiamo impostato che le settimane lunari inizino il 5 gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        lunarweekend(date,0,4) as end_of_week,
timestamp(lunarweekend(date,0,4)) as end_of_week_timestamp
;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `end_of_week`
- `end_of_week_timestamp`

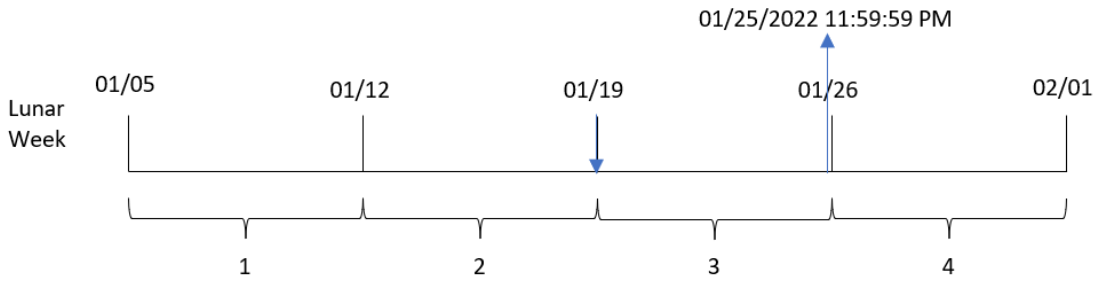
Tabella dei risultati

date	end_of_week	end_of_week_timestamp
1/7/2022	01/11/2022	1/11/2022 11:59:59 PM
1/19/2022	01/25/2022	1/25/2022 11:59:59 PM

date	end_of_week	end_of_week_timestamp
2/5/2022	02/08/2022	2/8/2022 11:59:59 PM
2/28/2022	03/01/2022	3/1/2022 11:59:59 PM
3/16/2022	03/22/2022	3/22/2022 11:59:59 PM
4/1/2022	04/05/2022	4/5/2022 11:59:59 PM
5/7/2022	05/10/2022	5/10/2022 11:59:59 PM
5/16/2022	05/17/2022	5/17/2022 11:59:59 PM
6/15/2022	06/21/2022	6/21/2022 11:59:59 PM
6/26/2022	06/28/2022	6/28/2022 11:59:59 PM
7/9/2022	07/12/2022	7/12/2022 11:59:59 PM
7/22/2022	07/26/2022	7/26/2022 11:59:59 PM
7/23/2022	07/26/2022	7/26/2022 11:59:59 PM
7/27/2022	08/02/2022	8/2/2022 11:59:59 PM
8/2/2022	08/02/2022	8/2/2022 11:59:59 PM
8/8/2022	08/09/2022	8/9/2022 11:59:59 PM
8/19/2022	08/23/2022	8/23/2022 11:59:59 PM
9/26/2022	09/27/2022	9/27/2022 11:59:59 PM
10/14/2022	10/18/2022	10/18/2022 11:59:59 PM
10/29/2022	11/01/2022	11/1/2022 11:59:59 PM

In questo caso, poiché nella funzione `lunarweekend()` viene utilizzato l'argomento `first_week_date` di 4, l'inizio dell'anno viene spostato dal 1° al 5 gennaio.

Schema della funzione `lunarweekend()`, esempio first_week_day



La transazione 8189 è avvenuta il 19 gennaio. Poiché le settimane lunari iniziano il 5 gennaio, la funzione `lunarweekend()` identifica che la settimana lunare contenente il 19 gennaio inizia anch'essa il 19 gennaio. Pertanto, la fine di quella settimana lunare avviene il 25 gennaio alle 23:59:59; questo è il valore restituito per il campo `end_of_week`.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce un timestamp per la fine della settimana lunare in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Aggiungere le seguenti misure:

```
=lunarweekend(date)
```

```
=timestamp(lunarweekend(date))
```

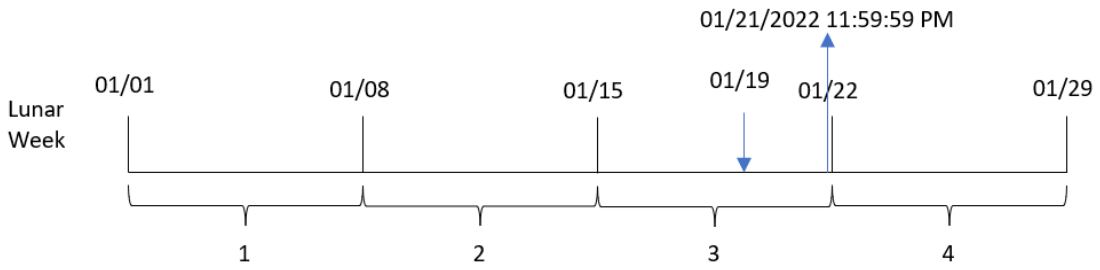
Tabella dei risultati

date	=lunarweekend(date)	=timestamp(lunarweekend(date))
1/7/2022	01/07/2022	1/7/2022 11:59:59 PM
1/19/2022	01/21/2022	1/21/2022 11:59:59 PM
2/5/2022	02/11/2022	2/11/2022 11:59:59 PM
2/28/2022	03/04/2022	3/4/2022 11:59:59 PM
3/16/2022	03/18/2022	3/18/2022 11:59:59 PM
4/1/2022	04/01/2022	4/1/2022 11:59:59 PM
5/7/2022	05/13/2022	5/13/2022 11:59:59 PM
5/16/2022	05/20/2022	5/20/2022 11:59:59 PM
6/15/2022	06/17/2022	6/17/2022 11:59:59 PM
6/26/2022	07/01/2022	7/1/2022 11:59:59 PM
7/9/2022	07/15/2022	7/15/2022 11:59:59 PM
7/22/2022	07/22/2022	7/22/2022 11:59:59 PM
7/23/2022	07/29/2022	7/29/2022 11:59:59 PM
7/27/2022	07/29/2022	7/29/2022 11:59:59 PM
8/2/2022	08/05/2022	8/5/2022 11:59:59 PM
8/8/2022	08/12/2022	8/12/2022 11:59:59 PM
8/19/2022	08/19/2022	8/19/2022 11:59:59 PM
9/26/2022	09/30/2022	9/30/2022 11:59:59 PM
10/14/2022	10/14/2022	10/14/2022 11:59:59 PM
10/29/2022	11/04/2022	11/4/2022 11:59:59 PM

La misura `end_of_week` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `lunarweekend()` e trasferendo il campo `date` come argomento della funzione.

La funzione `lunarweekend()` identifica in quale settimana lunare cade il valore della data, restituendo un timestamp per l'ultimo millisecondo di quella settimana.

Schema della funzione `lunarweekend()`, esempio di oggetto grafico



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekend()` identifica che la settimana lunare inizia il 15 gennaio. Pertanto, il valore `end_of_week` per questa transazione restituisce l'ultimo millisecondo della settimana lunare, ovvero il 21 gennaio alle 23:59:59.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Employee_Expenses`.
- Gli ID dei dipendenti, il nome del dipendente e le spese medie giornaliere dichiarate da ciascun dipendente.

L'utente finale desidera un oggetto grafico che visualizzi, in base all'ID dipendente e al nome del dipendente, le richieste di rimborso spese stimate ancora da sostenere per il resto della settimana lunare.

Script di caricamento

```
Employee_Expenses:
Load
*
Inline
[
employee_id,employee_name,avg_daily_claim
182,Mark, $15
183,Deryck, $12.5
184,Dexter, $12.5
185,Sydney,$27
186,Agatha,$18
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Aggiungere i seguenti campi come dimensioni:
   - `employee_id`
   - `employee_name`
3. Quindi, creare la seguente misura per calcolare gli interessi accumulati:
   `=(lunarweekend(today(1))-today(1))*avg_daily_claim`
4. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

employee_id	employee_name	=(lunarweekend(today(1))-today(1))*avg_daily_claim
182	Contrassegno	$75.00
183	Deryck	$62.50
184	Dexter	$62.50
185	Sydney	$135.00
186	Agatha	$90.00

La funzione`lunarkweekend()`, utilizzando come unico argomento la data odierna, restituisce la data finale della settimana lunare corrente. Quindi, sottraendo la data odierna dalla data di fine settimana lunare, l'espressione restituisce il numero di giorni rimanenti di questa settimana.

Questo valore viene quindi moltiplicato per la media delle richieste di rimborso spese giornaliere di ciascun dipendente per calcolare il valore stimato delle richieste che ogni dipendente dovrebbe presentare durante il periodo rimanente della settimana lunare.

lunarweekname

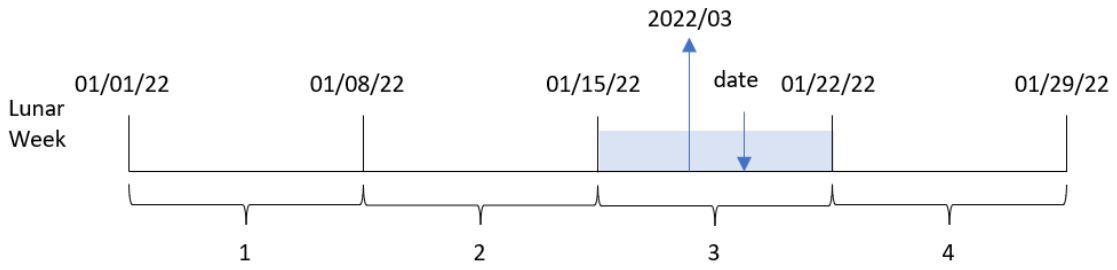
Questa funzione restituisce un valore di visualizzazione che mostra l'anno e il numero della settimana lunare corrispondente a un indicatore temporale del primo millisecondo del primo giorno della settimana lunare contenente **date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana e, a parte l'ultima settimana dell'anno, conterranno esattamente sette giorni.

Sintassi:

```
LunarWeekName(date [, period_no[, first_week_day]])
```

Tipo di dati restituiti: duale

Schema esemplificativo della funzione `lunarweekname()`



La funzione `lunarweekname()` determina in quale settimana lunare cade la data, iniziando il conteggio delle settimane dal 1° gennaio. Restituisce quindi un valore composto da `year/weekcount`.

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica la settimana lunare che contiene il valore **date**. I valori negativi di **period_no** indicano le settimane lunari precedenti, mentre i valori positivi indicano le settimane lunari successive.
first_week_day	Un differimento che può essere maggiore o minore di zero. Ciò modifica l'inizio dell'anno in base al numero specificato di giorni e/o frazioni di un giorno.

Casi di utilizzo

La funzione `lunarweekname()` è utile quando si desidera confrontare le aggregazioni per settimane lunari. Ad esempio, la funzione potrebbe essere utilizzata per determinare le vendite totali dei prodotti per settimana lunare. Le settimane lunari sono utili quando si vuole garantire che tutti i valori contenuti nella prima settimana dell'anno contengano solo valori a partire dal 1° gennaio.

È possibile creare queste dimensioni nello script di caricamento utilizzando la funzione che consente di creare un campo in una tabella Calendario principale. La funzione può anche essere utilizzata direttamente in un grafico come dimensione calcolata.

Esempi di funzioni

Esempio	Risultato
`lunarweekname('01/12/2013')`	Restituisce 2006/02.
`lunarweekname('01/12/2013', -1)`	Restituisce 2006/01.
`lunarweekname('01/12/2013', 0, 1)`	Restituisce 2006/02.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – data senza alcun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `lunar_week_name`, che restituisce l'anno e il numero della settimana lunare in cui sono avvenute le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        lunarweekname(date) as lunar_week_name
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
```

```
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `lunar_week_name`

Tabella dei risultati

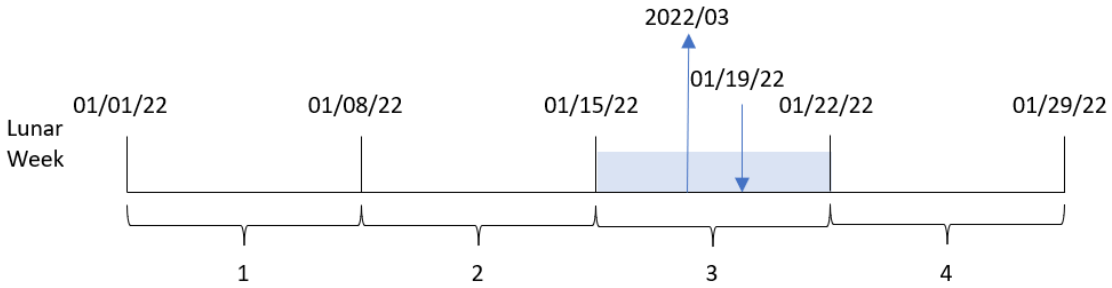
date	lunar_week_name
1/7/2022	2022/01
1/19/2022	2022/03
2/5/2022	2022/06
2/28/2022	2022/09
3/16/2022	2022/11
4/1/2022	2022/13
5/7/2022	2022/19
5/16/2022	2022/20
6/15/2022	2022/24
6/26/2022	2022/26
7/9/2022	2022/28
7/22/2022	2022/29
7/23/2022	2022/30
7/27/2022	2022/30
8/2/2022	2022/31

date	lunar_week_name
8/8/2022	2022/32
8/19/2022	2022/33
9/26/2022	2022/39
10/14/2022	2022/41
10/29/2022	2022/44

Il campo `lunar_week_name` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `lunarweekname()` e trasferendo il campo `date` come argomento della funzione.

La funzione `lunarweekname()` identifica la settimana lunare in cui cade il valore della data, restituendo l'anno e il numero della settimana di quella data.

Schema della funzione `lunarweekname()`, esempio senza argomenti aggiuntivi



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekname()` identifica che questa data cade nella settimana lunare che inizia il 15 gennaio; si tratta della terza settimana lunare dell'anno. Pertanto, il valore `lunar_week_name` restituito per tale transazione è 2022/03.

Esempio 2 - data con argomento period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_lunar_week_name`, che restituisce l'anno e il numero di settimana per la settimana lunare prima di quella in cui sono avvenute le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
```

```
Transactions:
    Load
        *,
        lunarweekname(date,-1) as previous_lunar_week_name
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_lunar_week_name`

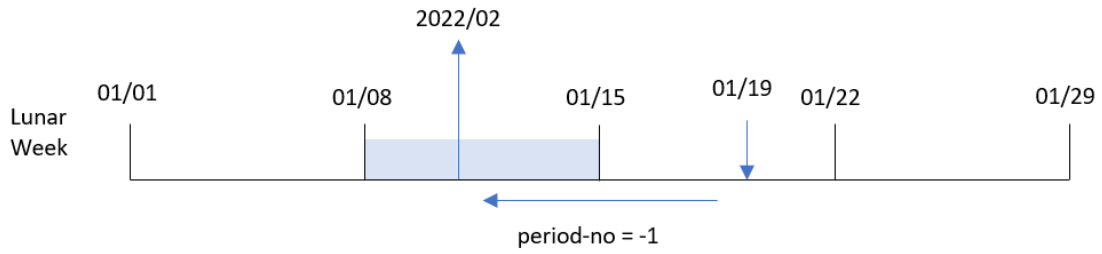
Tabella dei risultati

date	previous_lunar_week_name
1/7/2022	2021/52
1/19/2022	2022/02
2/5/2022	2022/05
2/28/2022	2022/08
3/16/2022	2022/10
4/1/2022	2022/12

date	previous_lunar_week_name
5/7/2022	2022/18
5/16/2022	2022/19
6/15/2022	2022/23
6/26/2022	2022/25
7/9/2022	2022/27
7/22/2022	2022/28
7/23/2022	2022/29
7/27/2022	2022/29
8/2/2022	2022/30
8/8/2022	2022/31
8/19/2022	2022/32
9/26/2022	2022/38
10/14/2022	2022/40
10/29/2022	2022/43

In questo caso, poiché è stato utilizzato un valore `period_no` di -1 come argomento di offset nella funzione `lunarweekname()`, la funzione identifica innanzitutto la settimana lunare in cui sono avvenute le transazioni. Quindi restituisce l'anno e il numero di una settimana prima.

Schema della funzione `lunarweekname()`, esempio di period_no



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekname()` identifica che questa transazione è avvenuta nella terza settimana lunare dell'anno, quindi restituisce l'anno e il valore di una settimana prima, 2022/02, per il campo `previous_lunar_week_name`.

Esempio 3 – data con argomento first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. In questo esempio, abbiamo impostato che le settimane lunari inizino il 5 gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        lunarweekname(date,0,4) as lunar_week_name
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

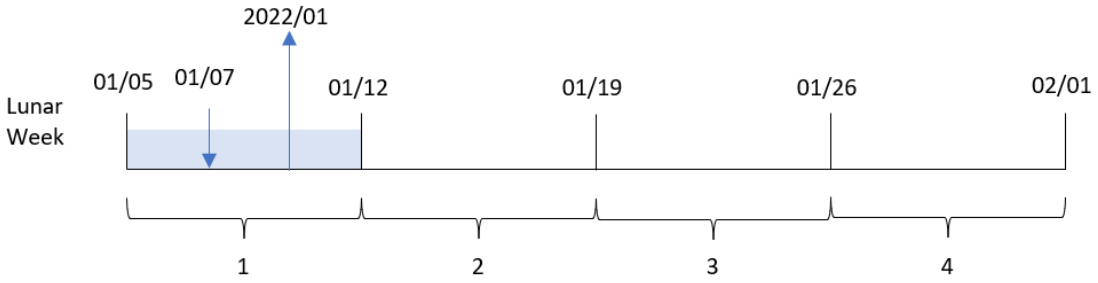
Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `lunar_week_name`

Tabella dei risultati

date	lunar_week_name
1/7/2022	2022/01
1/19/2022	2022/03
2/5/2022	2022/05
2/28/2022	2022/08
3/16/2022	2022/11
4/1/2022	2022/13
5/7/2022	2022/18
5/16/2022	2022/19
6/15/2022	2022/24
6/26/2022	2022/25
7/9/2022	2022/27
7/22/2022	2022/29
7/23/2022	2022/29
7/27/2022	2022/30
8/2/2022	2022/30
8/8/2022	2022/31
8/19/2022	2022/33
9/26/2022	2022/38
10/14/2022	2022/41
10/29/2022	2022/43

Schema della funzione `lunarweekname()`, esempio first_week_day



In questo caso, poiché nella funzione `lunarweekname()` viene utilizzato l'argomento `first_week_date` di 4, si sposta l'inizio delle settimane lunari dal 1° gennaio al 5 gennaio.

La transazione 8188 è avvenuta il 7 gennaio. Poiché le settimane lunari iniziano il 5 gennaio, la funzione `lunarweekname()` identifica che la settimana lunare contenente il 7 gennaio è la prima settimana lunare dell'anno. Pertanto, il valore `lunar_week_name` restituito per quella transazione è 2022/01.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce il numero della settimana lunare e l'anno in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Per calcolare la data di inizio della settimana lunare in cui avviene una transazione, creare la seguente misura:

```
=lunarweekname(date)
```

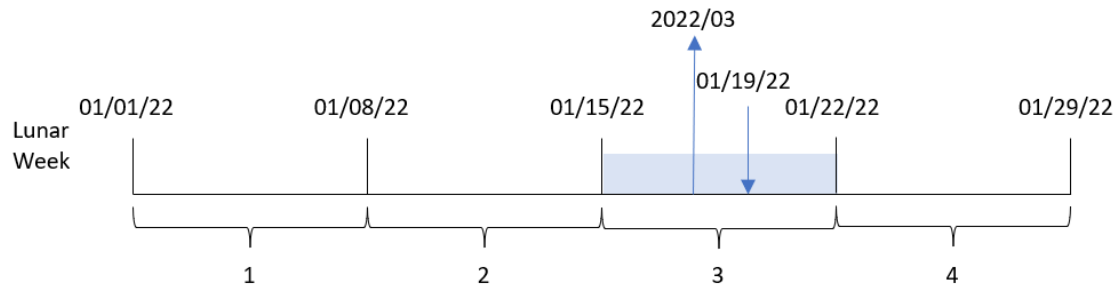
Tabella dei risultati

date	=lunarweekname(date)
1/7/2022	2022/01
1/19/2022	2022/03
2/5/2022	2022/06
2/28/2022	2022/09
3/16/2022	2022/11
4/1/2022	2022/13
5/7/2022	2022/19
5/16/2022	2022/20
6/15/2022	2022/24
6/26/2022	2022/26
7/9/2022	2022/28
7/22/2022	2022/29
7/23/2022	2022/30
7/27/2022	2022/30
8/2/2022	2022/31
8/8/2022	2022/32
8/19/2022	2022/33
9/26/2022	2022/39
10/14/2022	2022/41
10/29/2022	2022/44

La misura `lunar_week_name` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `lunarweekname()` e trasferendo il campo `date` come argomento della funzione.

La funzione `lunarweekname()` identifica la settimana lunare in cui cade il valore della data, restituendo l'anno e il numero della settimana di quella data.

Schema della funzione `lunarweekname()`, esempio di oggetto grafico



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekname()` identifica che questa data cade nella settimana lunare che inizia il 15 gennaio; si tratta della terza settimana lunare dell'anno. Pertanto, il valore `lunar_week_name` per tale transazione è 2022/03.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

L'utente finale desidera un oggetto grafico che presenti le vendite totali per settimana per l'anno in corso. La settimana 1, con una durata di sette giorni, dovrebbe iniziare il 1° gennaio. È possibile ottenere questo risultato anche quando questa dimensione non è disponibile nel modello di dati, utilizzando la funzione `lunarweekname()` come dimensione calcolata nel grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
```

```
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Creare una dimensione calcolata utilizzando la seguente espressione:
   `=lunarweekname(date)`
3. Calcolare le vendite totali utilizzando la seguente misura di aggregazione:
   `=sum(amount)`
4. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

=lunarweekname(date)	=sum(amount)
2022/01	$17.17
2022/03	$37.23
2022/06	$57.42
2022/09	$88.27
2022/11	$53.80
2022/13	$82.06
2022/19	$40.39
2022/20	$87.21
2022/24	$95.93
2022/26	$45.89
2022/28	$36.23
2022/29	$25.66
2022/30	$152.75

=lunarweekname(date)	=sum(amount)
2022/31	$76.11
2022/32	$25.12
2022/33	$46.23
2022/39	$84.21
2022/41	$96.24
2022/44	$67.67

lunarweekstart

Questa funzione restituisce un valore corrispondente a un indicatore temporale del primo millisecondo del primo giorno della settimana lunare contenente **date**. Le settimane lunari in Qlik Sense sono definite contando il 1° gennaio come primo giorno della settimana e, a parte l'ultima settimana dell'anno, conterranno esattamente sette giorni.

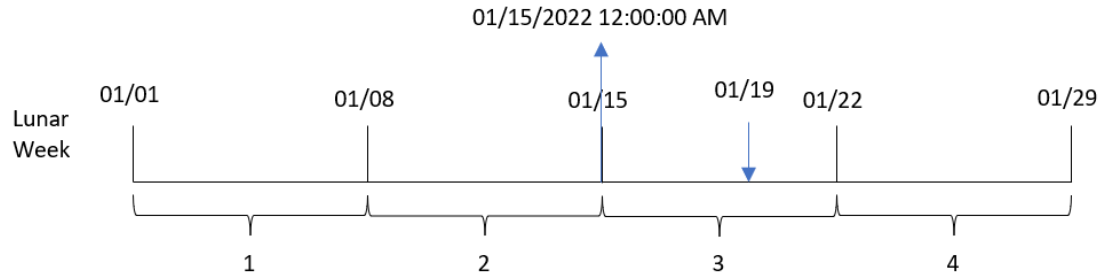
Sintassi:

```
LunarweekStart(date[, period_no[, first_week_day]])
```

Tipo di dati restituiti: duale

La funzione `lunarweekstart()` determina la settimana lunare in cui cade `date`. Quindi, restituisce un timestamp, nel formato data, per il primo millisecondo di quella settimana.

Schema esemplificativo della funzione `lunarweekstart()`



Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica la settimana lunare che contiene il valore **date**. I valori negativi di **period_no** indicano le settimane lunari precedenti, mentre i valori positivi indicano le settimane lunari successive.
first_week_day	Un differimento che può essere maggiore o minore di zero. Ciò modifica l'inizio dell'anno in base al numero specificato di giorni e/o frazioni di un giorno.

Casi di utilizzo

La funzione `lunarweekstart()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione della settimana trascorsa finora. A differenza della funzione `weekstart()`, all'inizio di ogni nuovo anno solare, la settimana inizia il 1° gennaio e ogni settimana successiva inizia sette giorni dopo. La funzione `lunarweekstart()` non è interessata dalla variabile di sistema `FirstWeekDay`.

Ad esempio, `lunarweekstart()` è utilizzabile per calcolare gli interessi accumulati in una settimana fino ad oggi.

Esempi di funzioni

Esempio	Risultato
`lunarweekstart ('01/12/2013')`	Restituisce 01/08/2013.
`lunarweekstart ('01/12/2013', -1)`	Restituisce 01/01/2013.
`lunarweekstart ('01/12/2013', 0, 1 )`	Restituisce 01/09/2013, perché l'impostazione first_week_day su 1 significa che l'inizio dell'anno è stato cambiato in 01/02/2013.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `start_of_week`, che restituisce un timestamp per l'inizio della settimana lunare in cui sono avvenute le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        lunarweekstart(date) as start_of_week,
        timestamp(lunarweekstart(date)) as start_of_week_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `start_of_week`
- `start_of_week_timestamp`

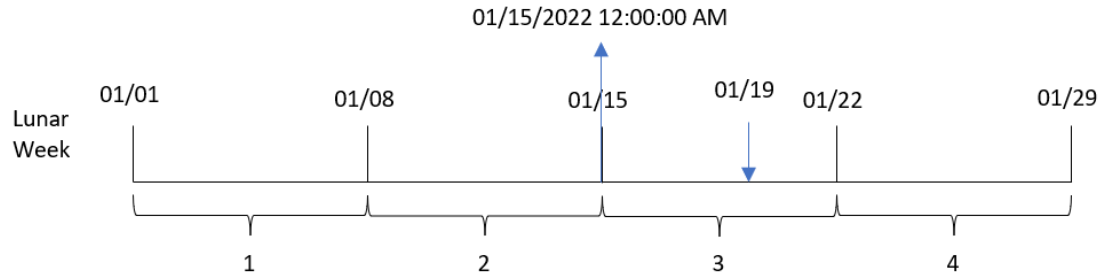
Tabella dei risultati

date	start_of_week	start_of_week_timestamp
1/7/2022	01/01/2022	1/1/2022 12:00:00 AM
1/19/2022	01/15/2022	1/15/2022 12:00:00 AM
2/5/2022	02/05/2022	2/5/2022 12:00:00 AM
2/28/2022	02/26/2022	2/26/2022 12:00:00 AM
3/16/2022	03/12/2022	3/12/2022 12:00:00 AM
4/1/2022	03/26/2022	3/26/2022 12:00:00 AM
5/7/2022	05/07/2022	5/7/2022 12:00:00 AM
5/16/2022	05/14/2022	5/14/2022 12:00:00 AM
6/15/2022	06/11/2022	6/11/2022 12:00:00 AM
6/26/2022	06/25/2022	6/25/2022 12:00:00 AM
7/9/2022	07/09/2022	7/9/2022 12:00:00 AM
7/22/2022	07/16/2022	7/16/2022 12:00:00 AM
7/23/2022	07/23/2022	7/23/2022 12:00:00 AM
7/27/2022	07/23/2022	7/23/2022 12:00:00 AM
8/2/2022	07/30/2022	7/30/2022 12:00:00 AM
8/8/2022	08/06/2022	8/6/2022 12:00:00 AM
8/19/2022	08/13/2022	8/13/2022 12:00:00 AM
9/26/2022	09/24/2022	9/24/2022 12:00:00 AM
10/14/2022	10/08/2022	10/8/2022 12:00:00 AM
10/29/2022	10/29/2022	10/29/2022 12:00:00 AM

Il campo `start_of_week` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `lunarweekstart()` e trasferendo il campo `date` come argomento della funzione.

La funzione `lunarweekstart()` identifica la settimana lunare in cui cade la data, restituendo un timestamp per il primo millisecondo di quella settimana.

Schema della funzione `lunarweekstart()`, esempio senza argomenti aggiuntivi



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekstart()` identifica che la settimana lunare inizia il 15 gennaio. Pertanto, il valore `start_of_week` per questa transazione restituisce il primo millisecondo di quel giorno, ovvero il 15 gennaio alle 12:00:00 AM.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_lunar_week_start`, che restituisce un timestamp per l'inizio della settimana lunare prima che avvenga la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        lunarweekstart(date,-1) as previous_lunar_week_start,
        timestamp(lunarweekstart(date,-1)) as previous_lunar_week_start_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
```

```
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

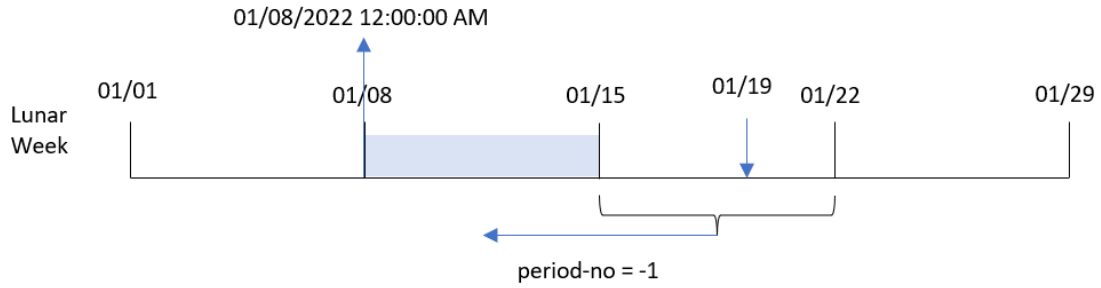
Risultati

Tabella dei risultati

date	previous_lunar_week_start	previous_lunar_week_start_timestamp
1/7/2022	12/24/2021	12/24/2021 12:00:00 AM
1/19/2022	01/08/2022	1/8/2022 12:00:00 AM
2/5/2022	01/29/2022	1/29/2022 12:00:00 AM
2/28/2022	02/19/2022	2/19/2022 12:00:00 AM
3/16/2022	03/05/2022	3/5/2022 12:00:00 AM
4/1/2022	03/19/2022	3/19/2022 12:00:00 AM
5/7/2022	04/30/2022	4/30/2022 12:00:00 AM
5/16/2022	05/07/2022	5/7/2022 12:00:00 AM
6/15/2022	06/04/2022	6/4/2022 12:00:00 AM
6/26/2022	06/18/2022	6/18/2022 12:00:00 AM
7/9/2022	07/02/2022	7/2/2022 12:00:00 AM
7/22/2022	07/09/2022	7/9/2022 12:00:00 AM
7/23/2022	07/16/2022	7/16/2022 12:00:00 AM
7/27/2022	07/16/2022	7/16/2022 12:00:00 AM
8/2/2022	07/23/2022	7/23/2022 12:00:00 AM
8/8/2022	07/30/2022	7/30/2022 12:00:00 AM
8/19/2022	08/06/2022	8/6/2022 12:00:00 AM
9/26/2022	09/17/2022	9/17/2022 12:00:00 AM
10/14/2022	10/01/2022	10/1/2022 12:00:00 AM
10/29/2022	10/22/2022	10/22/2022 12:00:00 AM

In questo caso, poiché il valore `period_no` di -1 è stato utilizzato come argomento offset nella funzione `lunarweekstart()`, la funzione per prima cosa identifica la settimana lunare in cui avvengono le transazioni. Si sposta poi una settimana prima e identifica il primo millisecondo di quella settimana lunare.

Schema della funzione `lunarweekstart()`, esempio di period_no



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekstart()` identifica che la settimana lunare inizia il 15 gennaio. Pertanto, la settimana lunare precedente è iniziata l'8 gennaio alle 12:00:00; questo è il valore restituito per il campo `previous_lunar_week_start`.

Esempio 3 – first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. In questo esempio, abbiamo impostato che le settimane lunari inizino il 5 gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        lunarweekstart(date,0,4) as start_of_week,
        timestamp(lunarweekstart(date,0,4)) as start_of_week_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
```

```
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `start_of_week`
- `start_of_week_timestamp`

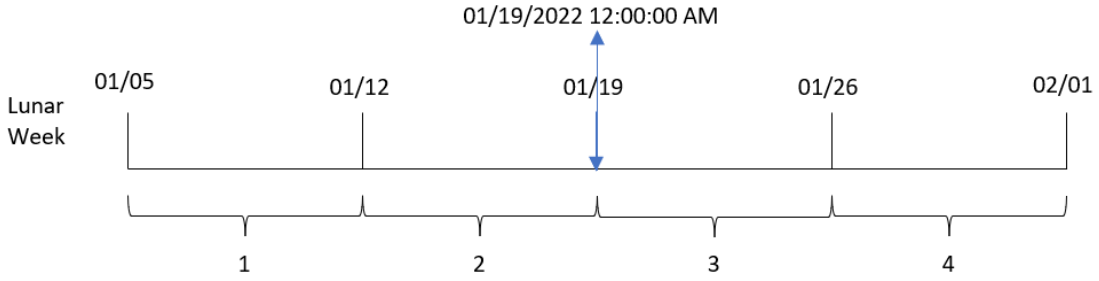
Tabella dei risultati

date	start_of_week	start_of_week_timestamp
1/7/2022	01/05/2022	1/5/2022 12:00:00 AM
1/19/2022	01/19/2022	1/19/2022 12:00:00 AM
2/5/2022	02/02/2022	2/2/2022 12:00:00 AM
2/28/2022	02/23/2022	2/23/2022 12:00:00 AM
3/16/2022	03/16/2022	3/16/2022 12:00:00 AM
4/1/2022	03/30/2022	3/30/2022 12:00:00 AM
5/7/2022	05/04/2022	5/4/2022 12:00:00 AM
5/16/2022	05/11/2022	5/11/2022 12:00:00 AM
6/15/2022	06/15/2022	6/15/2022 12:00:00 AM
6/26/2022	06/22/2022	6/22/2022 12:00:00 AM
7/9/2022	07/06/2022	7/6/2022 12:00:00 AM
7/22/2022	07/20/2022	7/20/2022 12:00:00 AM
7/23/2022	07/20/2022	7/20/2022 12:00:00 AM
7/27/2022	07/27/2022	7/27/2022 12:00:00 AM
8/2/2022	07/27/2022	7/27/2022 12:00:00 AM

date	start_of_week	start_of_week_timestamp
8/8/2022	08/03/2022	8/3/2022 12:00:00 AM
8/19/2022	08/17/2022	8/17/2022 12:00:00 AM
9/26/2022	09/21/2022	9/21/2022 12:00:00 AM
10/14/2022	10/12/2022	10/12/2022 12:00:00 AM
10/29/2022	10/26/2022	10/26/2022 12:00:00 AM

In questo caso, poiché nella funzione `lunarweekstart()` viene utilizzato l'argomento `first_week_date` di 4, l'inizio dell'anno viene spostato dal 1° al 5 gennaio.

Schema della funzione `lunarweekstart()`, esempio first_week_day



La transazione 8189 è avvenuta il 19 gennaio. Poiché le settimane lunari iniziano il 5 gennaio, la funzione `lunarweekstart()` identifica che la settimana lunare contenente il 19 gennaio inizia anch'essa il 19 gennaio alle 12:00:00 AM. Pertanto, ciò corrisponde al valore restituito per il campo `start_of_week`.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce un timestamp per l'inizio della settimana lunare in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
```

```
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Aggiungere le seguenti misure:

```
=lunarweekstart(date)
```

```
=timestamp(lunarweekstart(date))
```

Tabella dei risultati

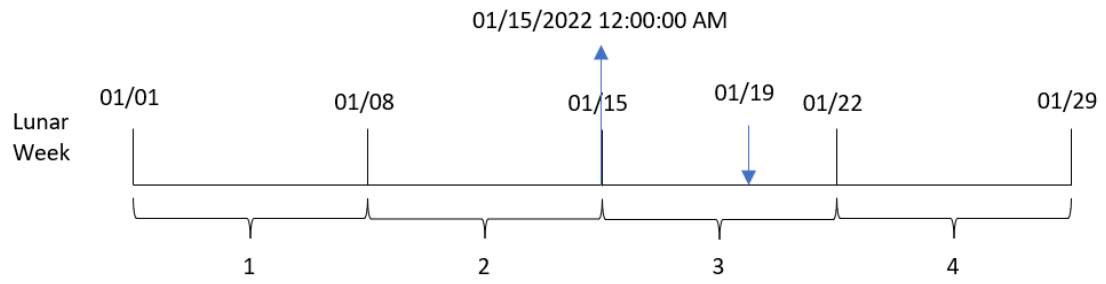
date	=lunarweekstart(date)	=timestamp(lunarweekstart(date))
1/7/2022	01/01/2022	1/1/2022 12:00:00 AM
1/19/2022	01/15/2022	1/15/2022 12:00:00 AM
2/5/2022	02/05/2022	2/5/2022 12:00:00 AM
2/28/2022	02/26/2022	2/26/2022 12:00:00 AM
3/16/2022	03/12/2022	3/12/2022 12:00:00 AM
4/1/2022	03/26/2022	3/26/2022 12:00:00 AM
5/7/2022	05/07/2022	5/7/2022 12:00:00 AM
5/16/2022	05/14/2022	5/14/2022 12:00:00 AM
6/15/2022	06/11/2022	6/11/2022 12:00:00 AM

date	=lunarweekstart(date)	=timestamp(lunarweekstart(date))
6/26/2022	06/25/2022	6/25/2022 12:00:00 AM
7/9/2022	07/09/2022	7/9/2022 12:00:00 AM
7/22/2022	07/16/2022	7/16/2022 12:00:00 AM
7/23/2022	07/23/2022	7/23/2022 12:00:00 AM
7/27/2022	07/23/2022	7/23/2022 12:00:00 AM
8/2/2022	07/30/2022	7/30/2022 12:00:00 AM
8/8/2022	08/06/2022	8/6/2022 12:00:00 AM
8/19/2022	08/13/2022	8/13/2022 12:00:00 AM
9/26/2022	09/24/2022	9/24/2022 12:00:00 AM
10/14/2022	10/08/2022	10/8/2022 12:00:00 AM
10/29/2022	10/29/2022	10/29/2022 12:00:00 AM

La misura `start_of_week` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `lunarweekstart()` e trasferendo il campo data come argomento della funzione.

La funzione `lunarweekstart()` identifica in quale settimana lunare cade il valore della data, restituendo un timestamp per l'ultimo millisecondo di quella settimana.

Schema della funzione `lunarweekstart()`, esempio di oggetto grafico



La transazione 8189 è avvenuta il 19 gennaio. La funzione `lunarweekstart()` identifica che la settimana lunare inizia il 15 gennaio. Pertanto, il valore `start_of_week` per questa transazione corrisponde al primo millisecondo di quel giorno, ovvero il 15 gennaio alle 12:00:00 AM.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di saldi di prestiti, caricato in una tabella denominata `Loans`.
- Dati costituiti dagli ID dei prestiti, dal saldo all'inizio della settimana e dal tasso di interesse semplice annuo applicato a ciascun prestito.

L'utente finale desidera un oggetto grafico che mostri, in base all'ID del prestito, gli interessi correnti che sono stati maturati per ciascun prestito nella settimana in corso.

Script di caricamento

```
Loans:
Load
*
Inline
[
loan_id,start_balance,rate
8188,$10000.00,0.024
8189,$15000.00,0.057
8190,$17500.00,0.024
8191,$21000.00,0.034
8192,$90000.00,0.084
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Aggiungere i seguenti campi come dimensioni:
    - `loan_id`
    - `start_balance`
3. Quindi, creare la seguente misura per calcolare gli interessi accumulati:
   `=start_balance*(rate*(today(1)-lunarweekstart(today(1)))/365)`
4. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

loan_id	start_balance	=start_balance*(rate*(today(1)- lunarweekstart (today(1)))/365)
8188	$10000.00	$15.07
8189	$15000.00	$128.84
8190	$17500.00	$63.29
8191	$21000.00	$107.59
8192	$90000.00	$1139.18

Utilizzando la data odierna come unico argomento, la funzione `lunarweekstart()` restituisce la data di inizio dell'anno corrente. Sottraendo tale risultato dalla data corrente, l'espressione restituisce il numero di giorni trascorsi fino ad ora questa settimana.

Questo valore viene quindi moltiplicato per il tasso di interesse e diviso per 365, per restituire il tasso di interesse effettivo incorso per questo periodo. Il risultato viene poi moltiplicato per il saldo iniziale del prestito per ottenere gli interessi maturati fino a questo momento della settimana.

makedate

Questa funzione restituisce una data calcolata dall'anno **YYYY**, dal mese **MM** e dal giorno **DD**.

Sintassi:

```
MakeDate(YYYY [ , MM [ , DD ] ])
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
YYYY	L'anno è un numero intero.
MM	Il mese è un numero intero. Se non si indica il mese, verrà utilizzato come valore 1 (gennaio).
DD	Il giorno è un numero intero. Se non si indica un giorno, verrà utilizzato come valore 1 (primo giorno).

Casi di utilizzo

La funzione `makedate()` viene comunemente utilizzata nello script per la generazione dei dati per generare un calendario. Ciò potrebbe essere usato anche quando il campo data non è direttamente disponibile come data, ma necessita di alcune trasformazioni per estrarre i componenti di anno, mese e giorno.

In questi esempi viene utilizzato il formato della data (MM/GG/AAAA). Il formato della data viene specificato nell'istruzione `SET DateFormat` nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di funzioni

Esempio	Risultato
`makedate(2012)`	Restituisce 01/01/2012.
`makedate(12)`	Restituisce 01/01/2012.
`makedate(2012,12)`	Restituisce 12/01/2012.
`makedate(2012,2,14)`	Restituisce 02/14/2012.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2018, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `transaction_date`, che restituisce una data nel formato MM/GG/AAAA.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        makedate(transaction_year, transaction_month, transaction_day) as transaction_date
        ;
Load * Inline [
transaction_id, transaction_year, transaction_month, transaction_day, transaction_amount,
transaction_quantity, customer_id
3750, 2018, 08, 30, 12423.56, 23, 2038593
3751, 2018, 09, 07, 5356.31, 6, 203521
3752, 2018, 09, 16, 15.75, 1, 5646471
3753, 2018, 09, 22, 1251, 7, 3036491
3754, 2018, 09, 22, 21484.21, 1356, 049681
3756, 2018, 09, 22, -59.18, 2, 2038593
3757, 2018, 09, 23, 3177.4, 21, 203521
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- transaction_year
- transaction_month
- transaction_day
- transaction_date

Tabella dei risultati

transaction_year	transaction_month	transaction_day	transaction_date
2018	08	30	08/30/2018
2018	09	07	09/07/2018
2018	09	16	09/16/2018
2018	09	22	09/22/2018
2018	09	23	09/23/2018

Il campo transaction_date viene creato nella precedente istruzione LOAD utilizzando la funzione makedate() e passando i campi anno, mese e giorno come argomenti della funzione.

La funzione combina e converte questi valori in un campo data, restituendo i risultati nel formato della variabile di sistema DateFormat.

Esempio 2 – DateFormat modificato

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, transaction_date, nel formato GG/MM/AAAA senza modificare la variabile di sistema DateFormat.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        date(makedate(transaction_year, transaction_month, transaction_day), 'DD/MM/YYYY') as
transaction_date
```

```
        ;
Load * Inline [
transaction_id, transaction_year, transaction_month, transaction_day, transaction_amount,
transaction_quantity, customer_id
3750, 2018, 08, 30, 12423.56, 23, 2038593
3751, 2018, 09, 07, 5356.31, 6, 203521
3752, 2018, 09, 16, 15.75, 1, 5646471
3753, 2018, 09, 22, 1251, 7, 3036491
3754, 2018, 09, 22, 21484.21, 1356, 049681
3756, 2018, 09, 22, -59.18, 2, 2038593
3757, 2018, 09, 23, 3177.4, 21, 203521
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `transaction_year`
- `transaction_month`
- `transaction_day`
- `transaction_date`

Tabella dei risultati

transaction_year	transaction_month	transaction_day	transaction_date
2018	08	30	30/08/2018
2018	09	07	07/09/2018
2018	09	16	16/09/2018
2018	09	22	22/09/2018
2018	09	23	23/09/2018

In questo caso, la funzione `makedate()` è nidificata all'interno della funzione `date()`. Il secondo argomento della funzione `date()` imposta il formato dei risultati della funzione `makedate()` come GG/MM/AAAA.

Esempio 3 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2018, caricato in una tabella denominata `Transactions`.
- Le date delle transazioni sono fornite in due campi: `year` e `month`.

Creare una misura dell'oggetto grafico, `transaction_date`, che restituisca una data nel formato MM/GG/AAAA.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load * Inline [
transaction_id, transaction_year, transaction_month, transaction_amount, transaction_quantity,
customer_id
3750, 2018, 08, 12423.56, 23, 2038593
3751, 2018, 09, 5356.31, 6, 203521
3752, 2018, 09, 15.75, 1, 5646471
3753, 2018, 09, 1251, 7, 3036491
3754, 2018, 09, 21484.21, 1356, 049681
3756, 2018, 09, -59.18, 2, 2038593
3757, 2018, 09, 3177.4, 21, 203521
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `year`
- `month`

Per determinare il valore `transaction_date`, creare questa misura:

```
=makedate(transaction_year,transaction_month)
```

Tabella dei risultati

transaction_year	transaction_month	transaction_date
2018	08	08/01/2018
2018	09	09/01/2018

La misura `transaction_date` viene creata nell'oggetto grafico utilizzando la funzione `makedate()` e passando i campi anno e mese come argomenti della funzione.

La funzione combina quindi questi valori e il valore presunto del giorno, pari a 01. Questi valori vengono poi convertiti in un campo data, restituendo i risultati nel formato della variabile di sistema `DateFormat`.

Esempio 4 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Creare un set di dati di calendario per l'anno solare 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Calendar:
       load
                *
    where year(date)=2022;
load
       date(recno()+makedate(2021,12,31)) as date
AutoGenerate 400;
```

Risultati

Tabella dei risultati

date
01/01/2022
01/02/2022
01/03/2022
01/04/2022
01/05/2022
01/06/2022
01/07/2022
01/08/2022
01/09/2022
01/10/2022
01/11/2022
01/12/2022
01/13/2022
01/14/2022
01/15/2022
01/16/2022
01/17/2022
01/18/2022
01/19/2022
01/20/2022
01/21/2022

date
01/22/2022
01/23/2022
01/24/2022
01/25/2022
+ 340 ulteriori righe

La funzione `makedate()` crea un valore di data per il 31 dicembre 2021. La funzione `recno()` fornisce il numero di record del record correntemente caricato nella tabella, a partire da 1. Pertanto, il primo record ha la data del 1° gennaio 2022. A ogni valore `recno()` successivo, la data viene incrementata di 1. Questa espressione è racchiusa in una funzione `date()` che converte il valore in una data. Questo processo viene ripetuto 400 volte dalla funzione `autogenerate`. Infine, sfruttando un caricamento precedente, è possibile utilizzare una condizione `where` per caricare solo le date a partire dall'anno 2022. Questo script genera un calendario contenente tutte le date del 2022.

maketime

Questa funzione restituisce una data calcolata dall'ora **hh**, dal minuto **mm** e dal secondo **ss**.

Sintassi:

```
MakeTime(hh [ , mm [ , ss ] ])
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
hh	L'ora è un numero intero.
mm	Il minuto è un numero intero. Se non si indicano i minuti, verrà utilizzato il valore 00.
ss	Il secondo è un numero intero. Se non si indicano i secondi, verrà utilizzato il valore 00.

Casi di utilizzo

La funzione `maketime()` viene comunemente utilizzata nello script per la generazione dei dati per generare un campo temporale. A volte, quando il campo dell'ora è derivato dal testo in ingresso, questa funzione può essere utilizzata per costruire l'ora utilizzando i suoi componenti.

Questi esempi utilizzano il formato orario `h:mm:ss`. Il formato dell'ora viene specificato nell'istruzione `SET TimeFormat` nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di funzioni

Esempio	Risultato
`maketime(22)`	Restituisce 22:00:00.
`maketime(22, 17)`	Restituisce 22:17:00.
`maketime(22,17,52 )`	Restituisce 22:17:52.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – maketime()

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni, caricato in una tabella denominata `Transactions`.
- Gli orari delle transazioni sono forniti in tre campi: `hours`, `minutes` e `seconds`.
- La creazione di un campo, `transaction_time`, che restituisce l'ora nel formato della variabile di sistema `TimeFormat`.

Script di caricamento

```
SET TimeFormat='h:mm:ss TT';

Transactions:
    Load
        *,
        maketime(transaction_hour, transaction_minute, transaction_second) as transaction_time
        ;
Load * Inline [
transaction_id, transaction_hour, transaction_minute, transaction_second, transaction_amount,
```

```
transaction_quantity, customer_id
3750, 18, 43, 30, 12423.56, 23, 2038593
3751, 6, 32, 07, 5356.31, 6, 203521
3752, 12, 09, 16, 15.75, 1, 5646471
3753, 21, 43, 41, 7, 3036491
3754, 17, 55, 22, 21484.21, 1356, 049681
3756, 2, 52, 22, -59.18, 2, 2038593
3757, 9, 25, 23, 3177.4, 21, 203521
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `transaction_hour`
- `transaction_minute`
- `transaction_second`
- `transaction_time`

Tabella dei risultati

transaction_hour	transaction_minute	transaction_second	transaction_time
2	52	22	2:52:22 AM
6	32	07	6:32:07 AM
9	25	23	9:25:23 AM
12	09	16	12:09:16 PM
17	55	22	5:55:22 PM
18	43	30	6:43:30 PM
21	43	41	9:43:41 PM

Il campo `transaction_time` viene creato nella precedente istruzione LOAD utilizzando la funzione `maketime()` e passando i campi ora, minuti e secondi come argomenti della funzione.

La funzione combina e converte questi valori in un campo temporale, restituendo i risultati nel formato orario della variabile di sistema `TimeFormat`.

Esempio 2 – time() function

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `transaction_time`, che ci permetterà di mostrare i risultati in formato orario 24 ore senza modificare la variabile di sistema `TimeFormat`.

Script di caricamento

```
SET TimeFormat='h:mm:ss TT';

Transactions:
    Load
        *,
        time(maketime(transaction_hour, transaction_minute, transaction_second),'h:mm:ss') as
transaction_time
        ;
Load * Inline [
transaction_id, transaction_hour, transaction_minute, transaction_second, transaction_amount,
transaction_quantity, customer_id
3750, 18, 43, 30, 12423.56, 23, 2038593
3751, 6, 32, 07, 5356.31, 6, 203521
3752, 12, 09, 16, 15.75, 1, 5646471
3753, 21, 43, 41, 7, 3036491
3754, 17, 55, 22, 21484.21, 1356, 049681
3756, 2, 52, 22, -59.18, 2, 2038593
3757, 9, 25, 23, 3177.4, 21, 203521
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `transaction_hour`
- `transaction_minute`
- `transaction_second`
- `transaction_time`

Tabella dei risultati

transaction_hour	transaction_minute	transaction_second	transaction_time
2	52	22	2:52:22
6	32	07	6:32:07
9	25	23	9:25:23
12	09	16	12:09:16
17	55	22	17:55:22
18	43	30	18:43:30
21	43	41	21:43:41

In questo caso, la funzione `maketime()` è nidificata all'interno della funzione `time()`. Il secondo argomento della funzione `time()` imposta il formato dei risultati della funzione `maketime()` come il valore `h:mm:ss` richiesto.

Esempio 3 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni, caricato in una tabella denominata `Transactions`.
- Tempi di transazione forniti in due campi: `hours` e `minutes`.
- La creazione di un campo, `transaction_time`, che restituisce l'ora nel formato della variabile di sistema `TimeFormat`.

Creare una misura dell'oggetto grafico, `transaction_time`, che restituisca un orario nel formato `h:mm:ss TT`.

Script di caricamento

```
SET TimeFormat='h:mm:ss TT';

Transactions:
Load * Inline [
transaction_id, transaction_hour, transaction_minute, transaction_amount, transaction_
quantity, customer_id
3750, 18, 43, 12423.56, 23, 2038593
3751, 6, 32, 5356.31, 6, 203521
3752, 12, 09, 15.75, 1, 5646471
3753, 21, 43, 7, 3036491
3754, 17, 55, 21484.21, 1356, 049681
3756, 2, 52, -59.18, 2, 2038593
3757, 9, 25, 3177.4, 21, 203521
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `transaction_hour`
- `transaction_minute`

Per calcolare il valore `transaction_time`, creare questa misura:

```
=maketime(transaction_hour,transaction_minute)
```

Tabella dei risultati

transaction_hour	transaction_minute	=maketime(transaction_hour, transaction_minute)
2	52	2:52:00 AM
6	32	6:32:00 AM
9	25	9:25:00 AM
12	09	12:09:00 PM
17	55	5:55:00 PM
18	43	6:43:00 PM
21	43	9:43:00 PM

La misura `transaction_time` viene creata nell'oggetto grafico utilizzando la funzione `maketime()` e passando i campi anno e minuto come argomenti della funzione.

La funzione combina quindi questi valori e i secondi vengono assunti pari a 00. Questi valori vengono poi convertiti in un campo temporale, restituendo i risultati nel formato della variabile di sistema `TimeFormat`.

Esempio 4 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Creare un set di dati di calendario per il mese di gennaio 2022, suddiviso in incrementi di otto ore.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

tmpCalendar:
       load
                *
    where year(date)=2022;
load
       date(recno()+makedate(2021,12,31)) as date
AutoGenerate 31;

Left join(tmpCalendar)
load
       maketime((recno()-1)*8,00,00) as time
autogenerate 3;

Calendar:
load
       timestamp(date + time) as timestamp
resident tmpCalendar;

drop table tmpCalendar;
```

Risultati

Tabella dei risultati

timestamp
1/1/2022 12:00:00 AM
1/1/2022 8:00:00 AM
1/1/2022 4:00:00 PM
1/2/2022 12:00:00 AM
1/2/2022 8:00:00 AM
1/2/2022 4:00:00 PM
1/3/2022 12:00:00 AM
1/3/2022 8:00:00 AM
1/3/2022 4:00:00 PM
1/4/2022 12:00:00 AM
1/4/2022 8:00:00 AM
1/4/2022 4:00:00 PM
1/5/2022 12:00:00 AM
1/5/2022 8:00:00 AM
1/5/2022 4:00:00 PM
1/6/2022 12:00:00 AM
1/6/2022 8:00:00 AM
1/6/2022 4:00:00 PM
1/7/2022 12:00:00 AM
1/7/2022 8:00:00 AM
1/7/2022 4:00:00 PM
1/8/2022 12:00:00 AM
1/8/2022 8:00:00 AM
1/8/2022 4:00:00 PM
1/9/2022 12:00:00 AM
+ 68 ulteriori righe

La funzione iniziale `autogenerate` crea un calendario contenente tutte le date di gennaio in una tabella chiamata `tmpCalendar`.

Viene creata una seconda tabella, contenente tre record. Per ogni record, viene preso il valore `recno() - 1` (valori 0, 1, 2) e il risultato viene moltiplicato per 8. Di conseguenza, si ottengono i valori 0, 8 e 16. Questi valori vengono utilizzati come parametro dell'ora in una funzione `maketime()`, con i valori dei minuti e dei secondi pari a 0. Di conseguenza, la tabella contiene tre campi orari: 12:00:00 AM, 8:00:00 AM e 4:00:00 PM.

Questa tabella è unita alla tabella `tmpCalendar`. Poiché non ci sono campi corrispondenti tra le due tabelle per il join, le righe dell'ora vengono aggiunte a ciascuna riga della data. Di conseguenza, ogni riga di data viene ripetuta tre volte per ogni valore temporale.

Infine, la tabella Calendario viene creata da un caricamento residente della tabella `tmpCalendar`. I campi data e ora vengono concatenati e avvolti nella funzione `timestamp()` per creare il campo timestamp.

La tabella `tmpCalendar` viene quindi rilasciata.

makeweekdate

Questa funzione restituisce una data calcolata dall'anno, dal numero di settimana e dal giorno della settimana.

Sintassi:

```
MakeWeekDate(weekyear [, week [, weekday [, first_week_day [, broken_weeks [, reference_day]]]]])
```

Tipo di dati restituiti: duale

La funzione `makeweekdate()` è disponibile sia come script che come grafico. La funzione calcola la data in base ai parametri inseriti nella funzione.

Argomenti

Argomento	Descrizione
weekyear	L'anno definito dalla funzione `WeekYear()` per la data specifica, ovvero l'anno a cui appartiene il numero della settimana.  *In alcuni casi l'anno settimanale può essere diverso dall'anno solare, ad esempio se la settimana 1 inizia già a dicembre dell'anno precedente.*
week	Il numero della settimana definito dalla funzione `Week()` per la data specifica. Se non viene dichiarato alcun numero di settimana, viene utilizzato -1.

Argomento	Descrizione
weekday	Il giorno della settimana definito dalla funzione `WeekDay()` per la data in questione. 0 è il primo giorno della settimana e 6 è l'ultimo giorno della settimana. Se non si indica alcun giorno della settimana, verrà utilizzato il valore 0. *Anche se 0 significa sempre primo giorno della settimana e 6 è sempre l'ultimo, il parametro **first_week_day** determina a quali giorni della settimana corrisponde. Se omesso, viene utilizzato il valore della variabile **FirstWeekDay**.* Se si utilizzano settimane parziali, insieme a una combinazione impossibile di parametri, si può ottenere un risultato che non appartiene all'anno scelto. **Esempio:** `MakeWeekDate(2021,1,0,6,1)` Restituisce "27 dicembre 2020", poiché questo giorno è il primo (la domenica) della settimana specificata. Il 1° gennaio 2021 era un venerdì.
first_week_day	Specifica il giorno di inizio della settimana. Se omesso, viene utilizzato il valore della variabile **FirstWeekDay**. I valori possibili per **first_week_day** sono 0 per lunedì, 1 per martedì, 2 per mercoledì, 3 per giovedì, 4 per venerdì, 5 per sabato e 6 per domenica. Per maggiori informazioni sulle variabili di sistema, vedere *FirstWeekDay (page 224)*.
broken_weeks	Se non si specifica **broken_weeks**, il valore della variabile **BrokenWeeks** verrà utilizzato per definire se le settimane sono parziali o meno.
reference_day	Se non si specifica **reference_day**, il valore della variabile **ReferenceDay** verrà utilizzato per definire quale giorno di gennaio impostare come giorno di riferimento per definire la settimana 1.

Casi di utilizzo

La funzione `makeweekdate()` viene comunemente utilizzata nello script per la generazione di dati per generare un elenco di date o per costruire date quando l'anno, la settimana e il giorno della settimana sono forniti nei dati di input.

Gli esempi seguenti presuppongono:

```
SET FirstWeekDay=0;
SET BrokenWeeks=0;
SET ReferenceDay=4;
```

Esempi di funzioni

Esempio	Risultato
`makeweekdate(2014,6,6)`	restituisce 02/09/2014
`makeweekdate(2014,6,1)`	restituisce 02/04/2014
`makeweekdate(2014,6)`	restituisce 02/03/2014 (viene utilizzato il giorno della settimana 0)

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – giorno incluso

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente il totale delle vendite settimanali per il 2022 in una tabella chiamata `Sales`.
- Le date delle transazioni sono fornite in due campi: `year`, `week` e `sales`.
- Un caricamento precedente, utilizzato per creare una misura, `end_of_week`, utilizza la funzione `makeweekdate()` per restituire la data del venerdì di quella settimana nel formato MM/GG/AAAA.

Per dimostrare che la data restituita è un venerdì, l'espressione `end_of_week` viene anche racchiusa nella funzione `weekday()` per mostrare il giorno della settimana.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=0;
SET BrokenWeeks=0;
SET ReferenceDay=4;

Transactions:
    Load
```

```
        *,
        makeweekdate(transaction_year, transaction_week,4) as end_of_week,
        weekday(makeweekdate(transaction_year, transaction_week,4)) as week_day
        ;
Load * Inline [
transaction_year, transaction_week, sales
2022, 01, 10000
2022, 02, 11250
2022, 03, 9830
2022, 04, 14010
2022, 05, 28402
2022, 06, 9992
2022, 07, 7292
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `transaction_year`
- `transaction_week`
- `end_of_week`
- `week_day`

Tabella dei risultati

transaction_year	transaction_week	end_of_week	week_day
2022	01	01/07/2022	Fri
2022	02	01/14/2022	Fri
2022	03	01/21/2022	Fri
2022	04	01/28/2022	Fri
2022	05	02/04/2022	Fri
2022	06	02/11/2022	Fri
2022	07	02/18/2022	Fri

Il campo `end_of_week` viene creato nell'istruzione LOAD precedente utilizzando la funzione `makeweekdate()`. I campi `transaction_year`, `transaction_week` vengono passati alla funzione come argomenti dell'anno e della settimana. Il valore 4 viene utilizzato per l'argomento giorno.

La funzione combina e converte questi valori in un campo data, restituendo i risultati nel formato della variabile di sistema `DateFormat`.

La funzione `makeweekdate()` e i relativi argomenti sono anche avvolti in una funzione `weekday()` per restituire il campo `week_day`; e come si può vedere nella tabella precedente, il campo `week_day` mostra che queste date si verificano di venerdì.

Esempio 2 - giorno escluso

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente i totali delle vendite settimanali per il 2022 in una tabella chiamata `Sales`.
- Le date delle transazioni sono fornite in due campi: `year`, `week` e `sales`.
- Un caricamento precedente, che viene sfruttato per creare una misura, `first_day_of_week`, utilizzando la funzione `makeweekdate()`. Questo restituirà la data del lunedì di quella settimana nel formato MM/GG/AAAA.

Per dimostrare che la data restituita è un lunedì, l'espressione `first_day_of_week` viene anche racchiusa nella funzione `weekday()` per mostrare il giorno della settimana.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=0;
SET BrokenWeeks=0;
SET ReferenceDay=4;

Transactions:
    Load
        *,
        makeweekdate(transaction_year, transaction_week) as first_day_of_week,
        weekday(makeweekdate(transaction_year, transaction_week)) as week_day
        ;
Load * Inline [
transaction_year, transaction_week, sales
2022, 01, 10000
2022, 02, 11250
2022, 03, 9830
2022, 04, 14010
2022, 05, 28402
2022, 06, 9992
2022, 07, 7292
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `transaction_year`
- `transaction_week`

- first_day_of_week
- week_day

Tabella dei risultati

transaction_year	transaction_week	first_day_of_week	week_day
2022	01	01/03/2022	Mon
2022	02	01/10/2022	Mon
2022	03	01/17/2022	Mon
2022	04	01/24/2022	Mon
2022	05	01/31/2022	Mon
2022	06	02/07/2022	Mon
2022	07	02/14/2022	Mon

Il campo first_day_of_week viene creato nell'istruzione LOAD precedente utilizzando la funzione makeweekdate(). I parametri transaction_year e transaction_week vengono passati come argomenti della funzione, mentre il parametro giorno viene lasciato vuoto.

La funzione combina e converte questi valori in un campo data, restituendo i risultati nel formato della variabile di sistema DateFormat.

La funzione makeweekdate() e i relativi argomenti sono anche avvolti in una funzione weekday() che restituisce il campo week_day. Come si può vedere nella tabella precedente, il campo week_day restituisce Lunedì in tutti i casi, poiché questo parametro è stato lasciato vuoto nella funzione makeweekdate(), che ha come valore predefinito 0 (primo giorno della settimana), e il primo giorno della settimana è impostato su Lunedì dalla variabile di sistema FirstWeekDay.

Esempio 3 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente i totali delle vendite settimanali per il 2022 in una tabella chiamata Sales.
- Le date delle transazioni sono fornite in due campi: year, week e sales.

In questo esempio, si utilizzerà un oggetto grafico per creare una misura equivalente al calcolo end_of_week del primo esempio. Questa misura utilizzerà la funzione makeweekdate() per restituire la data del venerdì di quella settimana nel formato MM/GG/AAAA.

Per dimostrare che la data restituita è un venerdì, viene creata una seconda misura per restituire il giorno della settimana.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=0;
SET BrokenWeeks=0;
SET ReferenceDay=4;

Master_Calendar:
Load * Inline [
transaction_year, transaction_week, sales
2022, 01, 10000
2022, 02, 11250
2022, 03, 9830
2022, 04, 14010
2022, 05, 28402
2022, 06, 9992
2022, 07, 7292
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:
    - `transaction_year`
    - `transaction_week`
2. Per eseguire il calcolo equivalente a quello del `end_of_week`campo del primo esempio, creare la seguente misura:
   `=makeweekdate(transaction_year,transaction_week,4)`
3. Per calcolare il giorno della settimana per ogni transazione, creare la seguente misura:
   `=weekday(makeweekdate(transaction_year,transaction_week,4))`

Tabella dei risultati

transaction_ year	transaction_ week	=makeweekdate (transaction_ year,transaction_week,4)	=weekday(makeweekdate (transaction_year,transaction_ week,4))
2022	01	01/07/2022	Fri
2022	02	01/14/2022	Fri
2022	03	01/21/2022	Fri
2022	04	01/28/2022	Fri
2022	05	02/04/2022	Fri

transaction_ year	transaction_ week	=makeweekdate (transaction_ year,transaction_week,4)	=weekday(makeweekdate (transaction_year,transaction_ week,4))
2022	06	02/11/2022	Fri
2022	07	02/18/2022	Fri

Un campo equivalente a `end_of_week` viene creato nell'oggetto grafico come misura utilizzando la funzione `makeweekdate()`. I campi `transaction_year` e `transaction_week` vengono passati come argomenti dell'anno e della settimana. Il valore 4 viene utilizzato per l'argomento giorno.

La funzione combina e converte questi valori in un campo data, restituendo i risultati nel formato della variabile di sistema `DateFormat`.

Anche la funzione `makeweekdate()` e i relativi argomenti sono disposti in una funzione `weekday()` per restituire un calcolo equivalente a quello del campo `week_day` del primo esempio. Come si può vedere nella tabella precedente, l'ultima colonna a destra mostra che queste date si verificano di venerdì.

Esempio 4 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

In questo esempio, si crea un elenco di date contenente tutti i venerdì dell'anno 2022.

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=0;
SET BrokenWeeks=0;
SET ReferenceDay=4;

Calendar:
    load
           *,
        weekday(date) as weekday
    where year(date)=2022;
load
    makeweekdate(2022,recno()-2,4) as date
AutoGenerate 60;
```

Risultati

Tabella dei risultati

date	weekday
01/07/2022	Fri
01/14/2022	Fri
01/21/2022	Fri
01/28/2022	Fri
02/04/2022	Fri
02/11/2022	Fri
02/18/2022	Fri
02/25/2022	Fri
03/04/2022	Fri
03/11/2022	Fri
03/18/2022	Fri
03/25/2022	Fri
04/01/2022	Fri
04/08/2022	Fri
04/15/2022	Fri
04/22/2022	Fri
04/29/2022	Fri
05/06/2022	Fri
05/13/2022	Fri
05/20/2022	Fri
05/27/2022	Fri
06/03/2022	Fri
06/10/2022	Fri
06/17/2022	Fri
+ altre 27 righe	

La funzione `makeweekdate()` trova ogni venerdì del 2022. L'utilizzo di un parametro di settimana pari a -2 garantisce che non vengano perse delle date. Infine, un caricamento precedente crea un campo `weekday` aggiuntivo per chiarezza, per mostrare che ogni valore `date` è un venerdì.

minute

Questa funzione restituisce un numero intero che rappresenta il minuto in cui la frazione di **expression** viene interpretata come ora in base all'interpretazione numerica standard.

Sintassi:

```
minute(expression)
```

Tipo di dati restituiti: numero intero

Casi di utilizzo

La funzione `minute()` è utile quando si desidera confrontare le aggregazioni per minuto. Ad esempio, è possibile utilizzare la funzione per visualizzare la distribuzione del conteggio delle attività per minuto.

È possibile creare queste dimensioni anche nello script di caricamento utilizzando la funzione che consente di creare un campo in una tabella Calendario principale. In alternativa, possono essere utilizzate direttamente in un grafico come dimensione calcolata.

Esempi di funzioni

Esempio	Risultato
`minute ( '09:14:36' )`	Restituisce 14.
`minute ( '0.5555' )`	Restituisce 19 (poiché 0.5555 = 13:19:55).

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Variabile (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente transazioni per timestamp, caricato in una tabella chiamata `Transactions`.
- Viene utilizzata la variabile di sistema `TimeStamp` predefinita (`M/D/YYYY h:mm:ss[.fff] TT`).
- La creazione di un campo, `minute`, per calcolare quando sono avvenute le transazioni.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
    Load
        *,
        minute(timestamp) as minute
        ;
Load
*
Inline
[
id,timestamp,amount
9497,'2022-01-05 19:04:57',47.25,
9498,'2022-01-03 14:21:53',51.75,
9499,'2022-01-03 05:40:49',73.53,
9500,'2022-01-04 18:49:38',15.35,
9501,'2022-01-01 22:10:22',31.43,
9502,'2022-01-05 19:34:46',13.24,
9503,'2022-01-04 22:58:34',74.34,
9504,'2022-01-06 11:29:38',50.00,
9505,'2022-01-02 08:35:54',36.34,
9506,'2022-01-06 08:49:09',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `timestamp`
- `minute`

Tabella dei risultati

timestamp	minuto
2022-01-01 22:10:22	10
2022-01-02 08:35:54	35
2022-01-03 05:40:49	40
2022-01-03 14:21:53	21
2022-01-04 18:49:38	49

timestamp	minuto
2022-01-04 22:58:34	58
2022-01-05 19:04:57	4
2022-01-05 19:34:46	34
2022-01-06 08:49:09	49
2022-01-06 11:29:38	29

I valori nel campo `minute` sono creati usando la funzione `minute()` e trasferendo il valore `timestamp` come espressione nell'istruzione LOAD precedente.

Esempio 2 - Oggetto grafico (grafico)

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- Viene utilizzata la variabile di sistema `TimeStamp` predefinita (`M/D/YYYY h:mm:ss[.fff] TT`).

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. I valori `minute` sono calcolati mediante una misura in un oggetto grafico.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
Load
*
Inline
[
id,timestamp,amount
9497,'2022-01-05 19:04:57',47.25,
9498,'2022-01-03 14:21:53',51.75,
9499,'2022-01-03 05:40:49',73.53,
9500,'2022-01-04 18:49:38',15.35,
9501,'2022-01-01 22:10:22',31.43,
9502,'2022-01-05 19:34:46',13.24,
9503,'2022-01-04 22:58:34',74.34,
9504,'2022-01-06 11:29:38',50.00,
9505,'2022-01-02 08:35:54',36.34,
9506,'2022-01-06 08:49:09',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `timestamp`.

Creare la seguente misura:

```
=minute(timestamp)
```

Tabella dei risultati

timestamp	minuto
2022-01-01 22:10:22	10
2022-01-02 08:35:54	35
2022-01-03 05:40:49	40
2022-01-03 14:21:53	21
2022-01-04 18:49:38	49
2022-01-04 22:58:34	58
2022-01-05 19:04:57	4
2022-01-05 19:34:46	34
2022-01-06 08:49:09	49
2022-01-06 11:29:38	29

I valori per `minute` sono creati usando la funzione `minute()` e trasferendo il valore `timestamp` come espressione in una misura per l'oggetto grafico.

Esempio 3 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di timestamp, generato per rappresentare gli ingressi in un tornello.
- Informazioni con ogni `timestamp` e `id` corrispondente, che viene caricato in una tabella chiamata `Ticket_Barrier_Tracker`.
- Viene utilizzata la variabile di sistema `TimeStamp` predefinita (`M/D/YYYY h:mm:ss[.fff] TT`).

L'utente desidera un oggetto grafico che mostri, al minuto, il numero di ingressi al tornello.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

tmpTimeStampCreator:
       load
                *
    where year(date)=2022;
load
       date(recno()+makedate(2021,12,31)) as date
AutoGenerate 1;

join load
       maketime(floor(rand()*24),floor(rand()*59),floor(rand()*59)) as time
autogenerate 10000;

Ticket_Barrier_Tracker:
load
       recno() as id,
       timestamp(date + time) as timestamp
resident tmpTimeStampCreator;

drop table tmpTimeStampCreator;
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Creare una dimensione calcolata utilizzando la seguente espressione:
   `=minute(timestamp)`
3. Aggiungere la seguente misura di aggregazione per calcolare il conteggio totale degli ingressi:
   `=count(id)`
4. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

minute(timestamp)	=count(id)
0	174
1	171
2	175
3	165
4	188
5	176
6	158
7	187

minute(timestamp)	=count(id)
8	178
9	178
10	197
11	161
12	166
13	184
14	159
15	161
16	152
17	160
18	176
19	164
20	170
21	170
22	142
23	145
24	155
+ altre 35 righe	

month

Questa funzione restituisce un valore duale: il nome del mese come definito nella variabile di ambiente **MonthNames** e un numero intero compreso tra 1 e 12. Il numero del mese viene calcolato a partire dall'interpretazione della data dell'espressione in base all'interpretazione numerica standard.

La funzione restituisce il nome del mese nel formato del sistema `MonthName` variabile per una particolare data. È comunemente utilizzata per creare un campo dati come dimensione in un Calendario principale.

Sintassi:

```
month(expression)
```

Tipo di dati restituiti: numero intero

Esempi di funzioni

Esempio	Risultato
`month( 2012-10-12 )`	restituisce Oct
`month( 35648 )`	restituisce Aug poiché 35648 = 1997-08-06

Esempio 1 – set di dati DateFormat (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di date denominato `Master_Calendar`. La variabile di sistema `DateFormat` è impostata su GG/MM/AAAA.
- Un caricamento precedente che crea un campo aggiuntivo, denominato`month_name`, mediante la funzione `month()`.
- Un campo aggiuntivo, denominato `long_date`, che utilizza la funzione `date()` per esprimere la data completa.

Script di caricamento

```
SET DateFormat='DD/MM/YYYY';

Master_Calendar:
Load
        date,
        date(date,'dd-MMMM-YYYY') as long_date,
        month(date) as month_name
Inline
[
date
03/01/2022
03/02/2022
03/03/2022
03/04/2022
03/05/2022
03/06/2022
03/07/2022
03/08/2022
03/09/2022
03/10/2022
03/11/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `long_date`
- `month_name`

Tabella dei risultati

data	long_date	month_name
03/01/2022	03-January- 2022	Jan
03/02/2022	03-February- 2022	Feb
03/03/2022	03-March- 2022	Mar
03/04/2022	03-April- 2022	Apr
03/05/2022	03-May- 2022	May
03/06/2022	03-June- 2022	Jun
03/07/2022	03-July- 2022	Jul
03/08/2022	03-August- 2022	Aug
03/09/2022	03-September- 2022	Sep
03/10/2022	03-October- 2022	Oct
03/11/2022	03-November- 2022	Nov

Il nome del mese viene valutato correttamente dalla funzione `month()` nello script.

Esempio 2 – Date ANSI (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di date denominato `Master_Calendar`. Viene utilizzata la variabile di sistema DateFormat GG/MM/AAAA. Tuttavia, le date incluse nel set di dati sono nel formato data standard ANSI.
- Un caricamento precedente che crea un campo aggiuntivo, denominato `month_name`, mediante la funzione month().

- Un campo aggiuntivo, denominato `long_date`, che utilizza la funzione date() per esprimere la data completa.

Script di caricamento

```
SET DateFormat='DD/MM/YYYY';
Master_Calendar:
Load
       date,
       date(date,'dd-MMMM-YYYY') as long_date,
       month(date) as month_name

Inline
[
date
2022-01-11
2022-02-12
2022-03-13
2022-04-14
2022-05-15
2022-06-16
2022-07-17
2022-08-18
2022-09-19
2022-10-20
2022-11-21
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `long_date`
- `month_name`

Tabella dei risultati

data	long_date	month_name
03/11/2022	11-March- 2022	11
03/12/2022	12-March- 2022	12
03/13/2022	13-March- 2022	13
03/14/2022	14-March- 2022	14
03/15/2022	15-March- 2022	15
03/16/2022	16-March- 2022	16
03/17/2022	17-March- 2022	17

data	long_date	month_name
03/18/2022	18-March- 2022	18
03/19/2022	19-March- 2022	19
03/20/2022	20-March- 2022	20
03/21/2022	21-March- 2022	21

Il nome del mese viene valutato correttamente dalla funzione `month()` nello script.

Esempio 3 – Date non formattate (script)

Script di caricamento e risultati

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati di date denominato `Master_Calendar`. Viene utilizzata la variabile di sistema `DateFormat` GG/MM/AAAA.
- Un caricamento precedente che crea un campo aggiuntivo, denominato `month_name`, mediante la funzione `month()` .
- La data non formattata originale, denominata `unformatted_date`.
- Un campo aggiuntivo, denominato `long_date`, che utilizza la funzione `date()` per esprimere la data completa.

Script di caricamento

```
SET DateFormat='DD/MM/YYYY';

Master_Calendar:
Load
      unformatted_date,
      date(unformatted_date,'dd-MMMM-YYYY') as long_date,
      month(unformatted_date) as month_name

Inline
[
unformatted_date
44868
44898
44928
44958
44988
45018
45048
45078
45008
```

```
45038
45068
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `unformatted_date`
- `long_date`
- `month_name`

Tabella dei risultati

unformatted_date	long_date	month_name
44868	03-January- 2022	Jan
44898	03-February- 2022	Feb
44928	03-March- 2022	Mar
44958	03-April- 2022	Apr
44988	03-May- 2022	May
45018	03-June- 2022	Jun
45048	03-July- 2022	Jul
45078	03-August- 2022	Aug
45008	03-September- 2022	Sep
45038	03-October- 2022	Oct
45068	03-November- 2022	Nov

Il nome del mese viene valutato correttamente dalla funzione `month()` nello script.

Esempio 4 – Calcolo del mese di scadenza

Script di caricamento ed espressione del grafico

Panoramica

Aprire Editor caricamento dati, quindi aggiungere lo script di caricamento in basso in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati degli ordini effettuati a marzo denominato `Subscriptions`. La tabella contiene tre campi:

- id
- order_date
- importo

Script di caricamento

```
Subscriptions:
Load
        id,
        order_date,
        amount
Inline
[
id,order_date,amount
1,03/01/2022,231.24
2,03/02/2022,567.28
3,03/03/2022,364.28
4,03/04/2022,575.76
5,03/05/2022,638.68
6,03/06/2022,785.38
7,03/07/2022,967.46
8,03/08/2022,287.67
9,03/09/2022,764.45
10,03/10/2022,875.43
11,03/11/2022,957.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `order_date`.

Per calcolare il mese in cui scadrà un ordine, creare questa misura: `=month(order_date+180)`.

Tabella dei risultati

order_date	=month(order_date+180)
03/01/2022	Jul
03/02/2022	Aug
03/03/2022	Aug
03/04/2022	Sep
03/05/2022	Oct
03/06/2022	Nov
03/07/2022	Dec
03/08/2022	Jan

order_date	=month(order_date+180)
03/09/2022	Mar
03/10/2022	Apr
03/11/2022	May

La funzione `month()` determina correttamente che un ordine effettuato l'11 marzo scadrebbe a luglio.

monthend

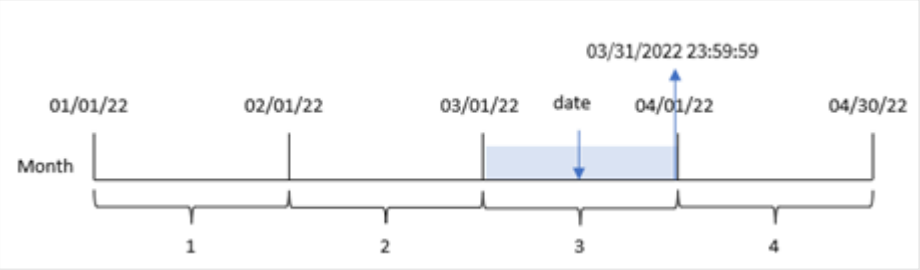
Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo dell'ultimo giorno del mese contenente `date`. Il formato di output predefinito sarà il formato `DateFormat` impostato nello script.

Sintassi:

```
MonthEnd(date[, period_no])
```

In altre parole, la funzione `monthend()` determina in quale mese cade la data. Quindi, restituisce un timestamp, nel formato data, per l'ultimo millisecondo di quel mese.

Schema della funzione `monthend`.



Casi di utilizzo

La funzione `monthend()` viene utilizzata come parte di un'espressione quando si desidera che il calcolo utilizzi la frazione del mese non ancora trascorsa. Ad esempio, se si vuole calcolare l'interesse totale non ancora maturato durante il mese.

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero che, se corrisponde a 0 o viene omesso, indica il mese contenente **date**. I valori negativi di **period_no** indicano i mesi precedenti, mentre i valori positivi indicano i mesi successivi.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`monthend('02/19/2012')`	Restituisce 02/29/2012 23:59:59.
`monthend('02/19/2001', -1)`	Restituisce 01/31/2001 23:59:59.

Esempio 1 – Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- Un campo data nel formato `MM/DD/YYYY` della variabile di sistema `DateFormat`.
- Un'istruzione LOAD precedente contenente:
  - La funzione `monthend()` impostata come campo, '`end_of_month`'.
  - La funzione `timestamp` impostata come campo, '`end_of_month_timestamp`'.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    monthend(date) as end_of_month,
    timestamp(monthend(date)) as end_of_month_timestamp
    ;
```

```
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `end_of_month`
- `end_of_month_timestamp`

Tabella dei risultati

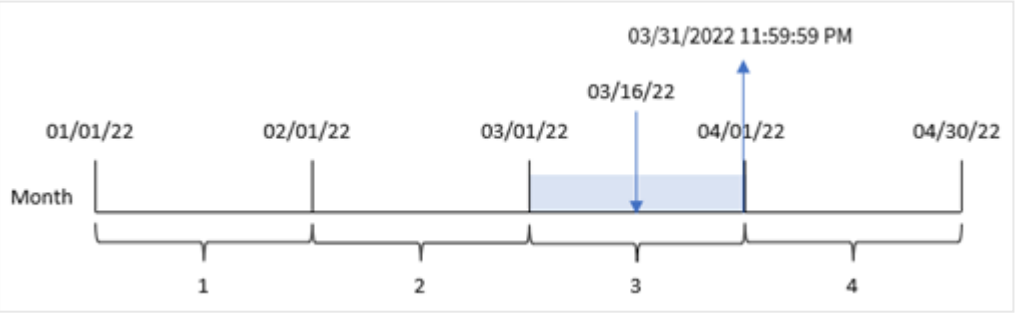
id	date	end_of_month	end_of_month_timestamp
8188	1/7/2022	01/31/2022	1/31/2022 11:59:59 PM
8189	1/19/2022	01/31/2022	1/31/2022 11:59:59 PM
8190	2/5/2022	02/28/2022	2/28/2022 11:59:59 PM
8191	2/28/2022	02/28/2022	2/28/2022 11:59:59 PM
8192	3/16/2022	03/31/2022	3/31/2022 11:59:59 PM
8193	4/1/2022	04/30/2022	4/30/2022 11:59:59 PM
8194	5/7/2022	05/31/2022	5/31/2022 11:59:59 PM

id	date	end_of_month	end_of_month_timestamp
8195	5/16/2022	05/31/2022	5/31/2022 11:59:59 PM
8196	6/15/2022	06/30/2022	6/30/2022 11:59:59 PM
8197	6/26/2022	06/30/2022	6/30/2022 11:59:59 PM
8198	7/9/2022	07/31/2022	7/31/2022 11:59:59 PM
8199	7/22/2022	07/31/2022	7/31/2022 11:59:59 PM
8200	7/23/2022	07/31/2022	7/31/2022 11:59:59 PM
8201	7/27/2022	07/31/2022	7/31/2022 11:59:59 PM
8202	8/2/2022	08/31/2022	8/31/2022 11:59:59 PM
8203	8/8/2022	08/31/2022	8/31/2022 11:59:59 PM
8204	8/19/2022	08/31/2022	8/31/2022 11:59:59 PM
8205	9/26/2022	09/30/2022	9/30/2022 11:59:59 PM
8206	10/14/2022	10/31/2022	10/31/2022 11:59:59 PM
8207	10/29/2022	10/31/2022	10/31/2022 11:59:59 PM

Il campo '`end_of_month`' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `monthend()` e trasferendo il campo data come argomento della funzione.

La funzione `monthend()` identifica in quale mese cade il valore della data, restituendo un timestamp per l'ultimo millisecondo di quel mese.

Schema della funzione `monthend` con marzo come mese selezionato.



La transazione 8192 è avvenuta il 16 marzo. La funzione `monthend()` restituisce l'ultimo millisecondo di quel mese, ovvero il 31 marzo alle 23:59:59.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, si tratta di creare un campo, 'previous_month_end', che restituisca il timestamp della fine del mese precedente la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    monthend(date,-1) as previous_month_end,
    timestamp(monthend(date,-1)) as previous_month_end_timestamp
    ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

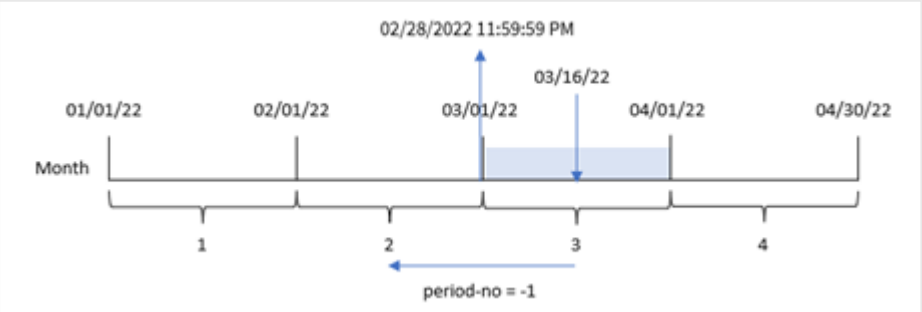
- id
- date
- previous_month_end
- previous_month_end_timestamp

Tabella dei risultati

id	date	previous_month_end	previous_month_end_timestamp
8188	1/7/2022	12/31/2021	12/31/2021 11:59:59 PM
8189	1/19/2022	12/31/2021	12/31/2021 11:59:59 PM
8190	2/5/2022	01/31/2022	1/31/2022 11:59:59 PM
8191	2/28/2022	01/31/2022	1/31/2022 11:59:59 PM
8192	3/16/2022	02/28/2022	2/28/2022 11:59:59 PM
8193	4/1/2022	03/31/2022	3/31/2022 11:59:59 PM
8194	5/7/2022	04/30/2022	4/30/2022 11:59:59 PM
8195	5/16/2022	04/30/2022	4/30/2022 11:59:59 PM
8196	6/15/2022	05/31/2022	5/31/2022 11:59:59 PM
8197	6/26/2022	05/31/2022	5/31/2022 11:59:59 PM
8198	7/9/2022	06/30/2022	6/30/2022 11:59:59 PM
8199	7/22/2022	06/30/2022	6/30/2022 11:59:59 PM
8200	7/23/2022	06/30/2022	6/30/2022 11:59:59 PM
8201	7/27/2022	06/30/2022	6/30/2022 11:59:59 PM
8202	8/2/2022	07/31/2022	7/31/2022 11:59:59 PM
8203	8/8/2022	07/31/2022	7/31/2022 11:59:59 PM
8204	8/19/2022	07/31/2022	7/31/2022 11:59:59 PM
8205	9/26/2022	08/31/2022	8/31/2022 11:59:59 PM
8206	10/14/2022	09/30/2022	9/30/2022 11:59:59 PM
8207	10/29/2022	09/30/2022	9/30/2022 11:59:59 PM

La funzione `monthend()` identifica innanzitutto il mese in cui avvengono le transazioni, in quanto un valore `period_no` di `-1` viene utilizzato come argomento di offset. Si sposta poi un mese prima e identifica il millisecondo finale di tale mese.

Schema della funzione `monthend` con la variabile period_no.



La transazione 8192 è avvenuta il 16 marzo. La funzione `monthend()` identifica che il mese precedente la transazione è avvenuta in febbraio. Quindi restituisce l'ultimo millisecondo di quel mese, il 28 febbraio alle 23:59:59.

Esempio 3 – Esempio di grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, il set di dati è invariato e caricato nell'app. Il compito è quello di creare un calcolo che restituisca un timestamp per la fine del mese in cui sono avvenute le transazioni come misura in un grafico dell'app.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `id`

Per calcolare la data di fine del mese in cui avviene una transazione, creare le seguenti misure:

- `=monthend(date)`
- `=timestamp(monthend(date))`

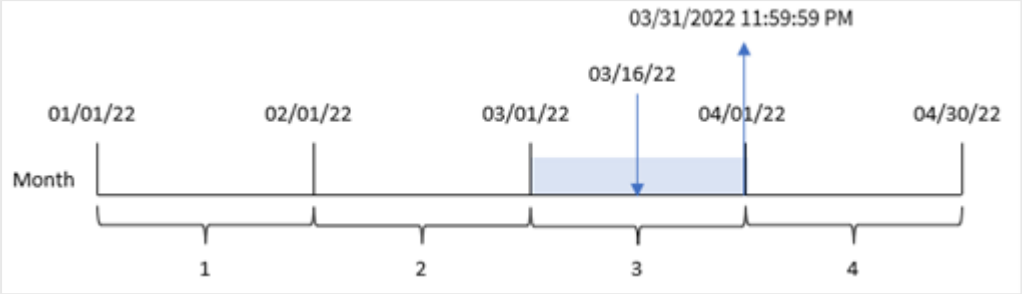
Tabella dei risultati

id	date	=monthend(date)	=timestamp(monthend(date))
8188	10/14/2022	10/31/2022	10/31/2022 11:59:59 PM
8189	10/29/2022	10/31/2022	10/31/2022 11:59:59 PM
8190	9/26/2022	09/30/2022	9/30/2022 11:59:59 PM
8191	8/2/2022	08/31/2022	8/31/2022 11:59:59 PM
8192	8/8/2022	08/31/2022	8/31/2022 11:59:59 PM
8193	8/19/2022	08/31/2022	8/31/2022 11:59:59 PM
8194	7/9/2022	07/31/2022	7/31/2022 11:59:59 PM
8195	7/22/2022	07/31/2022	7/31/2022 11:59:59 PM
8196	7/23/2022	07/31/2022	7/31/2022 11:59:59 PM
8197	7/27/2022	07/31/2022	7/31/2022 11:59:59 PM
8198	6/15/2022	06/30/2022	6/30/2022 11:59:59 PM
8199	6/26/2022	06/30/2022	6/30/2022 11:59:59 PM
8200	5/7/2022	05/31/2022	5/31/2022 11:59:59 PM
8201	5/16/2022	05/31/2022	5/31/2022 11:59:59 PM
8202	4/1/2022	04/30/2022	4/30/2022 11:59:59 PM
8203	3/16/2022	03/31/2022	3/31/2022 11:59:59 PM
8204	2/5/2022	02/28/2022	2/28/2022 11:59:59 PM
8205	2/28/2022	02/28/2022	2/28/2022 11:59:59 PM
8206	1/7/2022	01/31/2022	1/31/2022 11:59:59 PM
8207	1/19/2022	01/31/2022	1/31/2022 11:59:59 PM

La misura '`end_of_month`' viene creata nel grafico mediante l'utilizzo della funzione `monthend()` e trasferendo il campo data come argomento della funzione.

La funzione `monthend()` identifica in quale mese cade il valore della data, restituendo un timestamp per l'ultimo millisecondo di quel mese.

Schema della funzione `monthend` con la variabile period_no.



La transazione 8192 è avvenuta il 16 marzo. La funzione `monthend()` restituisce l'ultimo millisecondo di quel mese, ovvero il 31 marzo alle 23:59:59.

Esempio 4 – Scenario

Script di caricamento e risultati

Panoramica

In questo esempio, un set di dati è caricato in una tabella denominata "`Employee_Expenses`". La tabella contiene i seguenti campi:

- ID dipendenti
- Nomi dipendenti
- La media delle richieste di rimborso spese giornaliero di ciascun dipendente.

L'utente finale desidera un grafico che visualizzi, in base all'id e al nome del dipendente, la spesa stimata per il resto del mese.

Script di caricamento

```
Employee_Expenses:
Load
*
Inline
[
employee_id,employee_name,avg_daily_claim
182,Mark, $15
183,Deryck, $12.5
184,Dexter, $12.5
185,Sydney,$27
186,Agatha,$18
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- employee_id
- employee_name

Per calcolare gli interessi accumulati, creare questa misura:

```
=floor(monthend(today(1),0)-today(1))*avg_daily_claim
```

Questa misura è dinamica e produrrà risultati diversi a seconda della data di caricamento dei dati.

Impostare la misura **Formattazione numero** su **Denaro**.

Tabella dei risultati

employee_id	employee_name	=floor(monthend(today(1),0)-today(1))*avg_daily_claim
182	Contrassegno	$30.00
183	Deryck	$25.00
184	Dexter	$25.00
185	Sydney	$54.00
186	Agatha	$36.00

La funzione `monthend()` restituisce la data di fine del mese corrente utilizzando come unico argomento la data odierna. L'espressione restituisce il numero di giorni rimanenti nel mese sottraendo la data odierna dalla data di fine mese.

Questo valore viene quindi moltiplicato per la media delle richieste di rimborso spese giornaliere di ciascun dipendente per calcolare il valore stimato delle richieste che ogni dipendente dovrebbe presentare durante il periodo rimanente del mese.

monthname

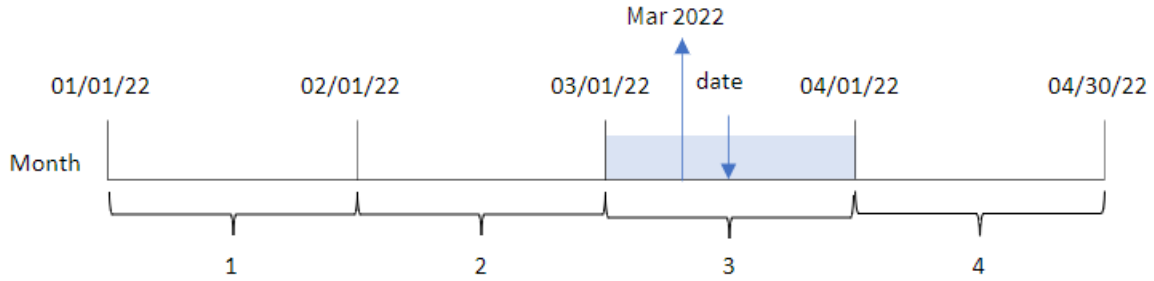
Questa funzione restituisce un valore di visualizzazione che mostra il mese (formattato in base alla variabile di script **MonthNames**) e l'anno il cui valore numerico sottostante corrisponde a un indicatore temporale recante il primo millisecondo del primo giorno del mese.

Sintassi:

```
MonthName(date[, period_no])
```

Tipo di dati restituiti: duale

Schema della funzione monthname



Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero che, se corrisponde a 0 o viene omesso, indica il mese contenente **date**. I valori negativi di **period_no** indicano i mesi precedenti, mentre i valori positivi indicano i mesi successivi.

Esempi di funzioni

Esempio	Risultato
`monthname('10/19/2013')`	Restituisce Oct 2013
`monthname('10/19/2013', -1)`	Restituisce Sep 2013

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `transaction_month`, che restituisce il mese in cui sono state effettuate le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';

Transactions:
    Load
        *,
        monthname(date) as transaction_month
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

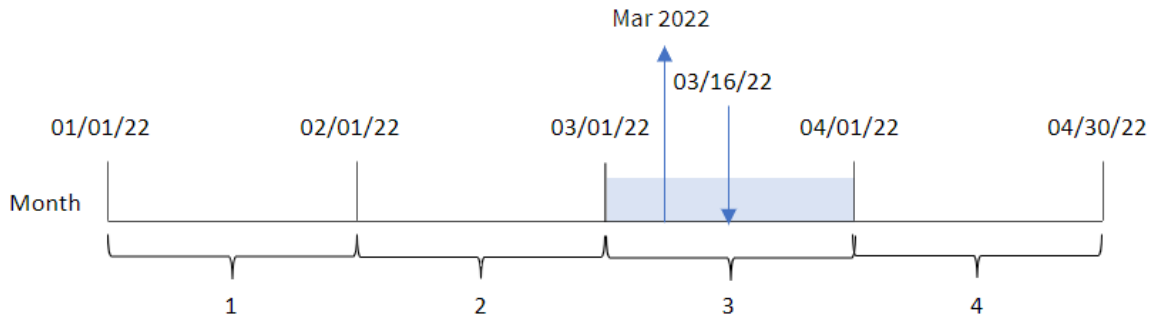
- `date`
- `transaction_month`

Tabella dei risultati

data	transaction_month
1/7/2022	Gen 2022
1/19/2022	Gen 2022
2/5/2022	Feb 2022
2/28/2022	Feb 2022
3/16/2022	Mar 2022
4/1/2022	Apr 2022
5/7/2022	May 2022
5/16/2022	May 2022
6/15/2022	Jun 2022
6/26/2022	Jun 2022
7/9/2022	Jul 2022
7/22/2022	Jul 2022
7/23/2022	Jul 2022
7/27/2022	Jul 2022
8/2/2022	Aug 2022
8/8/2022	Aug 2022
8/19/2022	Aug 2022
9/26/2022	Sep 2022
10/14/2022	Oct 2022
10/29/2022	Oct 2022

Il campo `transaction_month` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `monthname()` e trasferendo il campo `date` come argomento della funzione.

Schema della funzione monthname, esempio di base



La funzione `monthname()` identifica la transazione 8192 avvenuta nel marzo 2022 e restituisce questo valore utilizzando la variabile di sistema `MonthNames`.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati inline e lo stesso scenario del primo esempio.
- La creazione di un campo, `transaction_previous_month`, che restituisce data e ora per la fine del mese prima che fosse effettuata la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';

Transactions:
    Load
        *,
        monthname(date,-1) as transaction_previous_month
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
```

```
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `transaction_previous_month`

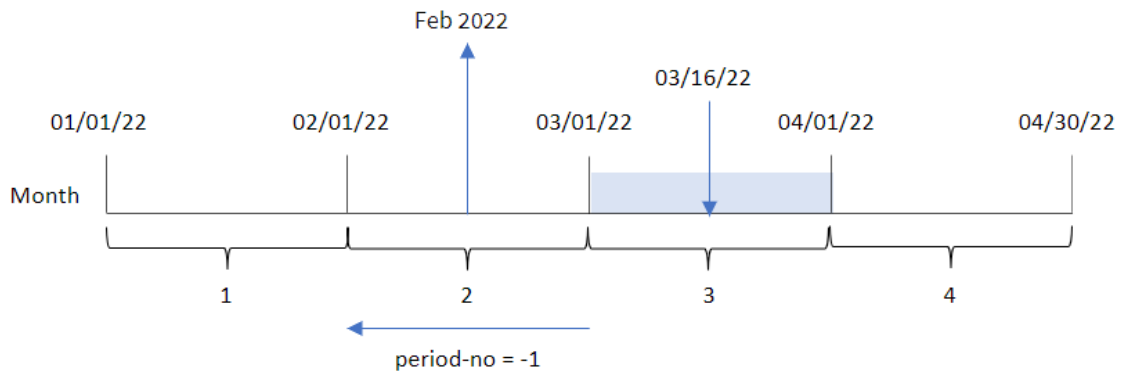
Tabella dei risultati

data	transaction_previous_month
1/7/2022	Dic 2021
1/19/2022	Dic 2021
2/5/2022	Gen 2022
2/28/2022	Gen 2022
3/16/2022	Feb 2022
4/1/2022	Mar 2022
5/7/2022	Apr 2022
5/16/2022	Apr 2022
6/15/2022	May 2022
6/26/2022	May 2022
7/9/2022	Jun 2022
7/22/2022	Jun 2022
7/23/2022	Jun 2022
7/27/2022	Jun 2022
8/2/2022	Jul 2022
8/8/2022	Jul 2022
8/19/2022	Jul 2022

data	transaction_previous_month
9/26/2022	Aug 2022
10/14/2022	Sep 2022
10/29/2022	Sep 2022

In questo caso, poiché il valore `period_no` di -1 è stato utilizzato come argomento offset nella funzione `monthname()`, la funzione per prima cosa identifica il mese in cui avvengono le transazioni. Quindi, passa a al mese anteriore e restituisce il nome del mese e l'anno.

Schema della funzione monthname, esempio di period_no



La transazione 8192 è avvenuta il 16 marzo. La funzione `monthname()` identifica che il mese precedente all'esecuzione della transazione è febbraio, quindi restituisce il mese nel formato della variabile di sistema `MonthNames`, insieme all'anno 2022.

Esempio 3 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati inline e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce data e ora per la fine del mese in cui sono avvenute le trasnazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';

Transactions:
Load
```

```
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`date`.

Creare la seguente misura:

```
=monthname(date)
```

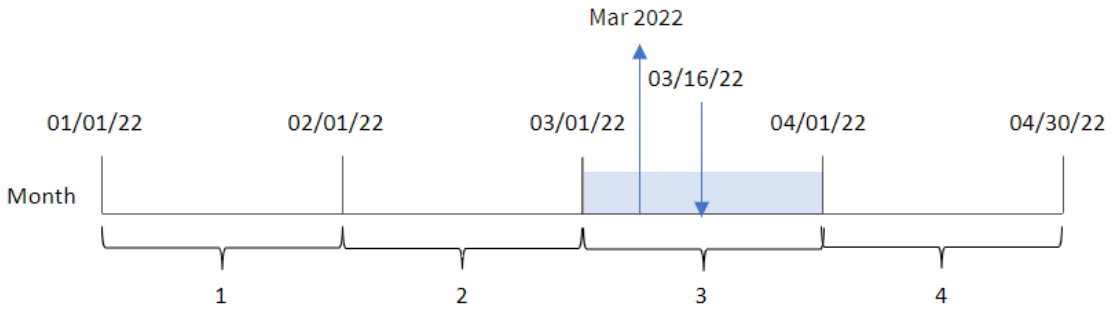
Tabella dei risultati

data	=monthname(date)
1/7/2022	Gen 2022
1/19/2022	Gen 2022
2/5/2022	Feb 2022
2/28/2022	Feb 2022
3/16/2022	Mar 2022
4/1/2022	Apr 2022
5/7/2022	May 2022
5/16/2022	May 2022
6/15/2022	Jun 2022

data	=monthname(date)
6/26/2022	Jun 2022
7/9/2022	Jul 2022
7/22/2022	Jul 2022
7/23/2022	Jul 2022
7/27/2022	Jul 2022
8/2/2022	Aug 2022
8/8/2022	Aug 2022
8/19/2022	Aug 2022
9/26/2022	Sep 2022
10/14/2022	Oct 2022
10/29/2022	Oct 2022

La misura `month_name` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `monthname()` e trasferendo il campo `date` come argomento della funzione.

Schema della funzione monthname, esempio di oggetto grafico



La funzione `monthname()` identifica la transazione 8192 avvenuta nel marzo 2022 e restituisce questo valore utilizzando la variabile di sistema `MonthNames`.

monthsend

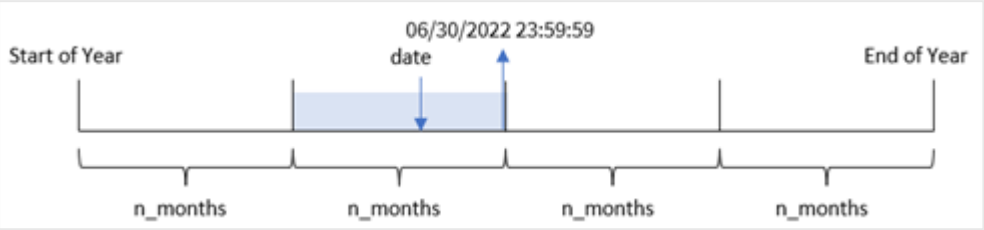
Questa funzione restituisce un valore corrispondente a un timestamp recante l'ultimo millisecondo del mese, del bimestre, del trimestre, del quadrimestre o del semestre contenente una data di base. È inoltre possibile individuare il timestamp per la fine di un periodo di tempo precedente o successivo. Il formato di output predefinito è il formato `DateFormat` impostato nello script.

Sintassi:

```
MonthsEnd(n_months, date[, period_no [, first_month_of_year]])
```

Tipo di dati restituiti: duale

Schema della funzione `monthsend`.



Argomenti

Argomento	Descrizione
n_months	Il numero di mesi che definisce il periodo. Un numero intero o un'espressione la cui risoluzione è un numero intero corrispondente a: 1 (equivalente alla funzione inmonth ()), 2 (bimestre), 3 (equivalente alla funzione inquarter()), 4 (quadrimestre) o 6 (semestre).
date	La data o la data e ora da valutare.
period_no	Il periodo può essere differito mediante **period_no**, un numero intero, o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica il periodo che contiene **base_date**. I valori negativi di **period_no** indicano i periodi precedenti, mentre i valori positivi indicano i periodi successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

La funzione `monthsend()` divide l'anno in segmenti in base all'argomento `n_months` fornito. Quindi valuta in quale segmento rientra ogni data fornita e restituisce l'ultimo millisecondo, in formato data, di quel segmento. La funzione può restituire il timestamp finale dei segmenti precedenti o successivi e ridefinire il primo mese dell'anno.

I seguenti segmenti dell'anno sono disponibili nella funzione come argomenti di `n_month`.

Argomenti `n_month`

Periodo	Numero di mesi
mese	1
bimestre	2
trimestre	3
quattro mesi	4
semestre	6

Casi di utilizzo

La funzione `monthsend()` viene utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione di mese trascorsa fino a quel momento. L'utente ha la possibilità, tramite una variabile, di selezionare il periodo di sua scelta. Ad esempio, il `monthsend()` può fornire una variabile di input che consenta all'utente di calcolare il totale degli interessi non ancora maturati durante il mese, il trimestre o il semestre.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`monthsend(4, '07/19/2013')`	Restituisce 08/31/2013.
`monthsend(4, '10/19/2013', -1)`	Restituisce 08/31/2013.
`monthsend(4, '10/19/2013', 0, 2)`	Restituisce 01/31/2014. Poiché l'inizio dell'anno corrisponde al mese 2.

Esempio 1 - Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022 viene caricato in una tabella denominata '`Transactions`'.
- Un campo data fornito nel formato (`(MM/DD/YYYY)`) della variabile di sistema `DateFormat`.
- Un'istruzione LOAD precedente contenente:

- La funzione `monthsend` impostata come campo, '`bi_monthly_end`'. Questo raggruppa le transazioni in segmenti bimestrali.
- La funzione `timestamp` che restituisce il timestamp iniziale del segmento per ogni transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    monthsend(2,date) as bi_monthly_end,
    timestamp(monthsend(2,date)) as bi_monthly_end_timestamp
    ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

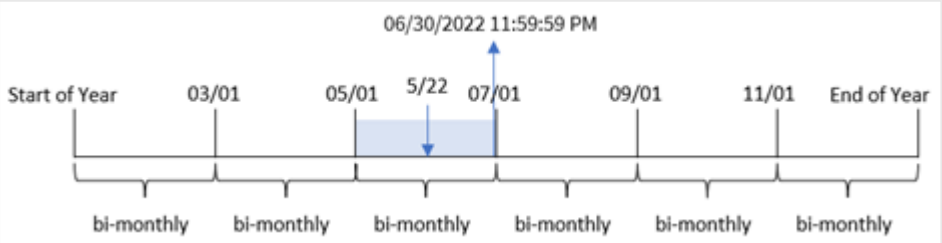
- `id`
- `date`
- `bi_monthly_end`
- `bi_monthly_end_timestamp`

Tabella dei risultati

id	date	bi_monthly_end	bi_monthly_end_timestamp
8188	1/7/2022	02/28/2022	2/28/2022 11:59:59 PM
8189	1/19/2022	02/28/2022	2/28/2022 11:59:59 PM
8190	2/5/2022	02/28/2022	2/28/2022 11:59:59 PM
8191	2/28/2022	02/28/2022	2/28/2022 11:59:59 PM
8192	3/16/2022	04/30/2022	4/30/2022 11:59:59 PM
8193	4/1/2022	04/30/2022	4/30/2022 11:59:59 PM
8194	5/7/2022	06/30/2022	6/30/2022 11:59:59 PM
8195	5/22/2022	06/30/2022	6/30/2022 11:59:59 PM
8196	6/15/2022	06/30/2022	6/30/2022 11:59:59 PM
8197	6/26/2022	06/30/2022	6/30/2022 11:59:59 PM
8198	7/9/2022	08/31/2022	8/31/2022 11:59:59 PM
8199	7/22/2022	08/31/2022	8/31/2022 11:59:59 PM
8200	7/23/2022	08/31/2022	8/31/2022 11:59:59 PM
8201	7/27/2022	08/31/2022	8/31/2022 11:59:59 PM
8202	8/2/2022	08/31/2022	8/31/2022 11:59:59 PM
8203	8/8/2022	08/31/2022	8/31/2022 11:59:59 PM
8204	8/19/2022	08/31/2022	8/31/2022 11:59:59 PM
8205	9/26/2022	10/31/2022	10/31/2022 11:59:59 PM
8206	10/14/2022	10/31/2022	10/31/2022 11:59:59 PM
8207	10/29/2022	10/31/2022	10/31/2022 11:59:59 PM

Il campo '`bi_monthly_end`' viene creato nell'istruzione LOAD precedente utilizzando la funzione `monthsend()`. Il primo argomento fornito è `2`, dividendo l'anno in segmenti bimestrali. Il primo argomento identifica il campo da valutare.

Schema della funzione `monthsend` con segmenti bimestrali.



La transazione 8195 avviene il 22 maggio. La funzione `monthsend()` divide inizialmente l'anno in segmenti bimestrali. La transazione 8195 rientra nel segmento tra il 1 maggio e il 30 giugno. Di conseguenza, la funzione restituisce l'ultimo millisecondo di questo segmento, il 30/06/2022 alle 23:59:59.

Esempio 2 - period_no

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, il compito è quello di creare un campo, '`prev_bi_monthly_end`', che restituisca il primo millisecondo del segmento bimestrale prima che avvenga la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    monthsend(2,date,-1) as prev_bi_monthly_end,
    timestamp(monthsend(2,date,-1)) as prev_bi_monthly_end_timestamp
    ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

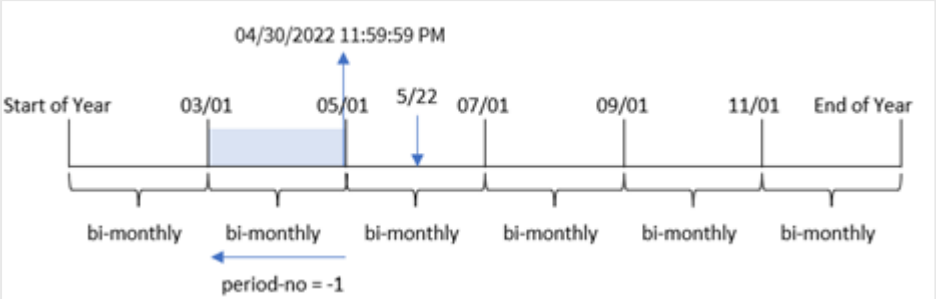
- `id`
- `date`
- `prev_bi_monthly_end`
- `prev_bi_monthly_end_timestamp`

Tabella dei risultati

id	date	prev_bi_monthly_end	prev_bi_monthly_end_timestamp
8188	1/7/2022	12/31/2021	12/31/2021 11:59:59 PM
8189	1/19/2022	12/31/2021	12/31/2021 11:59:59 PM
8190	2/5/2022	12/31/2021	12/31/2021 11:59:59 PM
8191	2/28/2022	12/31/2021	12/31/2021 11:59:59 PM
8192	3/16/2022	02/28/2022	2/28/2022 11:59:59 PM
8193	4/1/2022	02/28/2022	2/28/2022 11:59:59 PM
8194	5/7/2022	04/30/2022	4/30/2022 11:59:59 PM
8195	5/22/2022	04/30/2022	4/30/2022 11:59:59 PM
8196	6/15/2022	04/30/2022	4/30/2022 11:59:59 PM
8197	6/26/2022	04/30/2022	4/30/2022 11:59:59 PM
8198	7/9/2022	06/30/2022	6/30/2022 11:59:59 PM
8199	7/22/2022	06/30/2022	6/30/2022 11:59:59 PM
8200	7/23/2022	06/30/2022	6/30/2022 11:59:59 PM
8201	7/27/2022	06/30/2022	6/30/2022 11:59:59 PM
8202	8/2/2022	06/30/2022	6/30/2022 11:59:59 PM
8203	8/8/2022	06/30/2022	6/30/2022 11:59:59 PM
8204	8/19/2022	06/30/2022	6/30/2022 11:59:59 PM
8205	9/26/2022	08/31/2022	8/31/2022 11:59:59 PM
8206	10/14/2022	08/31/2022	8/31/2022 11:59:59 PM
8207	10/29/2022	08/31/2022	8/31/2022 11:59:59 PM

Utilizzando `-1` come argomento `period_no` della funzione `monthsend()`, dopo aver inizialmente diviso un anno in segmenti bimestrali, la funzione restituisce l'ultimo millisecondo del segmento bimestrale precedente al momento in cui avviene una transazione.

Schema della funzione `monthsend` che restituisce il segmento bimestrale precedente.



La transazione 8195 è stata effettuata nel segmento tra maggio e giugno. Di conseguenza, il segmento bimestrale precedente era compreso tra il 1° marzo e il 30 aprile e quindi la funzione restituisce l'ultimo millisecondo di questo segmento, il 30/04/2022 alle 23:59:59.

Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, il criterio organizzativo prevede che aprile sia il primo mese dell'anno finanziario.

Creare un campo, `bi_monthly_end`, che raggruppa le transazioni in segmenti bimestrali e restituisce il timestamp dell'ultimo millisecondo di quel segmento per ogni transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
    *,
    monthsend(2,date,0,4) as bi_monthly_end,
    timestamp(monthsend(2,date,0,4)) as bi_monthly_end_timestamp
    ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
```

```
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `bi_monthly_end`
- `bi_monthly_end_timestamp`

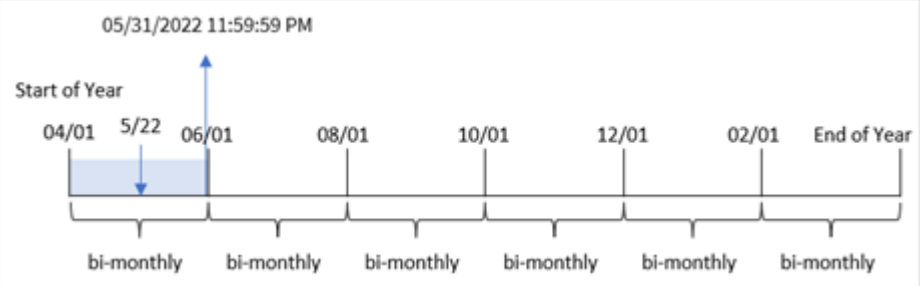
Tabella dei risultati

id	date	bi_monthly_end	bi_monthly_end_timestamp
8188	1/7/2022	01/31/2022	1/31/2022 11:59:59 PM
8189	1/19/2022	01/31/2022	1/31/2022 11:59:59 PM
8190	2/5/2022	03/31/2022	3/31/2022 11:59:59 PM
8191	2/28/2022	03/31/2022	3/31/2022 11:59:59 PM
8192	3/16/2022	03/31/2022	3/31/2022 11:59:59 PM
8193	4/1/2022	05/31/2022	5/31/2022 11:59:59 PM
8194	5/7/2022	05/31/2022	5/31/2022 11:59:59 PM
8195	5/22/2022	05/31/2022	5/31/2022 11:59:59 PM
8196	6/15/2022	07/31/2022	7/31/2022 11:59:59 PM
8197	6/26/2022	07/31/2022	7/31/2022 11:59:59 PM
8198	7/9/2022	07/31/2022	7/31/2022 11:59:59 PM
8199	7/22/2022	07/31/2022	7/31/2022 11:59:59 PM
8200	7/23/2022	07/31/2022	7/31/2022 11:59:59 PM
8201	7/27/2022	07/31/2022	7/31/2022 11:59:59 PM
8202	8/2/2022	09/30/2022	9/30/2022 11:59:59 PM
8203	8/8/2022	09/30/2022	9/30/2022 11:59:59 PM

id	date	bi_monthly_end	bi_monthly_end_timestamp
8204	8/19/2022	09/30/2022	9/30/2022 11:59:59 PM
8205	9/26/2022	09/30/2022	9/30/2022 11:59:59 PM
8206	10/14/2022	11/30/2022	11/30/2022 11:59:59 PM
8207	10/29/2022	11/30/2022	11/30/2022 11:59:59 PM

Utilizzando `4` come argomento `first_month_of_year` nella funzione `monthsend()`, la funzione inizia l'anno il 1° aprile, quindi divide l'anno in segmenti bimestrali: Apr-Mag, Giu-Lug, Ago-Set, Ott-Nov, Dic-Gen, Feb-Mar.

Schema della funzione `monthsend` con il primo mese dell'anno impostato come aprile



La transazione 8195 è avvenuta il 22 maggio e rientra nel segmento tra il 1 aprile e il 31 maggio. Di conseguenza, la funzione restituisce l'ultimo millisecondo di questo segmento, il 31/05/2022 alle 23:59:59.

Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'app.

In questo esempio, si tratta di creare un calcolo che raggruppi le transazioni in segmenti bimestrali e restituisca il timestamp dell'ultimo millisecondo del segmento per ciascuna transazione come misura in un oggetto grafico di un'app.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,2/19/2022,37.23
```

```
8189,3/7/2022,17.17
8190,3/30/2022,88.27
8191,4/5/2022,57.42
8192,4/16/2022,53.80
8193,5/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

`date`

Per ottenere il timestamp dell'ultimo millisecondo del segmento bimestrale in cui è avvenuta la transazione, creare le seguenti misure:

- `=monthsEnd(2,date)`
- `=timestamp(monthsend(2,date))`

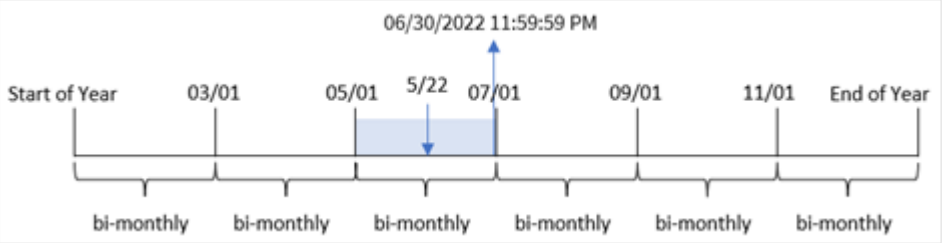
Tabella dei risultati

id	date	=monthsend(2,date)	=timestamp(monthsend(2,date))
8188	1/7/2022	02/28/2022	2/28/2022 11:59:59 PM
8189	1/19/2022	02/28/2022	2/28/2022 11:59:59 PM
8190	2/5/2022	02/28/2022	2/28/2022 11:59:59 PM
8191	2/28/2022	02/28/2022	2/28/2022 11:59:59 PM
8192	3/16/2022	04/30/2022	4/30/2022 11:59:59 PM
8193	4/1/2022	04/30/2022	4/30/2022 11:59:59 PM
8194	5/7/2022	06/30/2022	6/30/2022 11:59:59 PM
8195	5/22/2022	06/30/2022	6/30/2022 11:59:59 PM
8196	6/15/2022	06/30/2022	6/30/2022 11:59:59 PM
8197	6/26/2022	06/30/2022	6/30/2022 11:59:59 PM

id	date	=monthsend(2,date)	=timestamp(monthsend(2,date))
8198	7/9/2022	08/31/2022	8/31/2022 11:59:59 PM
8199	7/22/2022	08/31/2022	8/31/2022 11:59:59 PM
8200	7/23/2022	08/31/2022	8/31/2022 11:59:59 PM
8201	7/27/2022	08/31/2022	8/31/2022 11:59:59 PM
8202	8/2/2022	08/31/2022	8/31/2022 11:59:59 PM
8203	8/8/2022	08/31/2022	8/31/2022 11:59:59 PM
8204	8/19/2022	08/31/2022	8/31/2022 11:59:59 PM
8205	9/26/2022	10/31/2022	10/31/2022 11:59:59 PM
8206	10/14/2022	10/31/2022	10/31/2022 11:59:59 PM
8207	10/29/2022	10/31/2022	10/31/2022 11:59:59 PM

Il campo '`bi_monthly_end`' viene creato come misura nell'oggetto grafico mediante l'utilizzo della funzione `monthsend()`. Il primo argomento fornito è `2`, che divide l'anno in segmenti bimestrali. Il primo argomento identifica il campo da valutare.

Schema della funzione `monthsend` con segmenti bimestrali.



La transazione 8195 avviene il 22 maggio. La funzione `monthsend()` divide inizialmente l'anno in segmenti bimestrali. La transazione 8195 rientra nel segmento tra il 1 maggio e il 30 giugno. Di conseguenza, la funzione restituisce il primo millisecondo di questo segmento, il 30/06/2022 alle 23:59:59.

Esempio 5 – Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

In questo esempio, un set di dati è caricato in una tabella denominata "`Employee_Expenses`". La tabella contiene i seguenti campi:

- ID dipendenti
- Nomi dipendenti

- La media delle richieste di rimborso spese giornaliero di ciascun dipendente.

L'utente finale desidera un grafico che visualizzi, in base all'id e al nome del dipendente, la spesa stimata per il resto di un periodo di propria scelta. L'esercizio finanziario inizia a gennaio.

Script di caricamento

```
SET vPeriod = 1;

Employee_Expenses:
Load
*
Inline
[
employee_id,employee_name,avg_daily_claim
182,Mark, $15
183,Deryck, $12.5
184,Dexter, $12.5
185,Sydney,$27
186,Agatha,$18
];
```

Risultati

Caricare i dati e aprire un nuovo foglio.

All'inizio dello script di caricamento, viene creata una variabile `vPeriod`, legata al controllo dell'input variabile.

Procedere come segue:

1. Nel pannello delle risorse, fare clic **su Oggetti personalizzati**.
2. Selezionare **Qlik Dashboard bundle** e creare un oggetto **Input variabile**.
3. Immettere un titolo per l'oggetto grafico.
4. In **Variabile**, selezionare **vPeriod** come nome e impostare l'oggetto in modo che venga visualizzato come **Elenco a discesa**.
5. Sotto **Valori**, fare clic sui valori **Dinamici**. Inserire quanto segue:
   `='1~month|2~bi-month|3~quarter|4~tertial|6~half-year'.`

Creare una nuova tabella con tali campi come dimensioni:

- `employee_id`
- `employee_name`

Per calcolare gli interessi accumulati, creare questa misura:

```
=floor(monthsend($(vPeriod),today(1))-today(1))*avg_daily_claim
```

Questa misura è dinamica e produrrà risultati diversi a seconda della data di caricamento dei dati.

Impostare la misura **Formattazione numero** su **Denaro**.

Tabella dei risultati

employee_id	employee_ name	=floor(monthsend($(vPeriod),today(1))-today(1))*avg_daily_ claim
182	Contrassegno	$1410.00
183	Deryck	$1175.00
184	Dexter	$1175.00
185	Sydney	$2538.00
186	Agatha	$1692.00

La funzione `monthsend()` utilizza l'input dell'utente come primo argomento e la data odierna come secondo argomento. Restituisce la data di fine del periodo di tempo selezionato dall'utente. Quindi, l'espressione restituisce il numero di giorni che rimangono nel periodo di tempo selezionato, sottraendo la data odierna da questa data finale.

Questo valore viene quindi moltiplicato per la media delle richieste di rimborso spese giornaliere di ciascun dipendente per calcolare il valore stimato delle richieste che ogni dipendente dovrebbe presentare durante i giorni rimanenti di questo periodo.

monthsname

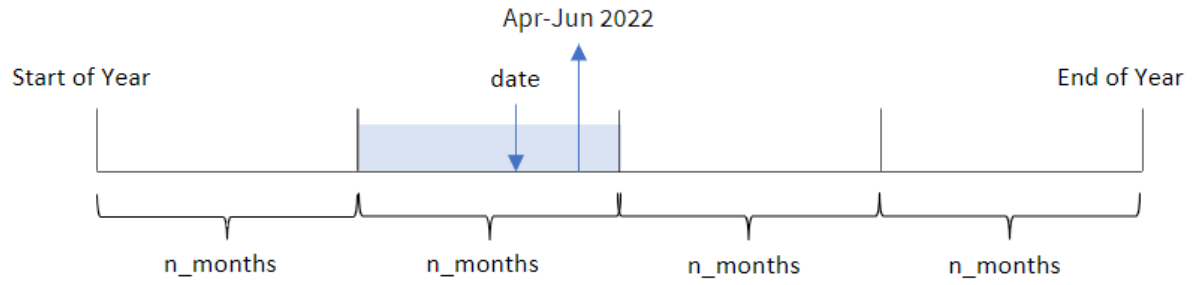
Questa funzione restituisce un valore di visualizzazione che rappresenta l'intervallo dei mesi del periodo (formattati in base alla variabile di script **MonthNames**) e l'anno. Il valore numerico sottostante corrisponde a un indicatore temporale recante il primo millisecondo del mese, del bimestre, del trimestre, del quadrimestre o del semestre contenente una data di base.

Sintassi:

```
MonthsName(n_months, date[, period_no[, first_month_of_year]])
```

Tipo di dati restituiti: duale

Schema della funzione monthname



La funzione `monthsname()` divide l'anno in segmenti in base all'argomento `n_months` fornito. Quindi, valuta il segmento a cui appartiene ogni valore `date` fornito e restituisce i nomi del mese di inizio e di fine di quel segmento, nonché l'anno. La funzione offre anche la possibilità di restituire questi limiti dai segmenti precedenti o successivi, oltre a ridefinire quale è il primo mese dell'anno.

I seguenti segmenti dell'anno sono disponibili nella funzione come argomenti di `n_month` :

Possibili argomenti di `n_month`

Periodi	Numero di mesi
mese	1
bimestre	2
trimestre	3
quattro mesi	4
semestre	6

Argomenti

Argomento	Descrizione
n_months	Il numero di mesi che definisce il periodo. Un numero intero o un'espressione la cui risoluzione è un numero intero corrispondente a: 1 (equivalente alla funzione inmonth ()), 2 (bimestre), 3 (equivalente alla funzione inquarter()), 4 (quadrimestre) o 6 (semestre).
date	La data o la data e ora da valutare.
period_no	Il periodo può essere differito mediante **period_no**, un numero intero, o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica il periodo che contiene **base_date**. I valori negativi di **period_no** indicano i periodi precedenti, mentre i valori positivi indicano i periodi successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

Casi di utilizzo

La funzione `monthsname()` è utile quando si desidera fornire all'utente la funzionalità per confrontare le aggregazioni in base a un periodo di sua scelta. Ad esempio, è possibile fornire una variabile di input per consentire all'utente di visualizzare le vendite totali dei prodotti per mese, trimestre o semestre.

Queste dimensioni possono essere create nello script di caricamento aggiungendo la funzione come campo in una tabella Calendario principale o, in alternativa, creando la dimensione direttamente in un grafico come dimensione calcolata.

Esempi di funzioni

Esempio	Risultato
`monthsname(4, '10/19/2013')`	Restituisce 'Sep-Dec 2013'. In questo e in altri esempi, l'istruzione **SET Monthnames** è impostata su Jan;Feb;Mar e così via.
`monthsname(4, '10/19/2013', -1)`	Restituisce 'May-Aug 2013'.
`monthsname(4, '10/19/2013', 0, 2)`	Restituisce 'Oct-Jan 2014', poiché come mese di inizio anno è stato specificato 2. Quindi, il periodo di quattro mesi termina il primo mese dell'anno successivo.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `bi_monthly_range`, che raggruppa le transazioni in segmenti bimestrali e restituisce i nomi limite di quel segmento per ogni transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
```

```
        *,
        monthsname(2,date) as bi_monthly_range
        ;
Load
*
Inline
[
id,date,amount
8188,2/19/2022,37.23
8189,3/7/2022,17.17
8190,3/30/2022,88.27
8191,4/5/2022,57.42
8192,4/16/2022,53.80
8193,5/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `bi_monthly_range`

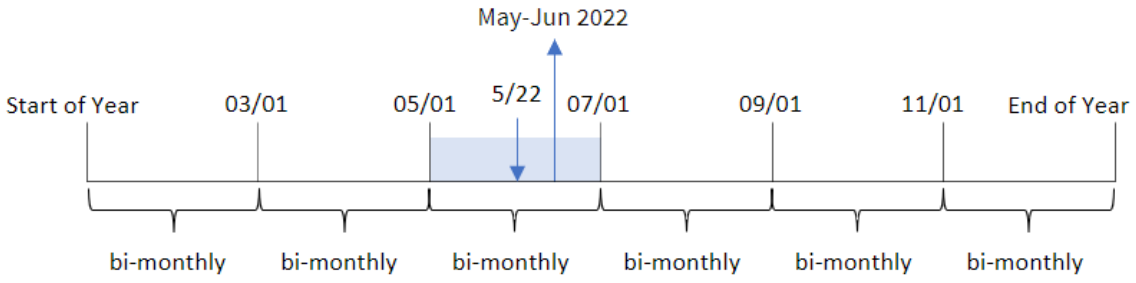
Tabella dei risultati

data	bi_monthly_range
2/19/2022	Gen-Feb 2022
3/7/2022	Mar-Apr 2022
3/30/2022	Mar-Apr 2022
4/5/2022	Mar-Apr 2022
4/16/2022	Mar-Apr 2022
5/1/2022	Mag-Giu 2022
5/7/2022	Mag-Giu 2022

data	bi_monthly_range
5/22/2022	Mag-Giu 2022
6/15/2022	Mag-Giu 2022
6/26/2022	Mag-Giu 2022
7/9/2022	Lug-Ago 2022
7/22/2022	Lug-Ago 2022
7/23/2022	Lug-Ago 2022
7/27/2022	Lug-Ago 2022
8/2/2022	Lug-Ago 2022
8/8/2022	Lug-Ago 2022
8/19/2022	Lug-Ago 2022
9/26/2022	Set-Ott 2022
10/14/2022	Set-Ott 2022
10/29/2022	Set-Ott 2022

Il campo `bi_monthly_range` viene creato nell'istruzione LOAD precedente utilizzando la funzione `monthsname()`. Il primo argomento fornito è 2, dividendo l'anno in segmenti bimestrali. Il primo argomento identifica il campo da valutare.

Schema della funzione monthsname, esempio di base



La transazione 8195 è avvenuta il 22 maggio. La funzione `monthsname()` divide inizialmente l'anno in segmenti bimestrali. La transazione 8195 rientra nel segmento tra il 1 maggio e il 30 giugno. Pertanto, la funzione restituisce questi mesi nel formato della variabile di sistema `MonthNames`, nonché l'anno, maggio-giugno 2022.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati inline e lo stesso scenario del primo esempio.
- La creazione di un campo, `prev_bi_monthly_range`, che raggruppa le transazioni in segmenti bimestrali e restituisce i nomi limite del segmento precedente per ogni transazione.

Aggiungere qui altro testo, se necessario, con elenchi, ecc.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        MonthsName(2,date,-1) as prev_bi_monthly_range
        ;
Load
*
Inline
[
id,date,amount
8188,2/19/2022,37.23
8189,3/7/2022,17.17
8190,3/30/2022,88.27
8191,4/5/2022,57.42
8192,4/16/2022,53.80
8193,5/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

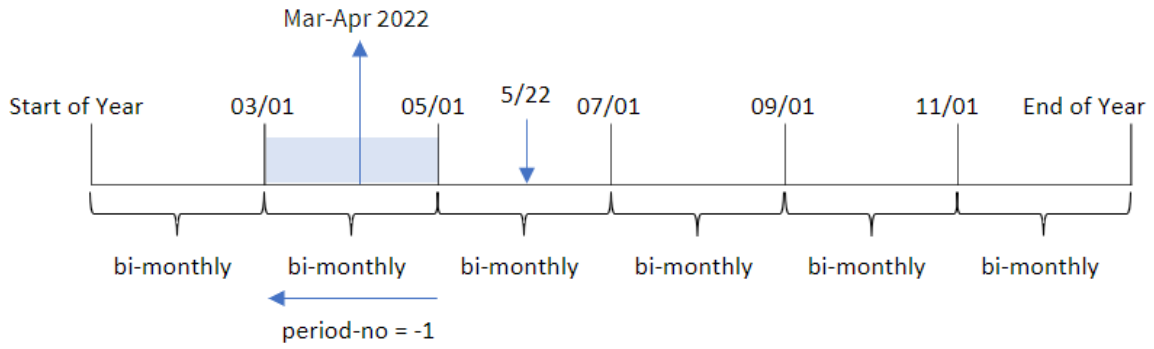
- `date`
- `prev_bi_monthly_range`

Tabella dei risultati

data	prev_bi_monthly_range
2/19/2022	Nov-Dic 2021
3/7/2022	Gen-Feb 2022
3/30/2022	Gen-Feb 2022
4/5/2022	Gen-Feb 2022
4/16/2022	Gen-Feb 2022
5/1/2022	Mar-Apr 2022
5/7/2022	Mar-Apr 2022
5/22/2022	Mar-Apr 2022
6/15/2022	Mar-Apr 2022
6/26/2022	Mar-Apr 2022
7/9/2022	Mag-Giu 2022
7/22/2022	Mag-Giu 2022
7/23/2022	Mag-Giu 2022
7/27/2022	Mag-Giu 2022
8/2/2022	Mag-Giu 2022
8/8/2022	Mag-Giu 2022
8/19/2022	Mag-Giu 2022
9/26/2022	Lug-Ago 2022
10/14/2022	Lug-Ago 2022
10/29/2022	Lug-Ago 2022

In questo esempio, -1 viene utilizzato come argomento `period_no` nella funzione `monthsname()`. Dopo aver inizialmente diviso un anno in segmenti bimestrali, la funzione restituisce quindi i limiti del segmento precedente per il momento in cui avviene una transazione.

Schema della funzione monthsname, esempio di period_no



La transazione 8195 è stata effettuata nel segmento tra maggio e giugno. Pertanto, il segmento bimestrale precedente è compreso tra il 1 marzo e il 30 aprile, quindi la funzione restituisce Mar-Apr 2022.

Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati inline e lo stesso scenario del primo esempio.
- La creazione di un campo differente, `bi_monthly_range`, che raggruppa le transazioni in segmenti bimestrali e restituisce i limiti del segmento per ogni transazione.

Tuttavia, in questo esempio, dobbiamo anche impostare aprile come primo mese dell'anno finanziario.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        MonthsName(2,date,0,4) as bi_monthly_range
        ;
Load
*
Inline
[
id,date,amount
8188,2/19/2022,37.23
8189,3/7/2022,17.17
8190,3/30/2022,88.27
8191,4/5/2022,57.42
```

```
8192,4/16/2022,53.80
8193,5/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `bi_monthly_range`

Tabella dei risultati

data	bi_monthly_range
2/19/2022	Feb-Mar 2021
3/7/2022	Feb-Mar 2021
3/30/2022	Feb-Mar 2021
4/5/2022	Apr-Mag 2022
4/16/2022	Apr-Mag 2022
5/1/2022	Apr-Mag 2022
5/7/2022	Apr-Mag 2022
5/22/2022	Apr-Mag 2022
6/15/2022	Giu-Lug 2022
6/26/2022	Giu-Lug 2022
7/9/2022	Giu-Lug 2022
7/22/2022	Giu-Lug 2022
7/23/2022	Giu-Lug 2022
7/27/2022	Giu-Lug 2022

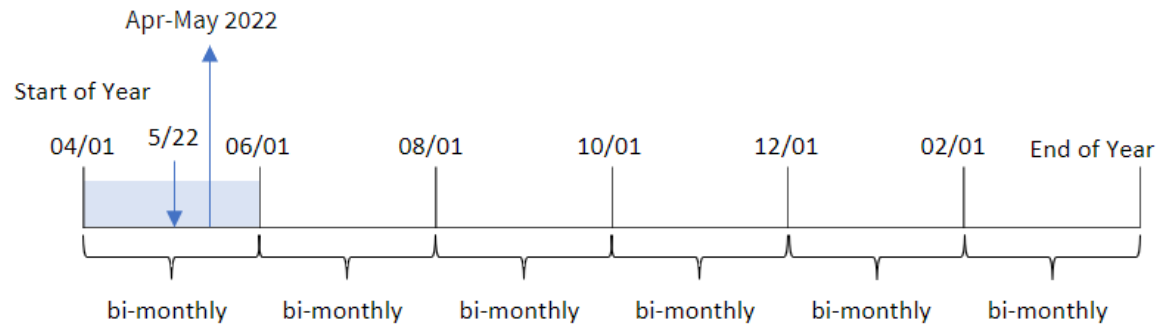
data	bi_monthly_range
8/2/2022	Ago-Set 2022
8/8/2022	Ago-Set 2022
8/19/2022	Ago-Set 2022
9/26/2022	Ago-Set 2022
10/14/2022	Ott-Nov 2022
10/29/2022	Ott-Nov 2022

Utilizzando 4 come argomento `first_month_of_year` nella funzione `monthsname()`, la funzione inizia l'anno il 1 aprile, quindi divide l'anno in segmenti bimestrali: Apr-May,Jun-Jul,Aug-Sep,Oct-Nov,Dec-Jan,Feb-Mar.

Testo del paragrafo per i Risultati.

La transazione 8195 è avvenuta il 22 maggio e rientra nel segmento tra il 1 aprile e il 31 maggio. Pertanto, la funzione restituisce Apr-Mag 2022.

Diagramma della funzione monthsname, esempio di first_month_of_year



Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati inline e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che raggruppa le transazioni in segmenti bimestrali e restituisce i limiti del segmento per ogni transazione viene creato come una misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,2/19/2022,37.23
8189,3/7/2022,17.17
8190,3/30/2022,88.27
8191,4/5/2022,57.42
8192,4/16/2022,53.80
8193,5/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`date`.

Creare la seguente misura:

```
=monthsname(2,date)
```

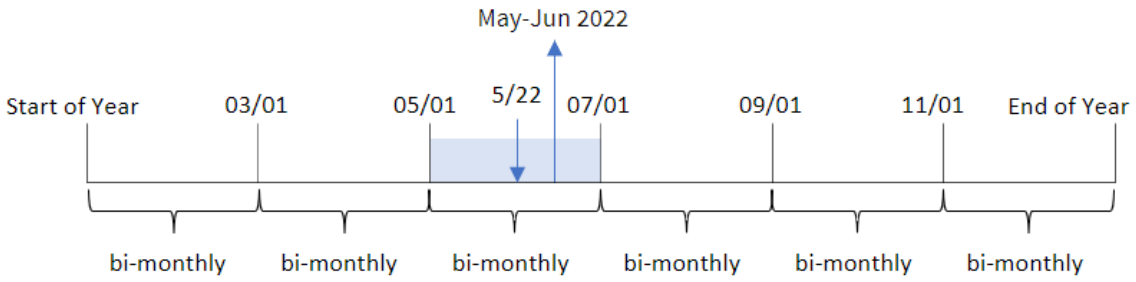
Tabella dei risultati

data	=monthsname(2,date)
2/19/2022	Gen-Feb 2022
3/7/2022	Mar-Apr 2022
3/30/2022	Mar-Apr 2022
4/5/2022	Mar-Apr 2022
4/16/2022	Mar-Apr 2022

data	=monthsname(2,date)
5/1/2022	Mag-Giu 2022
5/7/2022	Mag-Giu 2022
5/22/2022	Mag-Giu 2022
6/15/2022	Mag-Giu 2022
6/26/2022	Mag-Giu 2022
7/9/2022	Lug-Ago 2022
7/22/2022	Lug-Ago 2022
7/23/2022	Lug-Ago 2022
7/27/2022	Lug-Ago 2022
8/2/2022	Lug-Ago 2022
8/8/2022	Lug-Ago 2022
8/19/2022	Lug-Ago 2022
9/26/2022	Set-Ott 2022
10/14/2022	Set-Ott 2022
10/29/2022	Set-Ott 2022

Il campo `bi_monthly_range` viene creato come misura nell'oggetto grafico mediante l'utilizzo della funzione `monthsname()`. Il primo argomento fornito è 2, dividendo l'anno in segmenti bimestrali. Il primo argomento identifica il campo da valutare.

Schema della funzione monthsname, esempio di oggetto grafico



La transazione 8195 è avvenuta il 22 maggio. La funzione `monthsname()` divide inizialmente l'anno in segmenti bimestrali. La transazione 8195 rientra nel segmento tra il 1 maggio e il 30 giugno. Pertanto, la funzione restituisce questi mesi nel formato della variabile di sistema , nonché l'anno, maggio-giugno 2022.MonthNames

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente le transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

L'utente finale vuole creare un oggetto grafico che visualizzi le vendite totali per un periodo selezionato. Questo può essere ottenuto anche quando la dimensione non è disponibile nel modello dati, utilizzando la funzione `monthsname()` come dimensione calcolata che viene modificata dinamicamente da un controllo di input variabile.

Script di caricamento

```
SET vPeriod = 1;
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'1/7/2022',17.17
8189,'1/19/2022',37.23
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio.

All'inizio dello script di caricamento, viene creata una variabile (`vPeriod`) che sarà collegata al controllo di input della variabile. Quindi, configurare la variabile come oggetto personalizzato nel foglio.

Procedere come indicato di seguito:

1. Nel pannello delle risorse, fare clic **su Oggetti personalizzati**.
2. Selezionare **Includi dashboard Qlik** e creare un oggetto **Input variabile**.
3. Immettere un titolo per l'oggetto grafico.
4. In **Variabile**, selezionare **vPeriod** come Nome e impostare l'oggetto in modo che venga visualizzato come **elenco a discesa**.
5. In **Valori**, configurare l'oggetto in modo che utilizzi valori dinamici. Inserire quanto segue:
   `='1~month|2~bi-month|3~quarter|4~tertial|6~half-year'`

Quindi, creare la tabella dei risultati.

Procedere come indicato di seguito:

1. Creare una nuova tabella e aggiungere la seguente dimensione calcolata:
   `=monthsname($(vPeriod),date)`
2. Aggiungere questa misura per calcolare le vendite totali:
   `=sum(amount)`
3. Impostare la **Formattazione numero** della misura su **Denaro**. Fare clic su ✔ **Termina modifica**. È ora possibile modificare i dati mostrati nella tabella regolando il segmento temporale nell'oggetto della variabile.

Ecco come apparirà la tabella dei risultati quando l'opzione `tertial` è selezionata:

Tabella dei risultati

monthsname($(vPeriod),date)	sum(amount)
Gen-Apr 2022	253.89
Mag-ago 2022	713.58
Set-Dic 2022	248.12

monthsstart

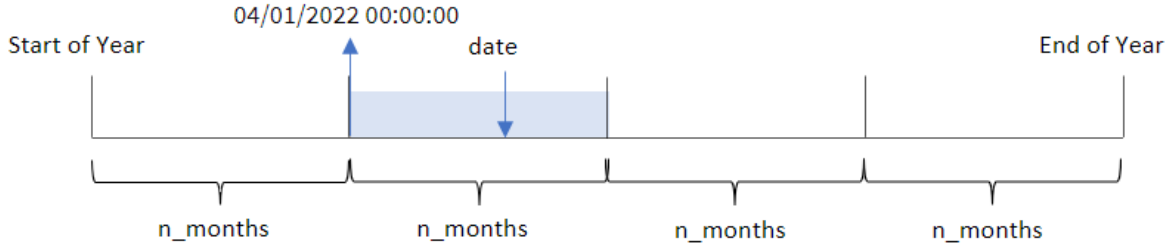
Questa funzione restituisce un valore corrispondente a un indicatore temporale recante il primo millisecondo del mese, del bimestre, del trimestre, del quadrimestre o del semestre contenente una data di base. È inoltre possibile individuare l'indicatore temporale per un periodo di tempo precedente o successivo.Il formato di output predefinito è il formato **DateFormat** impostato nello script.

Sintassi:

```
MonthsStart(n_months, date[, period_no [, first_month_of_year]])
```

Tipo di dati restituiti: duale

Schema della funzione `monthsstart()`.



La funzione `monthsstart()` divide l'anno in segmenti in base all'argomento `n_months` fornito. Quindi valuta in quale segmento rientra ogni data fornita e restituisce il primo millisecondo, in formato data, di quel segmento. La funzione offre anche la possibilità di restituire il timestamp iniziale dai segmenti precedenti o successivi, oltre a ridefinire qual è il primo mese dell'anno.

I seguenti segmenti dell'anno sono disponibili nella funzione come argomenti di `n_month` :

Possibili argomenti di `n_month`

Periodi	Numero di mesi
mese	1
bimestre	2
trimestre	3
quattro mesi	4
semestre	6

Argomenti

Argomento	Descrizione
n_months	Il numero di mesi che definisce il periodo. Un numero intero o un'espressione la cui risoluzione è un numero intero corrispondente a: 1 (equivalente alla funzione inmonth ()), 2 (bimestre), 3 (equivalente alla funzione inquarter()), 4 (quadrimestre) o 6 (semestre).
date	La data o la data e ora da valutare.
period_no	Il periodo può essere differito mediante **period_no**, un numero intero, o un'espressione la cui risoluzione è un numero intero, in cui il valore 0 indica il periodo che contiene **base_date**. I valori negativi di **period_no** indicano i periodi precedenti, mentre i valori positivi indicano i periodi successivi.

Argomento	Descrizione
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

Casi di utilizzo

La funzione `monthsstart()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione di un periodo non ancora trascorso. Questo potrebbe essere utilizzato, ad esempio, per fornire una variabile di input che consenta all'utente di calcolare il totale degli interessi accumulati fino a quel momento nel mese, nel trimestre o nel semestre.

Esempi di funzioni

Esempio	Risultato
`monthsstart(4, '10/19/2013')`	Restituisce 09/01/2013.
`monthsstart(4, '10/19/2013, -1)`	Restituisce 05/01/2013.
`monthsstart(4, '10/19/2013', 0, 2 )`	Restituisce 10/01/2013, poiché l'inizio dell'anno corrisponde al mese 2.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.

- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `bi_monthly_start`, che raggruppa le transazioni in segmenti bimestrali e restituisce il timestamp iniziale del segmento per ogni transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        monthsstart(2,date) as bi_monthly_start,
        timestamp(monthsstart(2,date)) as bi_monthly_start_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,2/19/2022,37.23
8189,3/7/2022,17.17
8190,3/30/2022,88.27
8191,4/5/2022,57.42
8192,4/16/2022,53.80
8193,5/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

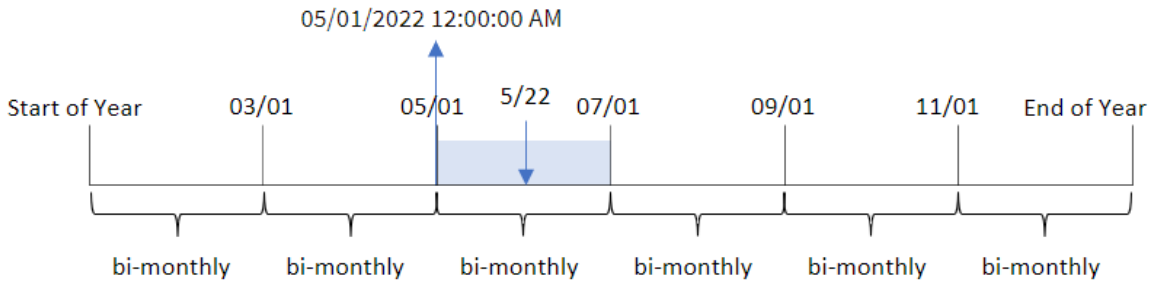
- `date`
- `bi_monthly_start`
- `bi_monthly_start_timestamp`

Tabella dei risultati

date	bi_monthly_start	bi_monthly_start_timestamp
2/19/2022	01/01/2022	1/1/2022 12:00:00 AM
3/7/2022	03/01/2022	3/1/2022 12:00:00 AM
3/30/2022	03/01/2022	3/1/2022 12:00:00 AM
4/5/2022	03/01/2022	3/1/2022 12:00:00 AM
4/16/2022	03/01/2022	3/1/2022 12:00:00 AM
5/1/2022	05/01/2022	5/1/2022 12:00:00 AM
5/7/2022	05/01/2022	5/1/2022 12:00:00 AM
5/22/2022	05/01/2022	5/1/2022 12:00:00 AM
6/15/2022	05/01/2022	5/1/2022 12:00:00 AM
6/26/2022	05/01/2022	5/1/2022 12:00:00 AM
7/9/2022	07/01/2022	7/1/2022 12:00:00 AM
7/22/2022	07/01/2022	7/1/2022 12:00:00 AM
7/23/2022	07/01/2022	7/1/2022 12:00:00 AM
7/27/2022	07/01/2022	7/1/2022 12:00:00 AM
8/2/2022	07/01/2022	7/1/2022 12:00:00 AM
8/8/2022	07/01/2022	7/1/2022 12:00:00 AM
8/19/2022	07/01/2022	7/1/2022 12:00:00 AM
9/26/2022	09/01/2022	9/1/2022 12:00:00 AM
10/14/2022	09/01/2022	9/1/2022 12:00:00 AM
10/29/2022	09/01/2022	9/1/2022 12:00:00 AM

Il campo `bi_monthly_start` viene creato nell'istruzione LOAD precedente utilizzando la funzione `monthsstart()`. Il primo argomento fornito è 2, dividendo l'anno in segmenti bimestrali. Il primo argomento identifica il campo da valutare.

Schema della funzione `monthsstart()`, esempio senza argomenti aggiuntivi



La transazione 8195 avviene il 22 maggio. La funzione `monthsstart()` divide inizialmente l'anno in segmenti bimestrali. La transazione 8195 rientra nel segmento tra il 1 maggio e il 30 giugno. Pertanto, la funzione restituisce il primo millisecondo di questo segmento, il 1° maggio 2022 alle ore 12:00:00 AM.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, '`prev_bi_monthly_start`', che restituisce il primo millisecondo del segmento bimestrale prima che avvenga la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        monthsstart(2,date,-1) as prev_bi_monthly_start,
           timestamp(monthsstart(2,date,-1)) as prev_bi_monthly_start_timestamp
           ;
Load
*
Inline
[
id,date,amount
8188,2/19/2022,37.23
8189,3/7/2022,17.17
8190,3/30/2022,88.27
8191,4/5/2022,57.42
8192,4/16/2022,53.80
8193,5/1/2022,82.06
```

```
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `prev_bi_monthly_start`
- `prev_bi_monthly_start_timestamp`

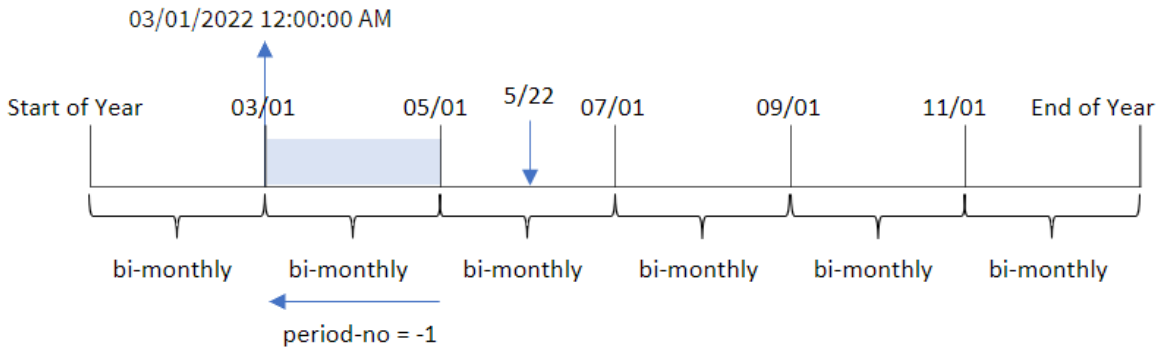
Tabella dei risultati

date	prev_bi_monthly_start	prev_bi_monthly_start_timestamp
2/19/2022	11/01/2021	1/11/2021 12:00:00 AM
3/7/2022	01/01/2022	1/1/2022 12:00:00 AM
3/30/2022	01/01/2022	1/1/2022 12:00:00 AM
4/5/2022	01/01/2022	1/1/2022 12:00:00 AM
4/16/2022	01/01/2022	1/1/2022 12:00:00 AM
5/1/2022	03/01/2022	3/1/2022 12:00:00 AM
5/7/2022	03/01/2022	3/1/2022 12:00:00 AM
5/22/2022	03/01/2022	3/1/2022 12:00:00 AM
6/15/2022	03/01/2022	3/1/2022 12:00:00 AM
6/26/2022	03/01/2022	3/1/2022 12:00:00 AM
7/9/2022	05/01/2022	5/1/2022 12:00:00 AM
7/22/2022	05/01/2022	5/1/2022 12:00:00 AM
7/23/2022	05/01/2022	5/1/2022 12:00:00 AM
7/27/2022	05/01/2022	5/1/2022 12:00:00 AM
8/2/2022	05/01/2022	5/1/2022 12:00:00 AM

date	prev_bi_monthly_start	prev_bi_monthly_start_timestamp
8/8/2022	05/01/2022	5/1/2022 12:00:00 AM
8/19/2022	05/01/2022	5/1/2022 12:00:00 AM
9/26/2022	07/01/2022	7/1/2022 12:00:00 AM
10/14/2022	07/01/2022	7/1/2022 12:00:00 AM
10/29/2022	07/01/2022	7/1/2022 12:00:00 AM

Utilizzando -1 come argomento `period_no` della funzione `monthsstart()`, dopo aver inizialmente diviso un anno in segmenti bimestrali, la funzione restituisce il primo millisecondo del segmento bimestrale precedente al momento in cui avviene una transazione.

Schema della funzione `monthsstart()`, esempio di period_no



La transazione 8195 è stata effettuata nel segmento tra maggio e giugno. Pertanto, il segmento bimestrale precedente era compreso tra il 1° marzo e il 30 aprile, quindi la funzione restituisce il primo millisecondo di questo segmento, il 1° marzo 2022 alle 12:00:00.

Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `bi_monthly_start`, che raggruppa le transazioni in segmenti bimestrali e restituisce il timestamp iniziale del set per ogni transazione.

Tuttavia, in questo esempio, dobbiamo anche impostare aprile come primo mese dell'anno finanziario.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        monthsstart(2,date,0,4) as bi_monthly_start,
        timestamp(monthsstart(2,date,0,4)) as bi_monthly_start_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `bi_monthly_start`
- `bi_monthly_start_timestamp`

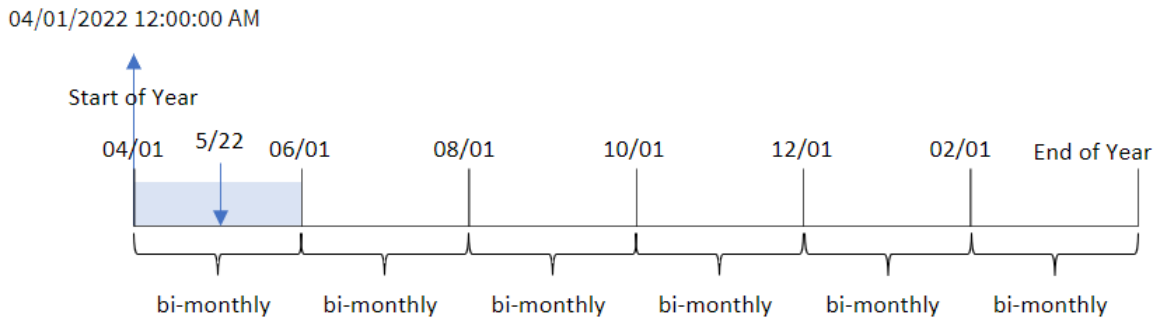
Tabella dei risultati

date	bi_monthly_start	bi_monthly_start_timestamp
2/19/2022	02/01/2022	2/1/2022 12:00:00 AM
3/7/2022	02/01/2022	2/1/2022 12:00:00 AM

date	bi_monthly_start	bi_monthly_start_timestamp
3/30/2022	02/01/2022	2/1/2022 12:00:00 AM
4/5/2022	04/01/2022	4/1/2022 12:00:00 AM
4/16/2022	04/01/2022	4/1/2022 12:00:00 AM
5/1/2022	04/01/2022	4/1/2022 12:00:00 AM
5/7/2022	04/01/2022	4/1/2022 12:00:00 AM
5/22/2022	04/01/2022	4/1/2022 12:00:00 AM
6/15/2022	06/01/2022	6/1/2022 12:00:00 AM
6/26/2022	06/01/2022	6/1/2022 12:00:00 AM
7/9/2022	06/01/2022	6/1/2022 12:00:00 AM
7/22/2022	06/01/2022	6/1/2022 12:00:00 AM
7/23/2022	06/01/2022	6/1/2022 12:00:00 AM
7/27/2022	06/01/2022	6/1/2022 12:00:00 AM
8/2/2022	08/01/2022	8/1/2022 12:00:00 AM
8/8/2022	08/01/2022	8/1/2022 12:00:00 AM
8/19/2022	08/01/2022	8/1/2022 12:00:00 AM
9/26/2022	08/01/2022	8/1/2022 12:00:00 AM
10/14/2022	10/01/2022	10/1/2022 12:00:00 AM
10/29/2022	10/01/2022	10/1/2022 12:00:00 AM

Utilizzando 4 come argomento `first_month_of_year` nella funzione `monthsstart()`, la funzione inizia l'anno il 1 aprile, quindi divide l'anno in segmenti bimestrali: Apr-May,Jun-Jul,Aug-Sep,Oct-Nov,Dec-Jan,Feb-Mar.

Schema della funzione `monthsstart()`, esempio di first_month_of_year



La transazione 8195 è avvenuta il 22 maggio e rientra nel segmento tra il 1 aprile e il 31 maggio. Pertanto, la funzione restituisce il primo millisecondo di questo segmento, il 1° aprile 2022 alle ore 12:00:00 AM.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che raggruppa le transazioni in segmenti bimestrali e restituisce il timestamp iniziale del set per ogni transazione viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,2/19/2022,37.23
8189,3/7/2022,17.17
8190,3/30/2022,88.27
8191,4/5/2022,57.42
8192,4/16/2022,53.80
8193,5/1/2022,82.06
8194,5/7/2022,40.39
8195,5/22/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Creare le seguenti misure:

```
=monthsstart(2,date)
```

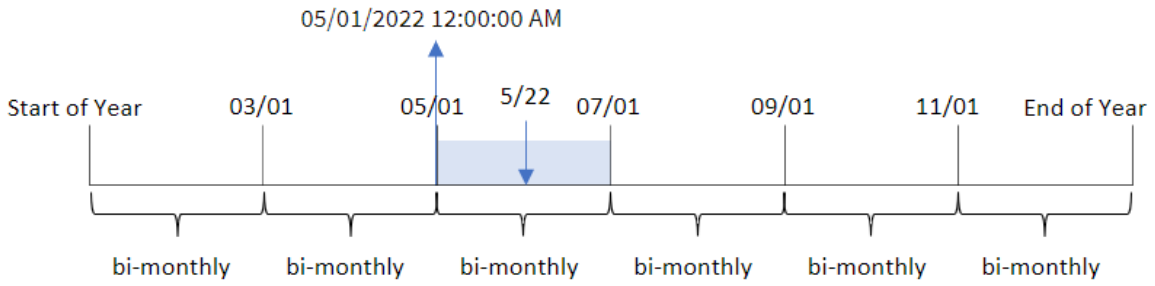
```
=timestamp(monthsstart(2,date))
```

Questi calcoli recuperano il timestamp iniziale del segmento bimestrale in cui ha avuto luogo ciascuna transazione.

Tabella dei risultati

date	=monthsstart(2,date)	=timestamp(monthsstart(2,date))
9/26/2022	09/01/2022	9/1/2022 12:00:00 AM
10/14/2022	09/01/2022	9/1/2022 12:00:00 AM
10/29/2022	09/01/2022	9/1/2022 12:00:00 AM
7/9/2022	07/01/2022	7/1/2022 12:00:00 AM
7/22/2022	07/01/2022	7/1/2022 12:00:00 AM
7/23/2022	07/01/2022	7/1/2022 12:00:00 AM
7/27/2022	07/01/2022	7/1/2022 12:00:00 AM
8/2/2022	07/01/2022	7/1/2022 12:00:00 AM
8/8/2022	07/01/2022	7/1/2022 12:00:00 AM
8/19/2022	07/01/2022	7/1/2022 12:00:00 AM
5/1/2022	05/01/2022	5/1/2022 12:00:00 AM
5/7/2022	05/01/2022	5/1/2022 12:00:00 AM
5/22/2022	05/01/2022	5/1/2022 12:00:00 AM
6/15/2022	05/01/2022	5/1/2022 12:00:00 AM
6/26/2022	05/01/2022	5/1/2022 12:00:00 AM
3/7/2022	03/01/2022	3/1/2022 12:00:00 AM
3/30/2022	03/01/2022	3/1/2022 12:00:00 AM
4/5/2022	03/01/2022	3/1/2022 12:00:00 AM
4/16/2022	03/01/2022	3/1/2022 12:00:00 AM
2/19/2022	01/01/2022	1/1/2021 12:00:00 AM

Schema della funzione `monthsstart()`, esempio di oggetto grafico



La transazione 8195 è avvenuta il 22 maggio. La funzione `monthsstart()` divide inizialmente l'anno in segmenti bimestrali. La transazione 8195 rientra nel segmento tra il 1 maggio e il 30 giugno. Pertanto, la funzione restituisce il primo millisecondo di questo segmento, in data 01/05/2022 alle ore 12:00:00 AM.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di saldi di prestiti, che viene caricato in una tabella chiamata `Loans`.
- Dati costituiti dagli ID dei prestiti, dal saldo all'inizio del mese e dal tasso di interesse semplice annuo applicato a ciascun prestito.

L'utente finale desidera un oggetto grafico che visualizzi, in base all'ID del prestito, gli interessi correnti maturati su ciascun prestito per un periodo a scelta. L'esercizio finanziario inizia a gennaio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Loans:
Load
*
Inline
[
loan_id,start_balance,rate
8188,$10000.00,0.024
8189,$15000.00,0.057
8190,$17500.00,0.024
8191,$21000.00,0.034
8192,$90000.00,0.084
];
```

Risultati

Caricare i dati e aprire un foglio.

All'inizio dello script di caricamento, viene creata una variabile (`vPeriod`) che sarà collegata al controllo di input della variabile. Quindi, configurare la variabile come oggetto personalizzato nel foglio.

Procedere come indicato di seguito:

1. Nel pannello delle risorse, fare clic **su Oggetti personalizzati**.
2. Selezionare **Includi dashboard Qlik** e creare un oggetto **Input variabile**.
3. Immettere un titolo per l'oggetto grafico.
4. In **Variabile**, selezionare **vPeriod** come Nome e impostare l'oggetto in modo che venga visualizzato come **elenco a discesa**.
5. In **Valori**, configurare l'oggetto in modo che utilizzi valori dinamici. Inserire quanto segue:
   `='1~month|2~bi-month|3~quarter|4~tertial|6~half-year'`

Quindi, creare la tabella dei risultati.

Procedere come indicato di seguito:

1. Creare una nuova tabella. Aggiungere i seguenti campi come dimensioni:
   - `employee_id`
   - `employee_name`
2. Creare una misura per calcolare gli interessi accumulati:
   `=start_balance*(rate*(today(1)-monthsstart($(vPeriod),today(1)))/365)`
3. Impostare la **Formattazione numero** della misura su **Denaro**. Fare clic su ✔ **Termina modifica**. È ora possibile modificare i dati mostrati nella tabella regolando il segmento temporale nell'oggetto della variabile.

Ecco come apparirà la tabella dei risultati quando sarà selezionata l'opzione del periodo `month`:

Tabella dei risultati

loan_id	start_balance	=start_balance*(rate*(today(1)-monthsstart($(vPeriod),today(1)))/365)
8188	$10000.00	$7.95
8189	$15000.00	$67.93
8190	$17500.00	$33.37
8191	$21000.00	$56.73
8192	$90000.00	$600.66

La funzione `monthsstart()`, utilizzando l'input dell'utente come primo argomento e la data odierna come secondo argomento, restituisce la data di inizio del periodo scelto dall'utente. Sottraendo tale risultato dalla data corrente, l'espressione restituisce il numero di giorni trascorsi fino a ora in questo periodo.

Questo valore viene quindi moltiplicato per il tasso di interesse e diviso per 365, per restituire il tasso di interesse effettivo incorso per questo periodo. Il risultato viene poi moltiplicato per il saldo iniziale del prestito per ottenere gli interessi maturati fino a ora in questo periodo.

monthstart

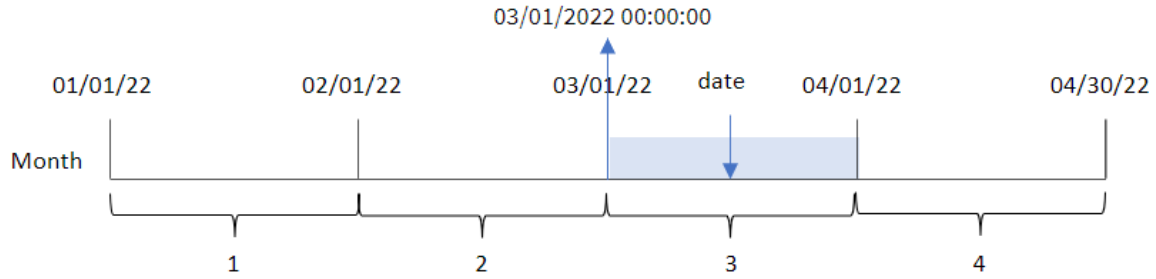
Questa funzione restituisce un valore corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno del mese contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

Sintassi:

```
MonthStart(date[, period_no])
```

Tipo di dati restituiti: duale

Schema della funzione `monthstart()`.



La funzione `monthstart()` determina in quale mese cade la data. Restituisce quindi un timestamp, in formato data, per il primo millisecondo di quel mese.

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero che, se corrisponde a 0 o viene omesso, indica il mese contenente **date**. I valori negativi di **period_no** indicano i mesi precedenti, mentre i valori positivi indicano i mesi successivi.

Casi di utilizzo

La funzione `monthstart()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione del mese trascorsa finora. Ad esempio, può essere utilizzato per calcolare gli interessi accumulati in un mese fino a una certa data.

Esempi di funzioni

Esempio	Risultato
`monthstart('10/19/2001')`	Restituisce 10/01/2001.
`monthstart('10/19/2001', -1)`	Restituisce 09/01/2001.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `start_of_month`, che restituisce un timestamp per l'inizio del mese in cui sono avvenute le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        monthstart(date) as start_of_month,
        timestamp(monthstart(date)) as start_of_month_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
```

```
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `start_of_month`
- `start_of_month_timestamp`

Tabella dei risultati

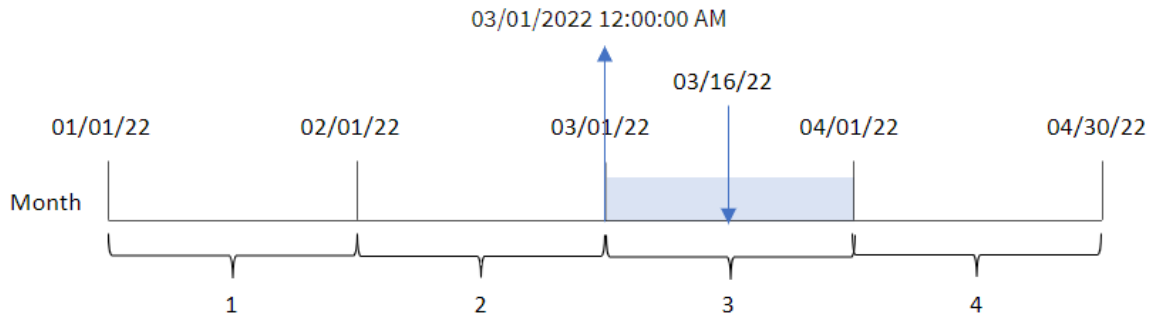
date	start_of_month	start_of_month_timestamp
1/7/2022	01/01/2022	1/1/2022 12:00:00 AM
1/19/2022	01/01/2022	1/1/2022 12:00:00 AM
2/5/2022	02/01/2022	2/1/2022 12:00:00 AM
2/28/2022	02/01/2022	2/1/2022 12:00:00 AM
3/16/2022	03/01/2022	3/1/2022 12:00:00 AM
4/1/2022	04/01/2022	4/1/2022 12:00:00 AM
5/7/2022	05/01/2022	5/1/2022 12:00:00 AM
5/16/2022	05/01/2022	5/1/2022 12:00:00 AM
6/15/2022	06/01/2022	6/1/2022 12:00:00 AM
6/26/2022	07/01/2022	6/1/2022 12:00:00 AM
7/9/2022	07/01/2022	7/1/2022 12:00:00 AM
7/22/2022	07/01/2022	7/1/2022 12:00:00 AM
7/23/2022	07/01/2022	7/1/2022 12:00:00 AM
7/27/2022	07/01/2022	7/1/2022 12:00:00 AM

date	start_of_month	start_of_month_timestamp
8/2/2022	08/01/2022	8/1/2022 12:00:00 AM
8/8/2022	08/01/2022	8/1/2022 12:00:00 AM
8/19/2022	08/01/2022	8/1/2022 12:00:00 AM
9/26/2022	09/01/2022	9/1/2022 12:00:00 AM
10/14/2022	10/01/2022	10/1/2022 12:00:00 AM
10/29/2022	10/01/2022	10/1/2022 12:00:00 AM

Il campo `start_of_month` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `monthstart()` e trasferendo il campo data come argomento della funzione.

La funzione `monthstart()` identifica in quale mese cade il valore della data, restituendo un timestamp per il primo millisecondo di quel mese.

Schema della funzione `monthstart()`, esempio senza argomenti aggiuntivi



La transazione 8192 è avvenuta il 16 marzo. La funzione `monthstart()` restituisce il primo millisecondo di quel mese, ovvero il 1° marzo alle ore 12:00:00 AM.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_month_start`, che restituisce il timestamp dell'inizio del mese precedente la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        monthstart(date,-1) as previous_month_start,
        timestamp(monthstart(date,-1)) as previous_month_start_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_month_start`
- `previous_month_start_timestamp`

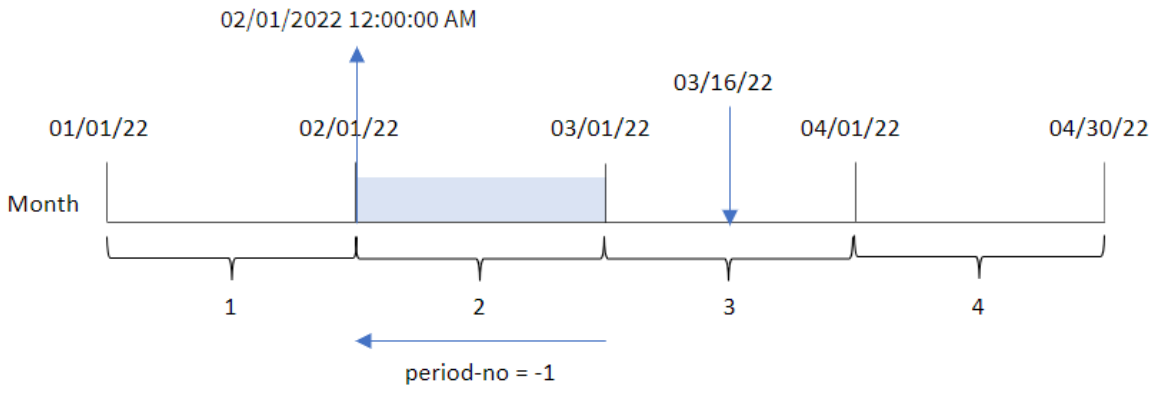
Tabella dei risultati

date	previous_month_start	previous_month_start_timestamp
1/7/2022	12/01/2021	12/1/2021 12:00:00 AM
1/19/2022	12/01/2021	12/1/2021 12:00:00 AM

date	previous_month_start	previous_month_start_timestamp
2/5/2022	01/01/2022	1/1/2022 12:00:00 AM
2/28/2022	01/01/2022	1/1/2022 12:00:00 AM
3/16/2022	02/01/2022	2/1/2022 12:00:00 AM
4/1/2022	03/01/2022	3/1/2022 12:00:00 AM
5/7/2022	04/01/2022	4/1/2022 12:00:00 AM
5/16/2022	04/01/2022	4/1/2022 12:00:00 AM
6/15/2022	05/01/2022	5/1/2022 12:00:00 AM
6/26/2022	05/01/2022	5/1/2022 12:00:00 AM
7/9/2022	06/01/2022	6/1/2022 12:00:00 AM
7/22/2022	06/01/2022	6/1/2022 12:00:00 AM
7/23/2022	06/01/2022	6/1/2022 12:00:00 AM
7/27/2022	06/01/2022	6/1/2022 12:00:00 AM
8/2/2022	07/01/2022	7/1/2022 12:00:00 AM
8/8/2022	07/01/2022	7/1/2022 12:00:00 AM
8/19/2022	07/01/2022	7/1/2022 12:00:00 AM
9/26/2022	08/01/2022	8/1/2022 12:00:00 AM
10/14/2022	09/01/2022	9/1/2022 12:00:00 AM
10/29/2022	09/01/2022	9/1/2022 12:00:00 AM

In questo caso, poiché il valore `period_no` di -1 è stato utilizzato come argomento offset nella funzione `monthstart()`, la funzione per prima cosa identifica il mese in cui avvengono le transazioni. Si sposta poi un mese prima e identifica il millisecondo iniziale di tale mese.

Schema della funzione `monthstart()`, esempio di period_no



La transazione 8192 è avvenuta il 16 marzo. La funzione `monthstart()` identifica che il mese precedente la transazione è avvenuta in febbraio. Quindi restituisce il primo millisecondo di quel mese, il 1° febbraio alle 12:00:00.

Esempio 3 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce un timestamp per l'inizio del mese in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Per calcolare la data di inizio del mese in cui avviene una transazione, creare le seguenti misure:

- `=monthstart(date)`
- `=timestamp(monthstart(date))`

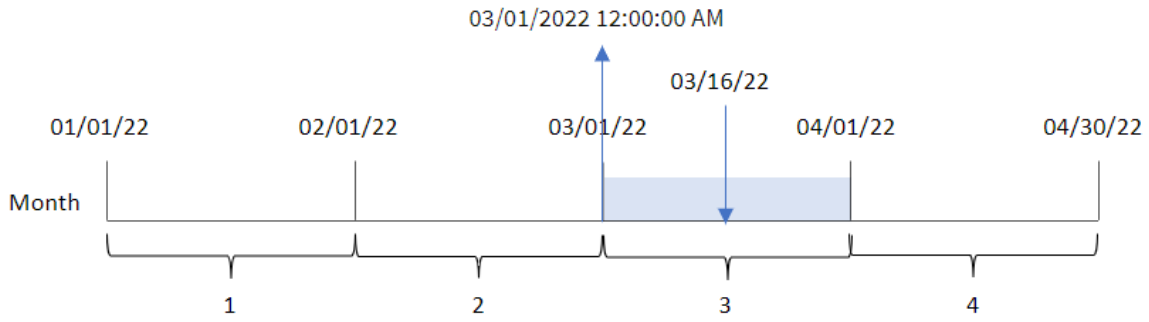
Tabella dei risultati

date	=monthstart(date)	=timestamp(monthstart(date))
10/14/2022	10/01/2022	10/1/2022 12:00:00 AM
10/29/2022	10/01/2022	10/1/2022 12:00:00 AM
9/26/2022	09/01/2022	9/1/2022 12:00:00 AM
8/2/2022	08/01/2022	8/1/2022 12:00:00 AM
8/8/2022	08/01/2022	8/1/2022 12:00:00 AM
8/19/2022	08/01/2022	8/1/2022 12:00:00 AM
7/9/2022	07/01/2022	7/1/2022 12:00:00 AM
7/22/2022	07/01/2022	7/1/2022 12:00:00 AM
7/23/2022	07/01/2022	7/1/2022 12:00:00 AM
7/27/2022	07/01/2022	7/1/2022 12:00:00 AM
6/15/2022	06/01/2022	6/1/2022 12:00:00 AM
6/26/2022	06/01/2022	6/1/2022 12:00:00 AM
5/7/2022	05/01/2022	5/1/2022 12:00:00 AM
5/16/2022	05/01/2022	5/1/2022 12:00:00 AM
4/1/2022	04/01/2022	4/1/2022 12:00:00 AM
3/16/2022	03/01/2022	3/1/2022 12:00:00 AM
2/5/2022	02/01/2022	2/1/2022 12:00:00 AM
2/28/2022	02/01/2022	2/1/2022 12:00:00 AM
1/7/2022	01/01/2022	1/1/2022 12:00:00 AM
1/19/2022	01/01/2022	1/1/2022 12:00:00 AM

La misura `start_of_month` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `monthstart()` e trasferendo il campo data come argomento della funzione.

La funzione `monthstart()` identifica in quale mese cade il valore della data, restituendo un timestamp per il primo millisecondo di quel mese.

Schema della funzione `monthstart()`, esempio di oggetto grafico



La transazione 8192 è avvenuta il 16 marzo. La funzione `monthstart()` identifica che la transazione ha avuto luogo nel mese di marzo e restituisce il primo millisecondo di quel mese, ovvero il 1° marzo alle 12:00:00 AM.

Esempio 4 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di saldi di prestiti, che viene caricato in una tabella chiamata `Loans`.
- Dati costituiti dagli ID dei prestiti, dal saldo all'inizio del mese e dal tasso di interesse semplice annuo applicato a ciascun prestito.

L'utente finale desidera un oggetto grafico che mostri, in base all'ID del prestito, gli interessi correnti che sono stati maturati per ciascun prestito nel mese in corso.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Loans:
Load
*
Inline
[
loan_id,start_balance,rate
8188,$10000.00,0.024
8189,$15000.00,0.057
8190,$17500.00,0.024
8191,$21000.00,0.034
8192,$90000.00,0.084
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:
   - `loan_id`
   - `start_balance`
2. Quindi, creare una misura per calcolare gli interessi accumulati:
   `=start_balance*(rate*(today(1)-monthstart(today(1)))/365)`
3. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

loan_id	start_balance	=start_balance*(rate*(today(1)-monthstart(today(1)))/365)
8188	$10000.00	$16.44
8189	$15000.00	$58.56
8190	$17500.00	$28.77
8191	$21000.00	$48.90
8192	$90000.00	$517.81

Utilizzando la data odierna come unico argomento, la funzione `monthstart()` restituisce la data di inizio del mese corrente. Sottraendo tale risultato dalla data corrente, l'espressione restituisce il numero di giorni trascorsi fino ad ora questo mese.

Questo valore viene quindi moltiplicato per il tasso di interesse e diviso per 365, per restituire il tasso di interesse effettivo incorso per questo periodo. Il risultato viene poi moltiplicato per il saldo iniziale del prestito per ottenere gli interessi maturati fino a ora in questo mese.

networkdays

La funzione **networkdays** restituisce il numero di giorni lavorativi (dal lunedì al venerdì) compresi tra e inclusi in **start_date** e **end_date**, tenendo in considerazione qualsiasi eventuale valore di festività **holiday** nel calendario.

Sintassi:

```
networkdays (start_date, end_date [, holiday])
```

Tipo di dati restituiti: numero intero

Schema del calendario che visualizza l'intervallo di date restituito dalla funzione networkdays

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10 start_date	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26 end_date	27
28	29	30	31			

La funzione `networkdays` presenta le seguenti limitazioni:

- Non esiste un metodo per modificare i giorni lavorativi. In altre parole, non c'è modo di modificare la funzione per le regioni o le situazioni che comportano un lavoro diverso dal lunedì al venerdì.
- Il parametro `holiday` deve essere una costante stringa. Le espressioni non sono accettate.

Argomenti

Argomento	Descrizione
start_date	La data di inizio da valutare.
end_date	La data di fine da valutare.
holiday	Periodi di vacanza da escludere dai giorni lavorativi. Una vacanza è definita come una stringa di data costante. È possibile specificare più periodi di vacanza, separati da virgole. **Esempio:** `'12/25/2013', '12/26/2013', '12/31/2013', '01/01/2014'`

Casi di utilizzo

La funzione `networkdays()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi il numero di giorni lavorativi che intercorrono tra due date. Ad esempio, se un utente desidera calcolare il salario totale che verrà percepito da un dipendente con contratto PAYE (pay-as-you-earn).

Esempi di funzioni

Esempio	Risultato
`networkdays ('12/19/2013', '01/07/2014')`	Restituisce 14. In questo esempio, le festività non vengono prese in considerazione.
`networkdays ('12/19/2013', '01/07/2014', '12/25/2013', '12/26/2013')`	Restituisce 12. In questo esempio, vengono presi in considerazione i giorni festivi 12/25/2013 e 12/26/2013.
`networkdays ('12/19/2013', '01/07/2014', '12/25/2013', '12/26/2013', '12/31/2013', '01/01/2014')`	Restituisce 10. In questo esempio, vengono presi in considerazione due periodi di festività.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente gli ID dei progetti, le date di inizio e di fine. Queste informazioni vengono caricate in una tabella denominata `Projects`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo aggiuntivo, `net_work_days`, per calcolare il numero di giorni lavorativi coinvolti in ogni progetto.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Projects:
```

```
    Load
        *,
        networkdays(start_date,end_date) as net_work_days
        ;
Load
id,
start_date,
end_date
Inline
[
id,start_date,end_date
1,01/01/2022,01/18/2022
2,02/10/2022,02/17/2022
3,05/17/2022,07/05/2022
4,06/01/2022,06/12/2022
5,08/10/2022,08/26/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- start_date
- end_date
- net_work_days

Tabella dei risultati

id	start_date	end_date	net_work_days
1	01/01/2022	01/18/2022	12
2	02/10/2022	02/17/2022	6
3	05/17/2022	07/05/2022	36
4	06/01/2022	06/12/2022	8
5	08/10/2022	08/26/2022	13

Poiché non ci sono festività programmate (che sarebbero state presenti nel terzo argomento della funzione networkdays()), la funzione sottrae il valore start_date da end_date, così come tutti i fine settimana, per calcolare il numero di giorni lavorativi tra le due date.

Schema di calendario che evidenzia i giorni di lavoro per il progetto 5 (senza giorni festivi)

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

Il calendario qui sopra illustra visivamente il progetto con un valore `id` di 5. Il progetto 5 inizia mercoledì 10 agosto 2022 e termina il 26 agosto 2022. Ignorando tutti i sabati e le domeniche, sono presenti 13 giorni lavorativi tra queste due date.

Esempio 2 – Singolo giorno festivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario ricavati dal precedente esempio.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo aggiuntivo, `net_work_days`, per calcolare il numero di giorni lavorativi coinvolti in ogni progetto.

In questo esempio, il 19 agosto 2022 è previsto un giorno di vacanza.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Projects:
```

```
    Load
        *,
        networkdays(start_date,end_date,'08/19/2022') as net_work_days
        ;
Load
id,
start_date,
end_date
Inline
[
id,start_date,end_date
1,01/01/2022,01/18/2022
2,02/10/2022,02/17/2022
3,05/17/2022,07/05/2022
4,06/01/2022,06/12/2022
5,08/10/2022,08/26/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- start_date
- end_date
- net_work_days

Tabella dei risultati

id	start_date	end_date	net_work_days
1	01/01/2022	01/18/2022	12
2	02/10/2022	02/17/2022	6
3	05/17/2022	07/05/2022	36
4	06/01/2022	06/12/2022	8
5	08/10/2022	08/26/2022	12

La singola vacanza programmata viene inserita come terzo argomento della funzione `networkdays()`.

Schema di calendario che evidenzia i giorni di lavoro per il progetto 5 (con un giorno festivo)

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19 Holiday	20
21	22	23	24	25	26	27
28	29	30	31			

Il calendario qui sopra illustra visivamente il progetto 5, dimostrando questo adattamento per includere il giorno di vacanza. Questa vacanza si verifica durante il progetto 5, venerdì 19 agosto 2022. Di conseguenza, il valore `net_work_days` totale del progetto 5 diminuisce di un giorno, passando da 13 a 12 giorni.

Esempio 3 – Vacanze multiple

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario ricavato dal primo esempio.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo aggiuntivo, `net_work_days`, per calcolare il numero di giorni lavorativi coinvolti in ogni progetto.

Tuttavia, in questo esempio sono previsti quattro giorni di vacanza, dal 18 al 21 agosto 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Projects:
    Load
        *,
        networkdays(start_date,end_date,'08/18/2022','08/19/2022','08/20/2022','08/21/2022')
as net_work_days
        ;
Load
id,
start_date,
end_date
Inline
[
id,start_date,end_date
1,01/01/2022,01/18/2022
2,02/10/2022,02/17/2022
3,05/17/2022,07/05/2022
4,06/01/2022,06/12/2022
5,08/10/2022,08/26/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `start_date`
- `end_date`
- `net_work_days`

Tabella dei risultati

id	start_date	end_date	net_work_days
1	01/01/2022	01/18/2022	12
2	02/10/2022	02/17/2022	6
3	05/17/2022	07/05/2022	36
4	06/01/2022	06/12/2022	8
5	08/10/2022	08/26/2022	11

I quattro giorni di vacanza programmati vengono inseriti come elenco separato da virgole, a partire dal terzo argomento della funzione `networkdays()`.

Schema di calendario che evidenzia i giorni di lavoro per il progetto 5 (con più giorni di vacanza)

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18 Holiday	19 Holiday	20
21	22	23	24	25	26	27
28	29	30	31			

Il calendario qui sopra illustra visivamente il progetto 5, dimostrando questo adattamento per includere i giorni di vacanza. Questo periodo di ferie programmate si verifica durante il progetto 5, con due giorni di giovedì e venerdì. Di conseguenza, il valore totale `net_work_days` del progetto 5 diminuisce da 13 a 11 giorni.

Esempio 4 – Singolo giorno di vacanza

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario ricavato dal primo esempio.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

Il 19 agosto 2022 è previsto un giorno di vacanza.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il campo `net_work_days` viene calcolato come misura in un oggetto grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Projects:
Load
id,
start_date,
end_date
Inline
[
id,start_date,end_date
1,01/01/2022,01/18/2022
2,02/10/2022,02/17/2022
3,05/17/2022,07/05/2022
4,06/01/2022,06/12/2022
5,08/10/2022,08/26/2022
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `start_date`
- `end_date`

Creare la seguente misura:

```
= networkdays(start_date,end_date,'08/19/2022')
```

Tabella dei risultati

id	start_date	end_date	net_work_days
1	01/01/2022	01/18/2022	12
2	02/10/2022	02/17/2022	6
3	05/17/2022	07/05/2022	36
4	06/01/2022	06/12/2022	8
5	08/10/2022	08/26/2022	12

La singola vacanza programmata viene inserita come terzo argomento della funzione `networkdays()`.

Schema del calendario che mostra i giorni di lavoro netti con una singola festività (oggetto grafico)

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19 Holiday	20
21	22	23	24	25	26	27
28	29	30	31			

Il calendario qui sopra illustra visivamente il progetto 5, dimostrando questo adattamento per includere il giorno di vacanza. Questa vacanza si verifica durante il progetto 5, venerdì 19 agosto 2022. Di conseguenza, il valore `net_work_days` totale del progetto 5 diminuisce di un giorno, passando da 13 a 12 giorni.

now

Questa funzione restituisce un indicatore temporale dell'ora attuale. La funzione restituisce i valori nel formato della variabile di sistema **TimeStamp**. Il valore predefinito **timer_mode** è 1.

Sintassi:

```
now([ timer_mode])
```

Tipo di dati restituiti: duale

La funzione `now()` può essere utilizzata sia nello script di caricamento che negli oggetti del grafico.

Argomenti

Argomento	Descrizione
timer_mode	Può presentare i valori seguenti: 0 (ora al momento dell'ultimo caricamento di dati completato) 1 (ora al momento della chiamata della funzione) 2 (ora di apertura dell'app) *Se si utilizza la funzione in uno script di caricamento dei dati, **timer_mode=0** restituirà l'ora del completamento dell'ultimo caricamento dei dati, mentre **timer_mode=1** restituirà l'ora della chiamata di funzione nel caricamento dei dati attuale.*

Casi di utilizzo

La funzione `now()` viene comunemente utilizzata come componente di un'espressione. Ad esempio, può essere utilizzato per calcolare il tempo rimanente nel ciclo di vita di un prodotto. La funzione `now()` verrebbe utilizzata al posto della funzione `today()` quando l'espressione richiede l'uso di una frazione di giorno.

La tabella seguente fornisce una spiegazione del risultato restituito dalla funzione `now()`, in presenza di diversi valori dell'argomento `timer_mode`:

Esempi di funzioni

valore timer_mode	Risultato se utilizzato nello script di caricamento	Risultato se utilizzato nell'oggetto grafico
0	Restituisce un timestamp, nel formato della variabile di sistema `TimeStamp`, dell'ultimo ricaricamento dei dati riuscito prima dell'ultimo ricaricamento dei dati.	Restituisce un timestamp, nel formato della variabile di sistema `TimeStamp`, per l'ultimo ricaricamento dei dati.
1	Restituisce un timestamp, nel formato della variabile di sistema `TimeStamp`, per l'ultimo ricaricamento dei dati.	Restituisce un timestamp, nel formato della variabile di sistema `TimeStamp`, della chiamata alla funzione.
2	Restituisce un timestamp, nel formato della variabile di sistema `TimeStamp`, per l'inizio della sessione dell'utente nell'applicazione. Non verrà aggiornato a meno che l'utente non ricarichi lo script.	Restituisce il timestamp, nel formato della variabile di sistema `TimeStamp`, per l'inizio della sessione dell'utente nell'applicazione. Questo verrà aggiornato all'inizio di una nuova sessione o quando i dati dell'applicazione verranno ricaricati.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Generazione di oggetti tramite script di caricamento

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Questo esempio crea tre variabili utilizzando la funzione `now()`. Ogni variabile utilizza una delle opzioni `timer_mode` per dimostrarne l'effetto.

Affinché le variabili dimostrino il loro scopo, e lo script e poi, dopo un breve periodo di tempo, ricaricare lo script una seconda volta. In questo modo le variabili `now(0)` e `now(1)` mostreranno valori diversi, dimostrando così correttamente il loro scopo.

Script di caricamento

```
LET vPreviousDataLoad = now(0);
LET vCurrentDataLoad = now(1);
LET vApplicationOpened = now(2);
```

Risultati

Una volta che i dati vengono caricati una seconda volta, creare tre caselle di testo seguendo le indicazioni riportate di seguito.

Per prima cosa, creare una casella di testo per i dati che sono stati caricati precedentemente.

Procedere come indicato di seguito:

1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la seguente misura all'oggetto:
   `=vPreviousDataLoad`

3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Ora di caricamento precedente' all'oggetto.

Quindi, creare una casella di testo per i dati che si stanno caricando.

Procedere come indicato di seguito:

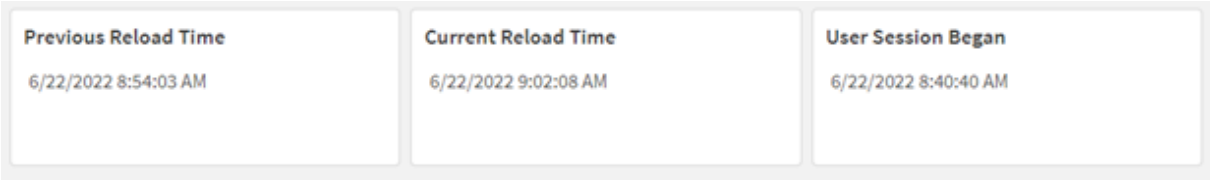
1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la seguente misura all'oggetto:
   `=vCurrentDataLoad`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Ora di caricamento attuale' all'oggetto.

Creare una casella di testo finale da mostrare quando è stata avviata la sessione dell'utente nell'applicazione.

Procedere come indicato di seguito:

1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la seguente misura all'oggetto:
   `=vApplicationOpened`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Inizio sessione utente' all'oggetto.

Variabili dello script di caricamento now()



L'immagine qui sopra mostra dei valori di esempio per ciascuna delle variabili create. Ad esempio, i valori potrebbero essere i seguenti:

- Ora di ricaricamento precedente: 22/06/2022 8:54:03 AM
- Ora di ricaricamento corrente: 22/06/2022 9:02:08 AM
- Inizio della sessione utente: 22/06/2022 8:40:40 AM

Esempio 2 – Generazione di oggetti senza script di caricamento

Script di caricamento ed espressione del grafico

Panoramica

In questo esempio, si creeranno tre oggetti grafico utilizzando la funzione `now()`, senza caricare alcuna variabile o dato nell'applicazione. Ogni oggetto del grafico utilizza una delle opzioni `timer_mode` per dimostrarne l'effetto.

Per questo esempio non esiste uno script di caricamento.

Procedere come indicato di seguito:

1. Aprire l'Editor caricamento dati.
2. Senza modificare lo script di caricamento esistente, fare clic su **Carica dati**.
3. Dopo un breve periodo di tempo, caricare lo script una seconda volta.

Risultati

Una volta che i dati vengono caricati una seconda volta, creare tre caselle di testo.

Per prima cosa, creare una casella di testo per l'ultimo ricaricamento di dati.

Procedere come indicato di seguito:

1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la misura seguente:
   `=now(0)`
3. Sotto **Aspetto**, selezionare **Show titles** e aggiungere all'oggetto il titolo 'Ricaricamento ultimi dati'.

Quindi, creare una casella di testo per mostrare l'ora corrente.

Procedere come indicato di seguito:

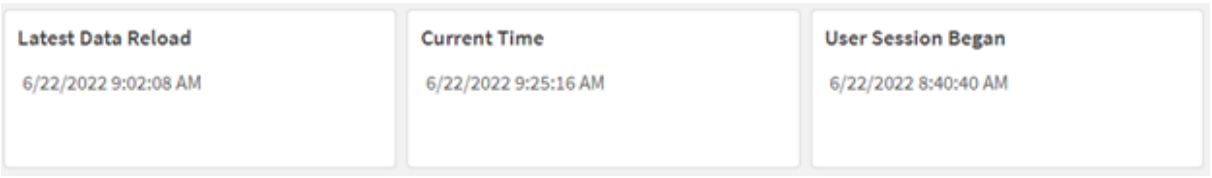
1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la misura seguente:
   `=now(1)`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Ora attuale' all'oggetto.

Creare una casella di testo finale da mostrare quando è stata avviata la sessione dell'utente nell'applicazione.

Procedere come indicato di seguito:

1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la misura seguente:
   `=now(2)`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Sessione utente iniziata' all'oggetto.

Esempi oggetto grafico now()



L'immagine qui sopra mostra dei valori di esempio per ciascuno degli oggetti creati. Ad esempio, i valori potrebbero essere i seguenti:

- Dati ultimo ricaricamento: 22/6/2022 9:02:08 AM
- Ora corrente: 22/6/2022 9:25:16 AM
- Inizio della sessione utente: 22/06/2022 8:40:40 AM

L'oggetto grafico 'Ultimo caricamento dati' utilizza un valore `timer_mode` pari a 0. Restituisce il timestamp dell'ultima volta in cui i dati sono stati ricaricati correttamente.

L'oggetto grafico 'Ora corrente' utilizza un valore `timer_mode` di 1. Questo restituisce l'ora corrente secondo l'orologio di sistema. Se il foglio o l'oggetto viene ricaricato, questo valore verrà aggiornato.

L'oggetto grafico 'Sessione utente iniziata' utilizza un valore `timer_mode` di 2. Restituisce il timestamp di quando l'applicazione è stata aperta e la sessione dell'utente è iniziata.

Esempio 3 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati costituito dall'inventario di un'operazione di estrazione di criptovalute, che viene caricato in una tabella denominata `Inventory`.
- Dati con i seguenti campi: `id`, `purchase_date` e `wph` (watt all'ora).

L'utente vorrebbe una tabella che visualizzi, per `id`, il costo totale che ogni impianto di mining ha sostenuto nel mese fino a quel momento, in termini di consumo energetico.

Questo valore deve essere aggiornato ogni volta che l'oggetto grafico viene aggiornato. Il costo attuale dell'elettricità è di 0,0678 dollari per kWH.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Inventory:
Load
*
Inline
[
id,purchase_date,wph
8188,1/7/2022,1123
8189,1/19/2022,1432
8190,2/28/2022,1227
8191,2/5/2022,1322
8192,3/16/2022,1273
8193,4/1/2022,1123
8194,5/7/2022,1342
8195,5/16/2022,2342
```

```
8196,6/15/2022,1231
8197,6/26/2022,1231
8198,7/9/2022,1123
8199,7/22/2022,1212
8200,7/23/2022,1223
8201,7/27/2022,1232
8202,8/2/2022,1232
8203,8/8/2022,1211
8204,8/19/2022,1243
8205,9/26/2022,1322
8206,10/14/2022,1133
8207,10/29/2022,1231
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `id`.

Creare la seguente misura:

```
=(now(1)-monthstart(now(1)))*24*wph/1000*0.0678
```

Se l'oggetto grafico viene aggiornato il 22/06/2022 alle 10:39:05, i risultati sono i seguenti:

Tabella dei risultati

id	=(now(1)-monthstart(now(1)))*24*wph/1000*0.0678
8188	$39.18
8189	$49.97
8190	$42.81
8191	$46.13
8192	$44.42
8193	$39.18
8194	$46.83
8195	$81.72
8196	$42.95
8197	$42.95
8198	$39.18
8199	$42.29
8200	$42.67
8201	$42.99
8202	$42.99

id	=(now(1)-monthstart(now(1)))*24*wph/1000*0.0678
8203	$42.25
8204	$43.37
8205	$46.13
8206	$39.53

L'utente desidera che i risultati dell'oggetto vengano aggiornati ogni volta che si aggiorna l'oggetto. Pertanto, l'argomento `timer_mode` viene fornito per le istanze della funzione `now()`nell'espressione. Il timestamp dell'inizio del mese, identificato dalla funzione `now()` come argomento timestamp nella funzione `monthstart()`, viene sottratto dall'ora corrente identificata dalla funzione `now()`. Fornisce il tempo totale trascorso finora nel mese, in giorni.

Questo valore viene moltiplicato per 24 (il numero di ore in un giorno) e quindi per il valore del campo `wph`.

Per convertire i watt all'ora in kilowatt all'ora, il risultato viene diviso per 1000 prima di essere moltiplicato per la tariffa kWH fornita.

quarterend

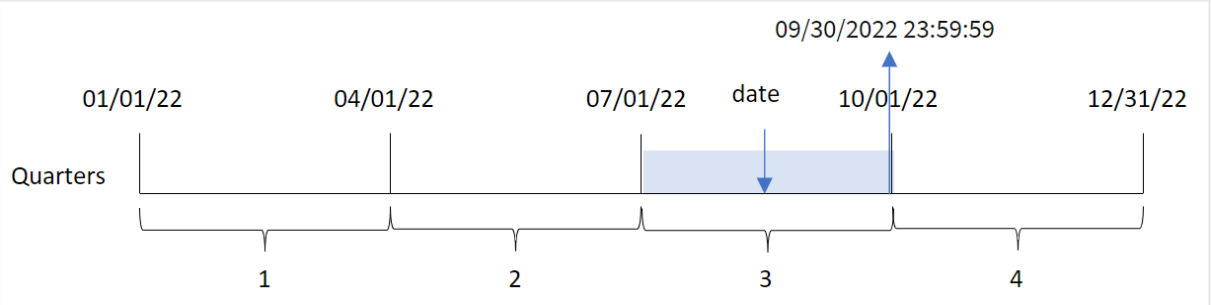
Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo del trimestre contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

Sintassi:

```
QuarterEnd(date[, period_no[, first_month_of_year]])
```

Tipo di dati restituiti: duale

Schema della funzione `quarterend()`



In altre parole, la funzione `quarterend()` determina in quale trimestre ricadrà la data. Quindi restituisce un timestamp, in formato data, per l'ultimo millisecondo dell'ultimo mese di quel trimestre. Il primo mese dell'anno è, per impostazione predefinita, gennaio. Tuttavia, è possibile modificare il mese da impostare come primo utilizzando l'argomento `first_month_of_year` nella funzione `quarterend()`.

La funzione `quarterend()` non considera la variabile di sistema `FirstMonthOfYear`. L'anno inizia il 1 gennaio a condizione che l'argomento `first_month_of_year` non venga utilizzato per modificarlo.

Casi di utilizzo

La funzione `quarterend()` viene comunemente utilizzata come parte di un'espressione quando si desidera che il calcolo utilizzi la frazione del trimestre non ancora trascorsa. Ad esempio, se si vuole calcolare l'interesse totale non ancora maturato durante il trimestre.

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero, in cui il valore 0 indica il trimestre che contiene **date**. I valori negativi di **period_no** indicano i trimestri precedenti, mentre i valori positivi indicano i trimestri successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

È possibile utilizzare i seguenti valori per impostare il primo mese dell'anno nell'argomento `first_month_of_year`:

valori first_month_of_year

Month	Valore
Febbraio	2
March	3
April	4
May	5
June	6
July	7
August	8
September	9
October	10
Novembre	11
December	12

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`quarterend('10/29/2005')`	Returns 12/31/2005 23:59:59.
`quarterend('10/29/2005', -1)`	Returns 09/30/2005 23:59:59.
`quarterend('10/29/2005', 0, 3)`	Returns 11/30/2005 23:59:59.

Esempio 1 - Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
  - La funzione `quarterend()` impostata come campo '`end_of_quarter`' restituisce un timestamp per la fine del trimestre in cui sono avvenute le transazioni.
  - La funzione `timestamp()` impostata come campo '`end_of_quarter_timestamp`' restituisce il timestamp esatto della fine del trimestre selezionato.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
```

```
       quarterend(date) as end_of_quarter,
       timestamp(quarterend(date)) as end_of_quarter_timestamp
       ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- date
- end_of_quarter
- end_of_quarter_timestamp

Tabella dei risultati

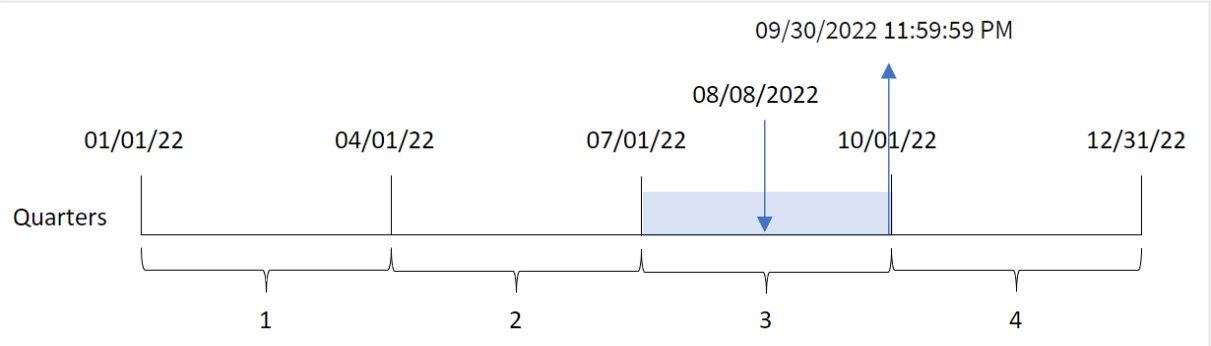
id	date	end_of_quarter	end_of_quarter_timestamp
8188	1/7/2022	03/31/2022	3/31/2022 11:59:59 PM
8189	1/19/2022	03/31/2022	3/31/2022 11:59:59 PM
8190	2/5/2022	03/31/2022	3/31/2022 11:59:59 PM
8191	2/28/2022	03/31/2022	3/31/2022 11:59:59 PM
8192	3/16/2022	03/31/2022	3/31/2022 11:59:59 PM
8193	4/1/2022	06/30/2022	6/30/2022 11:59:59 PM

id	date	end_of_quarter	end_of_quarter_timestamp
8194	5/7/2022	06/30/2022	6/30/2022 11:59:59 PM
8195	5/16/2022	06/30/2022	6/30/2022 11:59:59 PM
8196	6/15/2022	06/30/2022	6/30/2022 11:59:59 PM
8197	6/26/2022	06/30/2022	6/30/2022 11:59:59 PM
8198	7/9/2022	09/30/2022	9/30/2022 11:59:59 PM
8199	7/22/2022	09/30/2022	9/30/2022 11:59:59 PM
8200	7/23/2022	09/30/2022	9/30/2022 11:59:59 PM
8201	7/27/2022	09/30/2022	9/30/2022 11:59:59 PM
8202	8/2/2022	09/30/2022	9/30/2022 11:59:59 PM
8203	8/8/2022	09/30/2022	9/30/2022 11:59:59 PM
8204	8/19/2022	09/30/2022	9/30/2022 11:59:59 PM
8205	9/26/2022	09/30/2022	9/30/2022 11:59:59 PM
8206	10/14/2022	12/31/2022	12/31/2022 11:59:59 PM
8207	10/29/2022	12/31/2022	12/31/2022 11:59:59 PM

Il campo '`end_of_quarter`' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `quarterend()` e trasferendo il campo data come argomento della funzione.

La funzione `quarterend()` inizialmente identifica in quale trimestre ricada il valore della data e quindi restituisce un timestamp per l'ultimo millisecondo di quel trimestre.

Schema della funzione `quarterend()` con identificazione della fine del trimestre della transazione 8203



La transazione 8203 è avvenuta l'8 agosto. La funzione `quarterend()` identifica che la transazione è avvenuta nel terzo trimestre e restituisce l'ultimo millisecondo di quel trimestre, ovvero il 30 settembre alle 23:59:59.

Esempio 2 - period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
  - La funzione `quarterend()` impostata come campo '`previous_quarter_end`' restituisce un timestamp per la fine del trimestre precedente la transazione.
  - La funzione `timestamp()` impostata come campo '`previous_end_of_quarter_timestamp`' restituisce il timestamp esatto della fine del trimestre precedente la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       quarterend(date, -1) as previous_quarter_end,
       timestamp(quarterend(date, -1)) as previous_quarter_end_timestamp
       ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
```

```
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

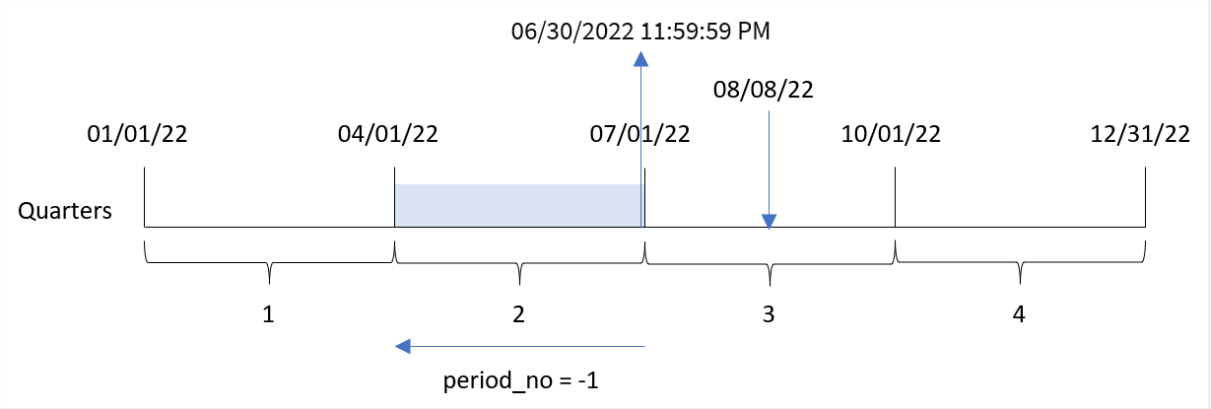
- id
- date
- previous_quarter_end
- previous_quarter_end_timestamp

Tabella dei risultati

id	date	previous_quarter_end	previous_quarter_end_timestamp
8188	1/7/2022	12/31/2021	12/31/2021 11:59:59 PM
8189	1/19/2022	12/31/2021	12/31/2021 11:59:59 PM
8190	2/5/2022	12/31/2021	12/31/2021 11:59:59 PM
8191	2/28/2022	12/31/2021	12/31/2021 11:59:59 PM
8192	3/16/2022	12/31/2021	12/31/2021 11:59:59 PM
8193	4/1/2022	03/31/2022	3/31/2022 11:59:59 PM
8194	5/7/2022	03/31/2022	3/31/2022 11:59:59 PM
8195	5/16/2022	03/31/2022	3/31/2022 11:59:59 PM
8196	6/15/2022	03/31/2022	3/31/2022 11:59:59 PM
8197	6/26/2022	03/31/2022	3/31/2022 11:59:59 PM
8198	7/9/2022	06/30/2022	6/30/2022 11:59:59 PM
8199	7/22/2022	06/30/2022	6/30/2022 11:59:59 PM
8200	7/23/2022	06/30/2022	6/30/2022 11:59:59 PM
8201	7/27/2022	06/30/2022	6/30/2022 11:59:59 PM
8202	8/2/2022	06/30/2022	6/30/2022 11:59:59 PM
8203	8/8/2022	06/30/2022	6/30/2022 11:59:59 PM
8204	8/19/2022	06/30/2022	6/30/2022 11:59:59 PM
8205	9/26/2022	06/30/2022	6/30/2022 11:59:59 PM
8206	10/14/2022	09/30/2022	9/30/2022 11:59:59 PM
8207	10/29/2022	09/30/2022	9/30/2022 11:59:59 PM

Poiché il valore `period_no` di `-1` viene utilizzato come argomento offset nella funzione `quarterend()`, la funzione per prima cosa identifica il trimestre in cui avvengono le transazioni. Si sposta poi un trimestre prima e identifica il millisecondo finale di tale trimestre.

Schema della funzione `quarterend()` con un valore period_no di -1



La transazione 8203 è avvenuta l'8 agosto. La funzione `quarterend()` identifica che il trimestre precedente la transazione era compreso tra il 1° aprile e il 30 giugno. La funzione restituisce quindi il millisecondo finale di quel trimestre, il 30 giugno alle 23:59:59.

Esempio 3 - first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
  - La funzione `quarterend()` impostata come campo '`end_of_quarter`' restituisce un timestamp per la fine del trimestre in cui sono avvenute le transazioni.
  - La funzione `timestamp()` impostata come campo '`end_of_quarter_timestamp`' restituisce il timestamp esatto della fine del trimestre selezionato.

Tuttavia, in questo esempio, la politica aziendale prevede che l'esercizio finanziario inizi il 1° marzo.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       quarterend(date, 0, 3) as end_of_quarter,
```

```
       timestamp(quarterend(date, 0, 3)) as end_of_quarter_timestamp
       ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

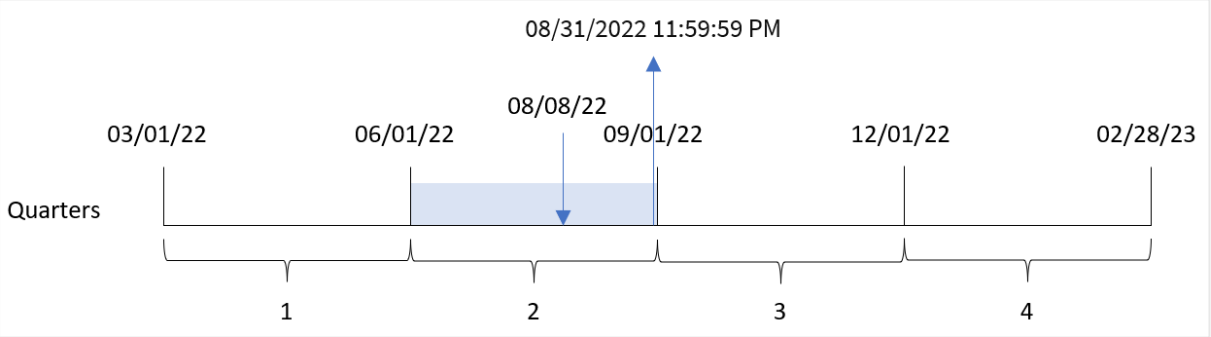
Tabella dei risultati

id	date	end_of_quarter	end_of_quarter_timestamp
8188	1/7/2022	02/28/2022	2/28/2022 11:59:59 PM
8189	1/19/2022	02/28/2022	2/28/2022 11:59:59 PM
8190	2/5/2022	02/28/2022	2/28/2022 11:59:59 PM
8191	2/28/2022	02/28/2022	2/28/2022 11:59:59 PM
8192	3/16/2022	05/31/2022	5/31/2022 11:59:59 PM
8193	4/1/2022	05/31/2022	5/31/2022 11:59:59 PM
8194	5/7/2022	05/31/2022	5/31/2022 11:59:59 PM
8195	5/16/2022	05/31/2022	5/31/2022 11:59:59 PM
8196	6/15/2022	08/31/2022	8/31/2022 11:59:59 PM
8197	6/26/2022	08/31/2022	8/31/2022 11:59:59 PM
8198	7/9/2022	08/31/2022	8/31/2022 11:59:59 PM

id	date	end_of_quarter	end_of_quarter_timestamp
8199	7/22/2022	08/31/2022	8/31/2022 11:59:59 PM
8200	7/23/2022	08/31/2022	8/31/2022 11:59:59 PM
8201	7/27/2022	08/31/2022	8/31/2022 11:59:59 PM
8202	8/2/2022	08/31/2022	8/31/2022 11:59:59 PM
8203	8/8/2022	08/31/2022	8/31/2022 11:59:59 PM
8204	8/19/2022	08/31/2022	8/31/2022 11:59:59 PM
8205	9/26/2022	11/30/2022	11/30/2022 11:59:59 PM
8206	10/14/2022	11/30/2022	11/30/2022 11:59:59 PM
8207	10/29/2022	11/30/2022	11/30/2022 11:59:59 PM

Poiché nella funzione `quarterend()` viene utilizzato l'argomento `first_month_of_year` di `3`, l'inizio dell'anno si sposta dal 1° gennaio al 1° marzo.

Lo schema della funzione `quarterend()` con marzo impostato come primo mese dell'anno



La transazione 8203 è avvenuta l'8 agosto. Poiché l'inizio dell'anno è il 1° marzo, i trimestri dell'anno sono compresi tra marzo-maggio, giugno-agosto, settembre-novembre e dicembre-febbraio.

La funzione `quarterend()` identifica che la transazione ha avuto luogo nel trimestre compreso tra l'inizio di giugno e agosto e restituisce l'ultimo millisecondo di quel trimestre, ovvero il 31 agosto alle 23:59:59.

Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che restituisce un timestamp per la fine del trimestre in cui sono avvenute le transazioni viene creato come misura in un grafico nell'app.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`

Per calcolare la data di fine del trimestre in cui avviene una transazione, creare le seguenti misure:

- `=quarterend(date)`
- `=timestamp(quarterend(date))`

Tabella dei risultati

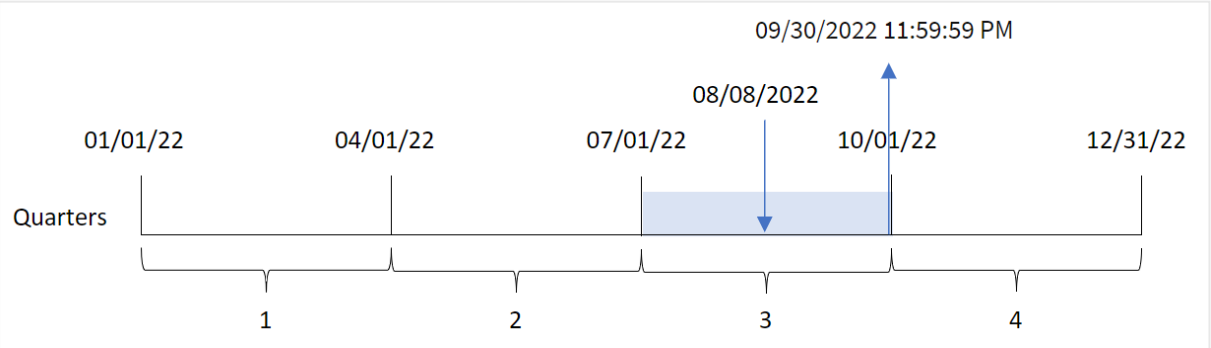
id	date	=quarterend(date)	=timestamp(quarterend(date))
8188	1/7/2022	03/31/2022	3/31/2022 11:59:59 PM
8189	1/19/2022	03/31/2022	3/31/2022 11:59:59 PM
8190	2/5/2022	03/31/2022	3/31/2022 11:59:59 PM

id	date	=quarterend(date)	=timestamp(quarterend(date))
8191	2/28/2022	03/31/2022	3/31/2022 11:59:59 PM
8192	3/16/2022	03/31/2022	3/31/2022 11:59:59 PM
8193	4/1/2022	06/30/2022	6/30/2022 11:59:59 PM
8194	5/7/2022	06/30/2022	6/30/2022 11:59:59 PM
8195	5/16/2022	06/30/2022	6/30/2022 11:59:59 PM
8196	6/15/2022	06/30/2022	6/30/2022 11:59:59 PM
8197	6/26/2022	06/30/2022	6/30/2022 11:59:59 PM
8198	7/9/2022	09/30/2022	9/30/2022 11:59:59 PM
8199	7/22/2022	09/30/2022	9/30/2022 11:59:59 PM
8200	7/23/2022	09/30/2022	9/30/2022 11:59:59 PM
8201	7/27/2022	09/30/2022	9/30/2022 11:59:59 PM
8202	8/2/2022	09/30/2022	9/30/2022 11:59:59 PM
8203	8/8/2022	09/30/2022	9/30/2022 11:59:59 PM
8204	8/19/2022	09/30/2022	9/30/2022 11:59:59 PM
8205	9/26/2022	09/30/2022	9/30/2022 11:59:59 PM
8206	10/14/2022	12/31/2022	12/31/2022 11:59:59 PM
8207	10/29/2022	12/31/2022	12/31/2022 11:59:59 PM

Il campo '`end_of_quarter`' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `quarterend()` e trasferendo il campo data come argomento della funzione.

La funzione `quarterend()` inizialmente identifica in quale trimestre ricada il valore della data e quindi restituisce un timestamp per l'ultimo millisecondo di quel trimestre.

Schema della funzione `quarterend()` con identificazione della fine del trimestre della transazione 8203



La transazione 8203 è avvenuta l'8 agosto. La funzione `quarterend()` identifica che la transazione è avvenuta nel terzo trimestre e restituisce l'ultimo millisecondo di quel trimestre, ovvero il 30 settembre alle 23:59:59.

Esempio 5 - Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata '`Employee_Expenses`'. La tabella contiene i seguenti campi:
    - ID dipendenti
    - Nomi dipendenti
    - La media delle richieste di rimborso spese giornaliero di ciascun dipendente.

L'utente finale desidera un oggetto grafico che visualizzi, in base all'ID dipendente e al nome del dipendente, le richieste di rimborso spese stimate ancora da sostenere per il resto del trimestre. L'esercizio finanziario inizia a gennaio.

Script di caricamento

```
Employee_Expenses:
Load
*
Inline
[
employee_id,employee_name,avg_daily_claim
182,Mark, $15
183,Deryck, $12.5
184,Dexter, $12.5
185,Sydney,$27
186,Agatha,$18
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `employee_id`
- `employee_name`

Per calcolare gli interessi maturati, creare la seguente misura:

- `=(quarterend(today(1))-today(1))*avg_daily_claim`

Impostare la misura **Formattazione numero** su **Denaro**.

Tabella dei risultati

employee_id	employee_name	=(quarterend(today(1))-today(1))*avg_daily_claim
182	Contrassegno	$480.00
183	Deryck	$400.00
184	Dexter	$400.00
185	Sydney	$864.00
186	Agatha	$576.00

La funzione `quarterend()` utilizza come unico argomento la data odierna e restituisce la data finale del mese corrente. Quindi, sottrae la data odierna dalla data di fine anno e l'espressione restituisce il numero di giorni rimanenti nel mese.

Questo valore viene quindi moltiplicato per la media delle richieste di rimborso spese giornaliere di ciascun dipendente per calcolare il valore stimato delle richieste che ogni dipendente dovrebbe presentare durante il periodo rimanente del trimestre.

quartername

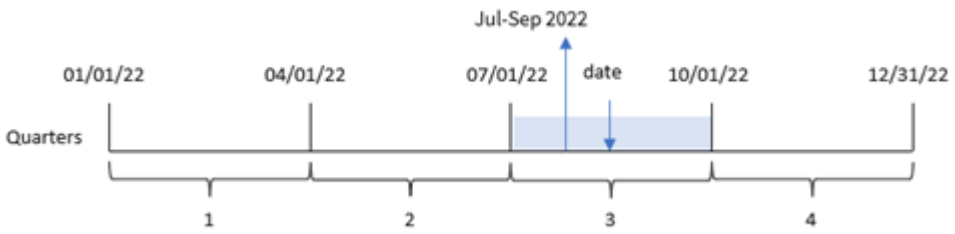
Questa funzione restituisce un valore di visualizzazione che mostra i mesi del trimestre (formattati in base alla variabile di script **MonthNames**) e l'anno con valore numerico sottostante corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno del trimestre.

Sintassi:

```
QuarterName(date[, period_no[, first_month_of_year]])
```

Tipo di dati restituiti: duale

Schema della funzione `quartername()`.



In altre parole, la funzione `quartername()` determina in quale trimestre ricadrà la data. Restituisce quindi un valore che mostra i mesi iniziali e finali di questo trimestre e dell'anno. Il valore numerico sottostante di questo risultato è il primo millisecondo del trimestre.

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero, in cui il valore 0 indica il trimestre che contiene **date**. I valori negativi di **period_no** indicano i trimestri precedenti, mentre i valori positivi indicano i trimestri successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

Casi di utilizzo

La funzione `quartername()` è utile quando si desidera confrontare le aggregazioni per trimestre. Ad esempio, può essere usata se si desidera visualizzare le vendite totali dei prodotti in base al trimestre.

Questa funzione può essere utilizzata nello script di caricamento per creare un campo in una tabella del Calendario principale. In alternativa, può essere utilizzata direttamente in un grafico come dimensione calcolata.

In questi esempi viene utilizzato il formato della data (MM/GG/AAAA). Il formato della data viene specificato nell'istruzione `SET DateFormat` nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di funzioni

Esempio	Risultato
`quartername('10/29/2013')`	Restituisce Oct-Dec 2013.
`quartername('10/29/2013', -1)`	Restituisce Jul-Sep 2013.
`quartername('10/29/2013', 0, 3)`	Restituisce Sep-Nov 2013.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – data senza alcun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `transaction_quarter`, che restituisce il trimestre in cui sono avvenute le transazioni.

Aggiungere qui altro testo, se necessario, con elenchi, ecc.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';

Transactions:
    Load
        *,
        quartername(date) as transaction_quarter
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
```

```
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `transaction_quarter`

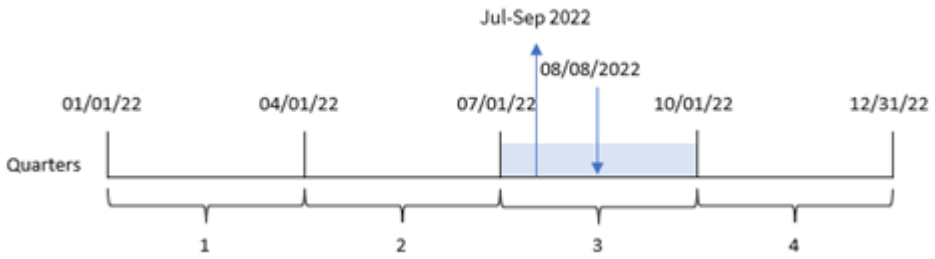
Tabella dei risultati

date	transaction_quarter
1/7/2022	gen-mar 2022
1/19/2022	gen-mar 2022
2/5/2022	gen-mar 2022
2/28/2022	gen-mar 2022
3/16/2022	gen-mar 2022
4/1/2022	apr-giu 2022
5/7/2022	apr-giu 2022
5/16/2022	apr-giu 2022
6/15/2022	apr-giu 2022
6/26/2022	apr-giu 2022
7/9/2022	lug-set 2022
7/22/2022	lug-set 2022
7/23/2022	lug-set 2022
7/27/2022	lug-set 2022
8/2/2022	lug-set 2022
8/8/2022	lug-set 2022
8/19/2022	lug-set 2022
9/26/2022	lug-set 2022
10/14/2022	ott-dic 2022
10/29/2022	ott-dic 2022

Il campo `transaction_quarter` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `quartername()` e trasferendo il campo data come argomento della funzione.

La funzione `quartername()` identifica inizialmente il trimestre in cui rientra il valore della data. Restituisce quindi un valore che mostra i mesi iniziali e finali di questo trimestre e dell'anno.

Schema della funzione `quartername()`, esempio senza argomenti aggiuntivi



La transazione 8203 è avvenuta l'8 agosto 2022. La funzione `quartername()` identifica che la transazione è avvenuta nel terzo trimestre e quindi restituisce luglio-settembre 2022. I mesi vengono visualizzati nello stesso formato della variabile di sistema `MonthNames`.

Esempio 2 – data con argomento period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_quarter`, che restituisce il trimestre precedente in cui sono state effettuate le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';

Transactions:
    Load
        *,
        quartername(date,-1) as previous_quarter
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
```

```
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_quarter`

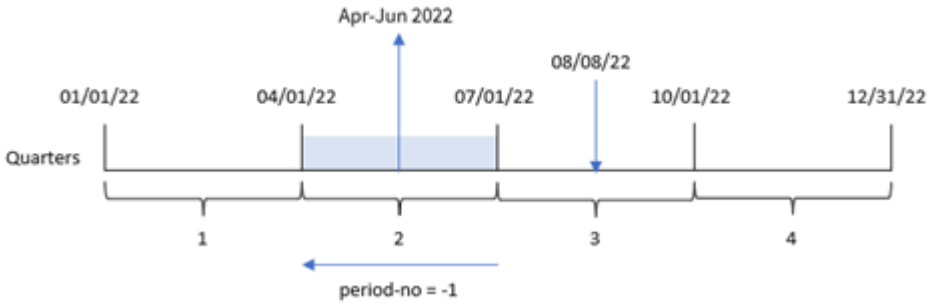
Tabella dei risultati

date	previous_quarter
1/7/2022	ott-dic 2021
1/19/2022	ott-dic 2021
2/5/2022	ott-dic 2021
2/28/2022	ott-dic 2021
3/16/2022	ott-dic 2021
4/1/2022	gen-mar 2022
5/7/2022	gen-mar 2022
5/16/2022	gen-mar 2022
6/15/2022	gen-mar 2022
6/26/2022	gen-mar 2022
7/9/2022	apr-giu 2022
7/22/2022	apr-giu 2022
7/23/2022	apr-giu 2022
7/27/2022	apr-giu 2022
8/2/2022	apr-giu 2022
8/8/2022	apr-giu 2022
8/19/2022	apr-giu 2022

date	previous_quarter
9/26/2022	apr-giu 2022
10/14/2022	lug-set 2022
10/29/2022	lug-set 2022

In questo caso, poiché il valore `period_no` di -1 è stato utilizzato come argomento offset nella funzione `quartername()`, la funzione per prima cosa identifica le transazioni avvenute nel terzo trimestre. Passa quindi al trimestre precedente e restituisce un valore che mostra i mesi iniziali e finali di questo trimestre e dell'anno.

Schema della funzione `quartername()`, esempio di period_no



La transazione 8203 è avvenuta l'8 agosto. La funzione `quartername()` identifica che il trimestre precedente la transazione era compreso tra il 1° aprile e il 30 giugno. Pertanto, restituisce Apr-Giu 2022.

Esempio 3 – data con argomento first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, dobbiamo impostare il 1° marzo come inizio dell'anno finanziario.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET MonthNames='Jan;Feb;Mar;Apr;May;Jun;Jul;Aug;Sep;Oct;Nov;Dec';

Transactions:
    Load
        *,
        quartername(date,0,3) as transaction_quarter
        ;
Load
*
```

```
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `transaction_quarter`

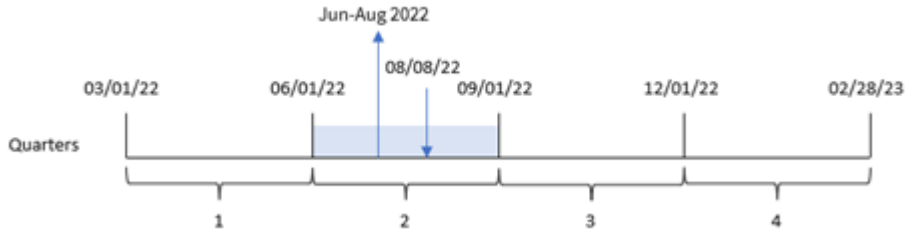
Tabella dei risultati

date	transaction_quarter
1/7/2022	Dic-Feb 2021
1/19/2022	Dic-Feb 2021
2/5/2022	Dic-Feb 2021
2/28/2022	Dic-Feb 2021
3/16/2022	Mar-Mag 2022
4/1/2022	Mar-Mag 2022
5/7/2022	Mar-Mag 2022
5/16/2022	Mar-Mag 2022
6/15/2022	Giu-Ago 2022
6/26/2022	Giu-Ago 2022

date	transaction_quarter
7/9/2022	Giu-Ago 2022
7/22/2022	Giu-Ago 2022
7/23/2022	Giu-Ago 2022
7/27/2022	Giu-Ago 2022
8/2/2022	Giu-Ago 2022
8/8/2022	Giu-Ago 2022
8/19/2022	Giu-Ago 2022
9/26/2022	Set-Nov 2022
10/14/2022	Set-Nov 2022
10/29/2022	Set-Nov 2022

In questo caso, poiché nella funzione `quartername()` viene utilizzato l'argomento `first_month_of_year` di 3, l'inizio dell'anno si sposta dal 1° gennaio al 1° marzo. Pertanto, i trimestri dell'anno sono separati in marzo-maggio, giugno-agosto, settembre-novembre e dicembre-febbraio.

Schema della funzione `quartername()`, esempio first_week_day



La transazione 8203 è avvenuta l'8 agosto. La funzione `quartername()` identifica che la transazione è avvenuta nel secondo trimestre, tra l'inizio di giugno e la fine di agosto. Pertanto, restituisce Giu-Ago 2022.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce un timestamp per la fine del trimestre in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Creare la seguente misura:

```
=quartername(date)
```

Tabella dei risultati

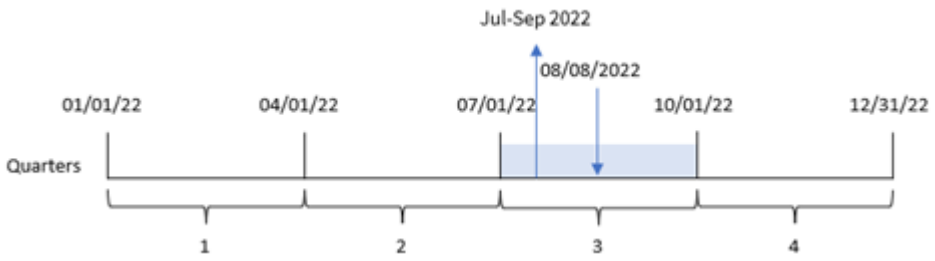
date	=quartername(date)
1/7/2022	gen-mar 2022
1/19/2022	gen-mar 2022
2/5/2022	gen-mar 2022
2/28/2022	gen-mar 2022
3/16/2022	gen-mar 2022
4/1/2022	apr-giu 2022
5/7/2022	apr-giu 2022

date	=quartername(date)
5/16/2022	apr-giu 2022
6/15/2022	apr-giu 2022
6/26/2022	apr-giu 2022
7/9/2022	lug-set 2022
7/22/2022	lug-set 2022
7/23/2022	lug-set 2022
7/27/2022	lug-set 2022
8/2/2022	lug-set 2022
8/8/2022	lug-set 2022
8/19/2022	lug-set 2022
9/26/2022	lug-set 2022
10/14/2022	ott-dic 2022
10/29/2022	ott-dic 2022

La misura `transaction_quarter` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `quartername()` e trasferendo il campo `date` come argomento della funzione.

La funzione `quartername()` identifica inizialmente il trimestre in cui rientra il valore della data. Restituisce quindi un valore che mostra i mesi iniziali e finali di questo trimestre e dell'anno.

Schema della funzione `quartername()`, esempio di oggetto grafico



La transazione 8203 è avvenuta l'8 agosto 2022. La funzione `quartername()` identifica che la transazione è avvenuta nel terzo trimestre e quindi restituisce luglio-settembre 2022. I mesi vengono visualizzati nello stesso formato della variabile di sistema `MonthNames`.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

L'utente finale desidera un oggetto grafico che presenti le vendite totali per trimestre per le transazioni. Questo può essere ottenuto anche quando la dimensione non è disponibile nel modello dati, utilizzando la funzione `quartername()` come dimensione calcolata nel grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'1/7/2022',17.17
8189,'1/19/2022',37.23
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'5/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'6/26/2022',45.89
8198,'7/9/2022',36.23
8199,'7/22/2022',25.66
8200,'7/23/2022',82.77
8201,'7/27/2022',69.98
8202,'8/2/2022',76.11
8203,'8/8/2022',25.12
8204,'8/19/2022',46.23
8205,'9/26/2022',84.21
8206,'10/14/2022',96.24
8207,'10/29/2022',67.67
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Creare una dimensione calcolata utilizzando la seguente espressione:
   `=quartername(date)`
3. Quindi, calcolare le vendite totali utilizzando la seguente misura di aggregazione:
   `=sum(amount)`
4. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

=quartername(date)	=sum(amount)
lug-set 2022	$446.31
apr-giu 2022	$351.48
gen-mar 2022	$253.89
ott-dic 2022	$163.91

quarterstart

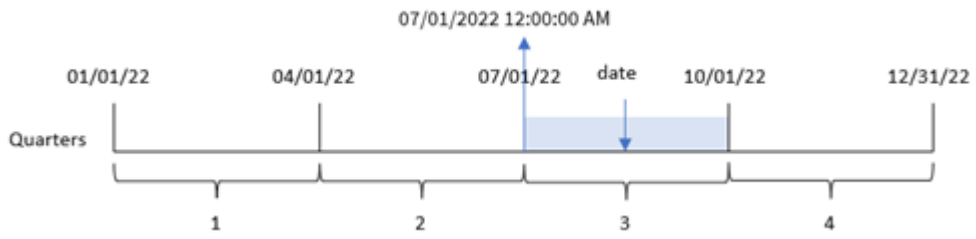
Questa funzione restituisce un valore corrispondente a un indicatore temporale recante il primo millisecondo del trimestre contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

Sintassi:

```
QuarterStart(date[, period_no[, first_month_of_year]])
```

Tipo di dati restituiti: duale

Schema della funzione `quarterstart()`.



In altre parole, la funzione `quarterstart()` determina in quale trimestre ricadrà `date`. Quindi restituisce un timestamp, in formato data, per l'ultimo millisecondo del primo mese di quel trimestre.

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero, in cui il valore 0 indica il trimestre che contiene **date**. I valori negativi di **period_no** indicano i trimestri precedenti, mentre i valori positivi indicano i trimestri successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

Casi di utilizzo

La funzione `quarterstart()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione del trimestre trascorsa finora. Ad esempio, potrebbe essere utilizzata se un utente desidera calcolare gli interessi accumulati in un trimestre fino a oggi.

Esempi di funzioni

Esempio	Risultato
`quarterstart('10/29/2005')`	Restituisce 10/01/2005.
`quarterstart('10/29/2005', -1 )`	Restituisce 07/01/2005.
`quarterstart('10/29/2005', 0, 3)`	Restituisce 09/01/2005.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `start_of_quarter`, che restituisce un timestamp per l'inizio del trimestre in cui sono avvenute le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        quarterstart(date) as start_of_quarter,
        timestamp(quarterstart(date)) as start_of_quarter_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

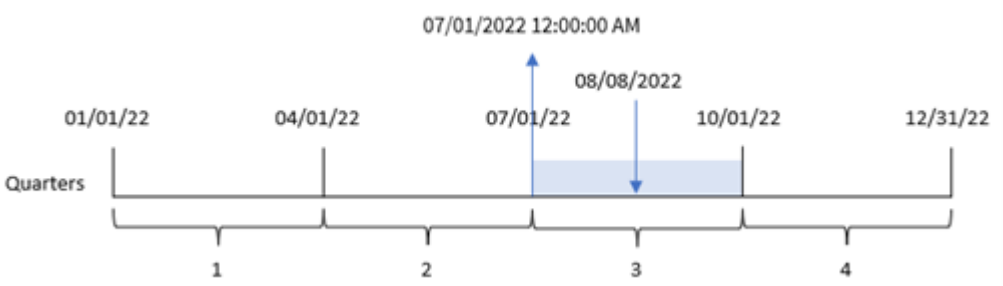
- `date`
- `start_of_quarter`
- `start_of_quarter_timestamp`

Tabella dei risultati

date	start_of_quarter	start_of_quarter_timestamp
1/7/2022	01/01/2022	1/1/2022 12:00:00 AM
1/19/2022	01/01/2022	1/1/2022 12:00:00 AM
2/5/2022	01/01/2022	1/1/2022 12:00:00 AM
2/28/2022	01/01/2022	1/1/2022 12:00:00 AM
3/16/2022	01/01/2022	1/1/2022 12:00:00 AM
4/1/2022	04/01/2022	4/1/2021 12:00:00 AM
5/7/2022	04/01/2022	4/1/2021 12:00:00 AM
5/16/2022	04/01/2022	4/1/2021 12:00:00 AM
6/15/2022	04/01/2022	4/1/2021 12:00:00 AM
6/26/2022	04/01/2022	4/1/2021 12:00:00 AM
7/9/2022	07/01/2022	7/1/2021 12:00:00 AM
7/22/2022	07/01/2022	7/1/2021 12:00:00 AM
7/23/2022	07/01/2022	7/1/2021 12:00:00 AM
7/27/2022	07/01/2022	7/1/2021 12:00:00 AM
8/2/2022	07/01/2022	7/1/2021 12:00:00 AM
8/8/2022	07/01/2022	7/1/2021 12:00:00 AM
8/19/2022	07/01/2022	7/1/2021 12:00:00 AM
9/26/2022	07/01/2022	7/1/2021 12:00:00 AM
10/14/2022	10/01/2022	10/1/2022 12:00:00 AM
10/29/2022	10/01/2022	10/1/2022 12:00:00 AM

Il campo `start_of_quarter` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `quarterstart()` e trasferendo il campo data come argomento della funzione. La funzione `uarterstart()` identifica inizialmente il trimestre in cui rientra il valore della data. Quindi restituisce un timestamp per il primo millisecondo di quel trimestre.

Schema della funzione `quarterstart()`, esempio senza argomenti aggiuntivi



La transazione 8203 è avvenuta l'8 agosto. La funzione `quarterstart()` identifica che la transazione è avvenuta nel terzo trimestre e restituisce il primo millisecondo del trimestre, ovvero il 1° luglio alle 12:00:00.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_quarter_start`, che restituisce il timestamp per l'inizio del trimestre prima che fosse effettuata la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        quarterstart(date,-1) as previous_quarter_start,
        timestamp(quarterstart(date,-1)) as previous_quarter_start_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
```

```
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_quarter_start`
- `previous_quarter_start_timestamp`

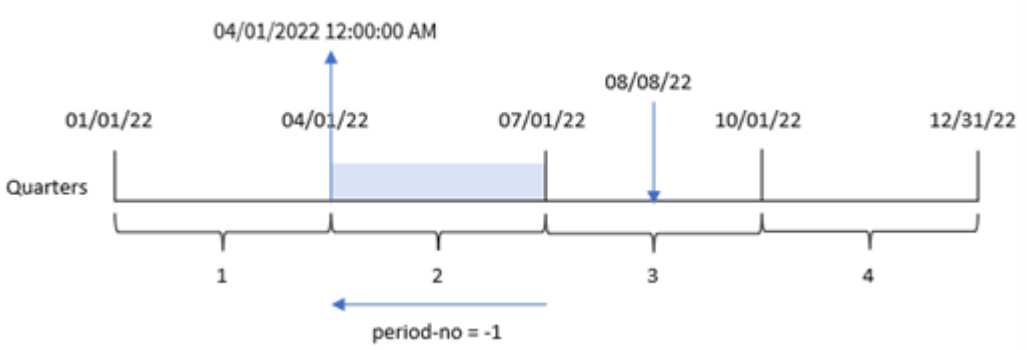
Tabella dei risultati

date	previous_quarter_start	previous_quarter_start_timestamp
1/7/2022	10/01/2021	10/1/2021 12:00:00 AM
1/19/2022	10/01/2021	10/1/2021 12:00:00 AM
2/5/2022	10/01/2021	10/1/2021 12:00:00 AM
2/28/2022	10/01/2021	10/1/2021 12:00:00 AM
3/16/2022	10/01/2021	10/1/2021 12:00:00 AM
4/1/2022	01/01/2022	1/1/2022 12:00:00 AM
5/7/2022	01/01/2022	1/1/2022 12:00:00 AM
5/16/2022	01/01/2022	1/1/2022 12:00:00 AM
6/15/2022	01/01/2022	1/1/2022 12:00:00 AM
6/26/2022	01/01/2022	1/1/2022 12:00:00 AM
7/9/2022	04/01/2022	4/1/2021 12:00:00 AM
7/22/2022	04/01/2022	4/1/2021 12:00:00 AM
7/23/2022	04/01/2022	4/1/2021 12:00:00 AM
7/27/2022	04/01/2022	4/1/2021 12:00:00 AM
8/2/2022	04/01/2022	4/1/2021 12:00:00 AM

date	previous_quarter_start	previous_quarter_start_timestamp
8/8/2022	04/01/2022	4/1/2021 12:00:00 AM
8/19/2022	04/01/2022	4/1/2021 12:00:00 AM
9/26/2022	04/01/2022	4/1/2021 12:00:00 AM
10/14/2022	07/01/2022	7/1/2022 12:00:00 AM
10/29/2022	07/01/2022	7/1/2022 12:00:00 AM

In questo caso, poiché il valore `period_no` di -1 è stato utilizzato come argomento offset nella funzione `quarterstart()`, la funzione per prima cosa identifica il trimestre in cui avvengono le transazioni. Si sposta poi un trimestre prima e identifica il primo millisecondo di tale trimestre.

Schema della funzione `quarterstart()`, esempio di period_no



La transazione 8203 è avvenuta l'8 agosto. La funzione `quarterstart()` identifica che il trimestre precedente la transazione era compreso tra il 1° aprile e il 30 giugno. Quindi restituisce il primo millisecondo di quel trimestre, il 1° aprile alle 12:00:00.

Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, dobbiamo impostare il 1° marzo come inizio dell'anno finanziario.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        quarterstart(date,0,3) as start_of_quarter,
```

```
        timestamp(quarterstart(date,0,3)) as start_of_quarter_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `start_of_quarter`
- `start_of_quarter_timestamp`

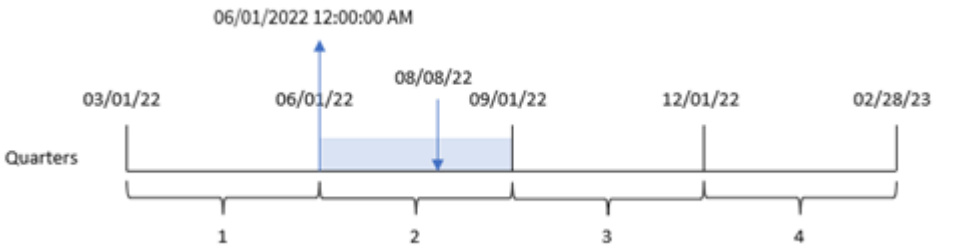
Tabella dei risultati

date	start_of_quarter	start_of_quarter_timestamp
1/7/2022	12/01/2021	12/1/2021 12:00:00 AM
1/19/2022	12/01/2021	12/1/2021 12:00:00 AM
2/5/2022	12/01/2021	12/1/2021 12:00:00 AM
2/28/2022	12/01/2021	12/1/2021 12:00:00 AM
3/16/2022	03/01/2022	3/1/2022 12:00:00 AM
4/1/2022	03/01/2022	3/1/2022 12:00:00 AM
5/7/2022	03/01/2022	3/1/2022 12:00:00 AM

date	start_of_quarter	start_of_quarter_timestamp
5/16/2022	03/01/2022	3/1/2022 12:00:00 AM
6/15/2022	06/01/2022	6/1/2022 12:00:00 AM
6/26/2022	06/01/2022	6/1/2022 12:00:00 AM
7/9/2022	06/01/2022	6/1/2022 12:00:00 AM
7/22/2022	06/01/2022	6/1/2022 12:00:00 AM
7/23/2022	06/01/2022	6/1/2022 12:00:00 AM
7/27/2022	06/01/2022	6/1/2022 12:00:00 AM
8/2/2022	06/01/2022	6/1/2022 12:00:00 AM
8/8/2022	06/01/2022	6/1/2022 12:00:00 AM
8/19/2022	06/01/2022	6/1/2022 12:00:00 AM
9/26/2022	09/01/2022	9/1/2022 12:00:00 AM
10/14/2022	09/01/2022	9/1/2022 12:00:00 AM
10/29/2022	09/01/2022	9/1/2022 12:00:00 AM

In questa istanza, poiché l'argomento `first_month_of_year` di 3 viene utilizzato nella funzione `quarterstart()`, l'inizio dell'anno si sposta dal 1° gennaio al 1° marzo.

Schema della funzione `quarterstart()`, esempio di first_month_of_year



La transazione 8203 è avvenuta l'8 agosto. Poiché l'inizio dell'anno è il 1° marzo, i trimestri dell'anno sono compresi tra marzo-maggio, giugno-agosto, settembre-novembre e dicembre-febbraio. La funzione `quarterstart()` identifica che la transazione ha avuto luogo nel trimestre compreso tra l'inizio di giugno e agosto e restituisce il primo millisecondo di tale trimestre, ovvero il 1° giugno alle 12:00:00.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce un timestamp per la fine del trimestre in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Aggiungere le seguenti misure:

- `=quarterstart(date)`
- `=timestamp(quarterstart(date))`

Tabella dei risultati

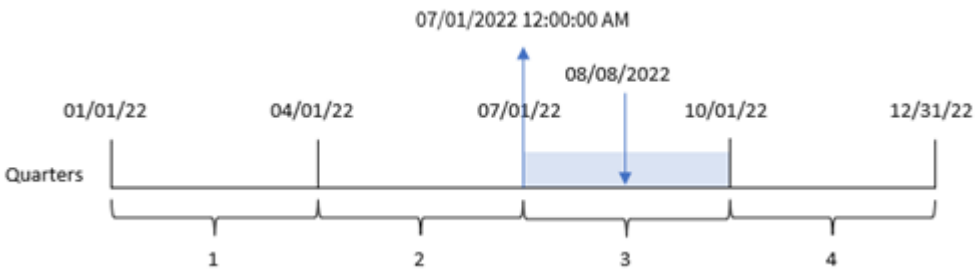
date	=quarterstart(date)	=timestamp(quarterstart(date))
10/14/2022	10/01/2022	10/1/2022 12:00:00 AM

date	=quarterstart(date)	=timestamp(quarterstart(date))
10/29/2022	10/01/2022	10/1/2022 12:00:00 AM
7/9/2022	07/01/2022	7/1/2022 12:00:00 AM
7/22/2022	07/01/2022	7/1/2022 12:00:00 AM
7/23/2022	07/01/2022	7/1/2022 12:00:00 AM
7/27/2022	07/01/2022	7/1/2022 12:00:00 AM
8/2/2022	07/01/2022	7/1/2022 12:00:00 AM
8/8/2022	07/01/2022	7/1/2022 12:00:00 AM
8/19/2022	07/01/2022	7/1/2022 12:00:00 AM
9/26/2022	07/01/2022	7/1/2022 12:00:00 AM
4/1/2022	04/01/2022	4/1/2022 12:00:00 AM
5/7/2022	04/01/2022	4/1/2022 12:00:00 AM
5/16/2022	04/01/2022	4/1/2022 12:00:00 AM
6/15/2022	04/01/2022	4/1/2022 12:00:00 AM
6/26/2022	04/01/2022	4/1/2022 12:00:00 AM
1/7/2022	01/01/2022	1/1/2022 12:00:00 AM
1/19/2022	01/01/2022	1/1/2022 12:00:00 AM
2/5/2022	01/01/2022	1/1/2022 12:00:00 AM
2/28/2022	01/01/2022	1/1/2022 12:00:00 AM
3/16/2022	01/01/2022	1/1/2022 12:00:00 AM

La misura `start_of_quarter` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `quarterstart()` e trasferendo il campo `date` come argomento della funzione.

La funzione `quarterstart()` identifica il trimestre in cui cade il valore della data, restituendo un timestamp per il primo millisecondo di quel trimestre.

Schema della funzione `quarterstart()`, esempio di oggetto grafico



La transazione 8203 è avvenuta l'8 agosto. La funzione `quarterstart()` identifica che la transazione è avvenuta nel terzo trimestre e restituisce il primo millisecondo del trimestre. Il valore restituito è il 1° luglio alle 12:00:00.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di saldi di prestiti, che viene caricato in una tabella chiamata `Loans`.
- Dati costituiti dagli ID dei prestiti, dal saldo all'inizio del trimestre e dal tasso di interesse semplice annuo applicato a ciascun prestito.

L'utente finale desidera un oggetto grafico che mostri, in base all'ID del prestito, gli interessi correnti che sono stati maturati per ciascun prestito nel trimestre in corso.

Script di caricamento

```
Loans:
Load
*
Inline
[
loan_id,start_balance,rate
8188,$10000.00,0.024
8189,$15000.00,0.057
8190,$17500.00,0.024
8191,$21000.00,0.034
8192,$90000.00,0.084
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:
   - `loan_id`
   - `start_balance`
2. Quindi, creare questa misura per calcolare l'interesse accumulato:
   `=start_balance*(rate*(today(1)-quarterstart(today(1)))/365)`
3. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

loan_id	start_balance	=start_balance*(rate*(today(1)-quarterstart(today(1)))/365)
8188	$10000.00	$15.07
8189	$15000.00	$128.84
8190	$17500.00	$63.29
8191	$21000.00	$107.59
8192	$90000.00	$1139.18

Utilizzando la data odierna come unico argomento, la funzione `quarterstart()` restituisce la data di inizio dell'anno corrente. Sottraendo tale risultato dalla data corrente, l'espressione restituisce il numero di giorni trascorsi fino ad ora questo trimestre.

Questo valore viene quindi moltiplicato per il tasso di interesse e diviso per 365, per restituire il tasso di interesse effettivo incorso per questo periodo. Il risultato viene poi moltiplicato per il saldo iniziale del prestito per ottenere gli interessi maturati fino a questo momento del trimestre.

second

Questa funzione restituisce un numero intero che rappresenta il secondo in cui la frazione di **expression** viene interpretata come ora in base all'interpretazione numerica standard.

Sintassi:

```
second (expression)
```

Tipo di dati restituiti: numero intero

Casi di utilizzo

La funzione `second()` è utile quando si desidera confrontare le aggregazioni per secondo. Ad esempio, la funzione può essere utilizzata se si desidera visualizzare la distribuzione del conteggio delle attività per secondo.

Queste dimensioni possono essere sia create nello script di caricamento, utilizzando la funzione per creare un campo in una tabella del Calendario principale, sia utilizzate direttamente in un grafico come dimensione calcolata.

Esempi di funzioni

Esempio	Risultato
`second( '09:14:36' )`	restituisce 36
`second( '0.5555' )`	restituisce 55 (poiché 0.5555 = 13:19:55)

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Variabile

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente transazioni per timestamp, caricato in una tabella chiamata `Transactions`.
- Viene utilizzata la variabile di sistema `TimeStamp` predefinita (`M/D/YYYY h:mm:ss[.fff] TT`).
- La creazione di un campo, `second`, per calcolare quando sono avvenuti gli acquisti.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
    Load
        *,
        second(date) as second
        ;
Load
*
Inline
[
id,date,amount
9497,'01/05/2022 7:04:57 PM',47.25
9498,'01/03/2022 2:21:53 PM',51.75
9499,'01/03/2022 5:40:49 AM',73.53
9500,'01/04/2022 6:49:38 PM',15.35
9501,'01/01/2022 10:10:22 PM',31.43
9502,'01/05/2022 7:34:46 PM',13.24
9503,'01/06/2022 10:58:34 PM',74.34
9504,'01/06/2022 11:29:38 AM',50.00
```

```
9505,'01/02/2022 8:35:54 AM',36.34
9506,'01/06/2022 8:49:09 AM',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `second`

Tabella dei risultati

date	secondo
01/01/2022 10:10:22 PM	22
01/02/2022 8:35:54 AM	54
01/03/2022 5:40:49 AM	49
01/03/2022 2:21:53 PM	53
01/04/2022 6:49:38 PM	38
01/05/2022 7:04:57 PM	57
01/05/2022 7:34:46 PM	46
01/06/2022 8:49:09 AM	9
01/06/2022 11:29:38 AM	38
01/06/2022 10:58:34 PM	34

I valori nel campo `second` ora sono creati usando la funzione `second()` e trasferendo la data come espressione nell'istruzione LOAD precedente.

Esempio 2 - Oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. I valori `second` sono calcolati mediante una misura in un oggetto grafico.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

Transactions:
```

```
Load
*
Inline
[
id,date,amount
9497,'01/05/2022 7:04:57 PM',47.25
9498,'01/03/2022 2:21:53 PM',51.75
9499,'01/03/2022 5:40:49 AM',73.53
9500,'01/04/2022 6:49:38 PM',15.35
9501,'01/01/2022 10:10:22 PM',31.43
9502,'01/05/2022 7:34:46 PM',13.24
9503,'01/06/2022 10:58:34 PM',74.34
9504,'01/06/2022 11:29:38 AM',50.00
9505,'01/02/2022 8:35:54 AM',36.34
9506,'01/06/2022 8:49:09 AM',74.23
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:`date`.

Creare la seguente misura:

```
=second(date)
```

Tabella dei risultati

date	=second(date)
01/01/2022 10:10:22 PM	22
01/02/2022 8:35:54 AM	54
01/03/2022 5:40:49 AM	49
01/03/2022 2:21:53 PM	53
01/04/2022 6:49:38 PM	38
01/05/2022 7:04:57 PM	57
01/05/2022 7:34:46 PM	46
01/06/2022 8:49:09 AM	9
01/06/2022 11:29:38 AM	38
01/06/2022 10:58:34 PM	34

I valori per '`second`' sono creati usando la funzione `second()` e trasferendo la data come espressione in una misura per l'oggetto grafico.

Esempio 3 – Scenario

Script di caricamento ed espressioni del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un insieme di timestamp, generato per rappresentare il traffico verso il sito web di vendita dei biglietti di un particolare festival. Questi timestamp e un `id` corrispondente vengono caricati in una tabella chiamata `Web_Traffic`.
- Viene utilizzata la variabile di sistema `TimeStamp M/D/YYYY h:mm:ss[.fff] TT`.

In questo scenario, erano presenti 10.000 biglietti, messi in vendita alle 9:00 del 20 maggio 2021. Un minuto dopo, i biglietti erano esauriti.

L'utente desidera un oggetto grafico che mostri, al secondo, il numero di visite al sito web.

Script di caricamento

```
SET TimestampFormat='M/D/YYYY h:mm:ss[.fff] TT';

tmpTimeStampCreator:
load
    makedate(2022,05,20) as date
AutoGenerate 1;

join load
    maketime(9+floor(rand()*2),0,floor(rand()*59)) as time
autogenerate 10000;

Web_Traffic:
load
    recno() as id,
    timestamp(date + time) as timestamp
resident tmpTimeStampCreator;

drop table tmpTimeStampCreator;
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Quindi, creare una dimensione calcolata utilizzando la seguente espressione:
   `=second(timestamp)`
3. Creare una misura di aggregazione per calcolare il conteggio totale delle voci:
   `=count(id)`

La tabella dei risultati sarà simile alla tabella seguente, ma con valori diversi per la misura di aggregazione:

Tabella dei risultati

second(timestamp)	=count(id)
0	150
1	184
2	163
3	178
4	179
5	158
6	177
7	169
8	149
9	186
10	169
11	179
12	186
13	182
14	180
15	153
16	191
17	203
18	158
19	159
20	163
+39 ulteriori righe	

setdateyear

Questa funzione utilizza come input un **timestamp** e un **year** e aggiorna il **timestamp** con l'**year** specificato nell'input.

Sintassi:

```
setdateyear (timestamp, year)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
timestamp	Un indicatore temporale standard di Qlik Sense (spesso solo una data).
year	Un anno a quattro cifre.

Esempi e risultati:
In questi esempi viene utilizzato il formato della data **DD/MM/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di script

Esempio	Risultato
`setdateyear ('29/10/2005', 2013)`	Restituisce '29/10/2013
`setdateyear ('29/10/2005 04:26:14', 2013)`	Restituisce '29/10/2013 04:26:14' Per visualizzare la sezione relativa alla data e all'ora dell'indicatore temporale in una visualizzazione, è necessario impostare la formattazione dei numeri su Data e scegliere un valore per Formattazione che consenta di mostrare i valori della data e dell'ora.

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
SetYear:
Load *,
SetDateYear(testdates, 2013) as NewYear
Inline [
testdates
1/11/2012
10/12/2012
1/5/2013
2/1/2013
19/5/2013
15/9/2013
11/12/2013
2/3/2014
14/5/2014
13/6/2014
7/7/2014
4/8/2014
];
```

La tabella risultante contiene le date originali e una colonna in cui l'anno è stato impostato su 2013.

Tabella dei risultati

testdates	NewYear
1/11/2012	1/11/2013
10/12/2012	10/12/2013
2/1/2012	2/1/2013
1/5/2013	1/5/2013
19/5/2013	19/5/2013
15/9/2013	15/9/2013
11/12/2013	11/12/2013
2/3/2014	2/3/2013
14/5/2014	14/5/2013
13/6/2014	13/6/2013
7/7/2014	7/7/2013
4/8/2014	4/8/2013

setdateyearmonth

Questa funzione utilizza come input un **timestamp**, un **month** e un **year** e aggiorna il **timestamp** con l'**year** e il **month** specificati nell'input. .

Sintassi:

```
SetDateYearMonth (timestamp, year, month)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
timestamp	Un indicatore temporale standard di Qlik Sense (spesso solo una data).
year	Un anno a quattro cifre.
month	Un mese a una o due cifre.

Esempi e risultati:
In questi esempi viene utilizzato il formato della data **DD/MM/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati. Modificare il formato negli esempi in base alle proprie necessità.

Esempi di script

Esempio	Risultato
`setdateyearmonth ('29/10/2005', 2013, 3)`	Restituisce '29/03/2013
`setdateyearmonth ('29/10/2005 04:26:14', 2013, 3)`	Restituisce '29/03/2013 04:26:14' Per visualizzare la sezione relativa alla data e all'ora dell'indicatore temporale in una visualizzazione, è necessario impostare la formattazione dei numeri su Data e scegliere un valore per Formattazione che consenta di mostrare i valori della data e dell'ora.

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
SetYearMonth:
Load *,
SetDateYearMonth(testdates, 2013,3) as NewYearMonth
Inline [
testdates
1/11/2012
10/12/2012
2/1/2013
19/5/2013
15/9/2013
11/12/2013
14/5/2014
13/6/2014
7/7/2014
4/8/2014
];
```

La tabella risultante contiene le date originali e una colonna in cui l'anno è stato impostato su 2013.

Tabella dei risultati

testdates	NewYearMonth
1/11/2012	1/3/2013
10/12/2012	10/3/2013
2/1/2012	2/3/2013
19/5/2013	19/3/2013

testdates	NewYearMonth
15/9/2013	15/3/2013
11/12/2013	11/3/2013
14/5/2014	14/3/2013
13/6/2014	13/3/2013
7/7/2014	7/3/2013
4/8/2014	4/3/2013

timezone

Questa funzione restituisce il fuso orario impostato sul computer dove è in esecuzione il motore Qlik.

Sintassi:

```
TimeZone( )
```

Tipo di dati restituiti: duale

Esempio:

`timezone( )`
Se si desidera visualizzare un fuso orario differente per una misura della propria app, è possibile usare la funzione `localtime()` in una misura.

today

Questa funzione restituisce la data attuale. La funzione restituisce i valori nel formato della variabile di sistema `DateFormat`.

Sintassi:

```
today([ timer_mode])
```

Tipo di dati restituiti: duale

La funzione `today()` può essere utilizzata sia nello script di caricamento che negli oggetti del grafico.

Il valore predefinito `timer_mode` è 1.

Argomenti

Argomento	Descrizione
timer_mode	Può presentare i valori seguenti: 0 (giorno dell'ultimo caricamento di dati completato) 1 (giorno della chiamata della funzione) 2 (giorno di apertura dell'app) *Se si utilizza la funzione in uno script di caricamento, **timer_mode=0** restituirà il giorno di completamento dell'ultimo caricamento dei dati, mentre **timer_mode=1** restituirà il giorno del caricamento dei dati attuale.*

Esempi di funzioni

valore timer_mode	Risultato se utilizzato nello script di caricamento	Risultato se utilizzato nell'oggetto grafico
0	Restituisce una data, nel formato della variabile di sistema `DateFormat`, dell'ultimo ricaricamento dei dati riuscito prima dell'ultimo ricaricamento dei dati.	Restituisce una data, nel formato della variabile di sistema `DateFormat`, per l'ultimo ricaricamento dei dati.
1	Restituisce una data, nel formato della variabile di sistema `DateFormat`, per l'ultimo ricaricamento dei dati.	Restituisce una data, nel formato della variabile di sistema `DateFormat`, della chiamata alla funzione.
2	Restituisce una data, nel formato della variabile di sistema `DateFormat`, per l'inizio della sessione dell'utente nell'applicazione. Non verrà aggiornato a meno che l'utente non ricarichi lo script.	Restituisce la data, nel formato della variabile di sistema `DateFormat`, per l'inizio della sessione dell'utente nell'applicazione. Questo valore verrà aggiornato all'inizio di una nuova sessione o quando i dati dell'applicazione verranno ricaricati.

Casi di utilizzo

La funzione `today()` viene comunemente utilizzata come componente di un'espressione. Ad esempio, può essere utilizzata per calcolare gli interessi accumulati in un mese fino alla data corrente.

La tabella seguente fornisce una spiegazione del risultato restituito dalla funzione `today()`, in presenza di diversi valori dell'argomento `timer_mode`:

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema,

a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Generazione di oggetti tramite script di caricamento

Script di caricamento e risultati

Panoramica

L'esempio seguente crea tre variabili utilizzando la funzione `today()`. Ogni variabile utilizza una delle opzioni `timer_mode` per dimostrarne l'effetto.

Affinché le variabili dimostrino il loro scopo, ricaricare lo script e poi, dopo 24 ore, ricaricare lo script una seconda volta. In questo modo le variabili `today(0)` e `today(1)` mostreranno valori diversi, dimostrando così correttamente il loro scopo.

Script di caricamento

```
LET vPreviousDataLoad = today(0);
LET vCurrentDataLoad = today(1);
LET vApplicationOpened = today(2);
```

Risultati

Una volta che i dati vengono caricati una seconda volta, creare tre caselle di testo seguendo le indicazioni riportate di seguito.

Per prima cosa, creare una casella di testo per i dati che sono stati caricati precedentemente.

Procedere come indicato di seguito:

1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la seguente misura all'oggetto:
   `=vPreviousDataLoad`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Ora di caricamento precedente' all'oggetto.

Quindi, creare una casella di testo per i dati che si stanno caricando.

Procedere come indicato di seguito:

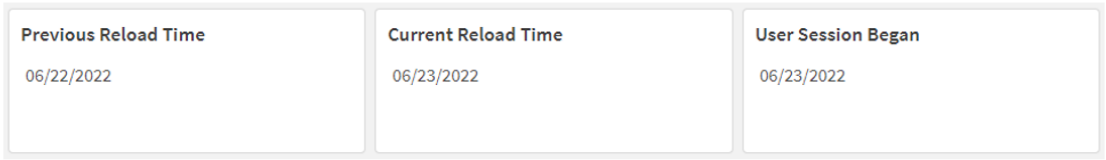
1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la seguente misura all'oggetto:
   `=vCurrentDataLoad`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Ora di caricamento attuale' all'oggetto.

Creare una casella di testo finale da mostrare quando è stata avviata la sessione dell'utente nell'applicazione.

Procedere come indicato di seguito:

1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la seguente misura all'oggetto:
   `=vApplicationOpened`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Inizio sessione utente' all'oggetto.

Schema delle variabili create con la funzione `today()` nello script di caricamento



L'immagine qui sopra mostra dei valori di esempio per ciascuna delle variabili create. Ad esempio, i valori potrebbero essere i seguenti:

- Ora di ricaricamento precedente: 22/06/2022
- Ora di ricaricamento attuale: 23/06/2022
- Inizio della sessione utente: 23/06/2022

Esempio 2 – Generazione di oggetti senza script di caricamento

Script di caricamento ed espressione del grafico

Panoramica

L'esempio seguente crea tre oggetti grafico utilizzando la funzione `today()`. Ogni oggetto del grafico utilizza una delle opzioni `timer_mode` per dimostrarne l'effetto.

Per questo esempio non esiste uno script di caricamento.

Risultati

Una volta che i dati vengono caricati una seconda volta, creare tre caselle di testo.

Per prima cosa, creare una casella di testo per l'ultimo ricaricamento di dati.

Procedere come indicato di seguito:

1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la misura seguente:
   `=today(0)`
3. Sotto **Aspetto**, selezionare **Show titles** e aggiungere all'oggetto il titolo 'Ricaricamento ultimi dati'.

Quindi, creare una casella di testo per mostrare l'ora corrente.

Procedere come indicato di seguito:

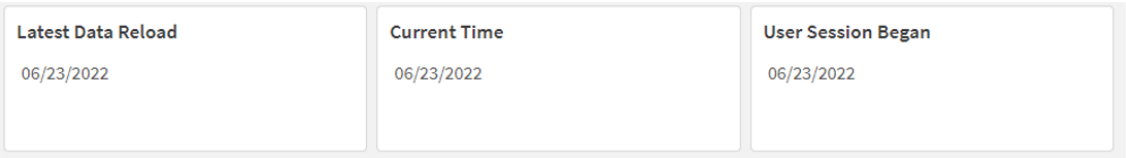
1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la misura seguente:
   `=today(1)`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Ora attuale' all'oggetto.

Creare una casella di testo finale da mostrare quando è stata avviata la sessione dell'utente nell'applicazione.

Procedere come indicato di seguito:

1. Utilizzando l'oggetto grafico **Testo e immagine**, creare una casella di testo.
2. Aggiungere la misura seguente:
   `=today(2)`
3. In **Aspetto**, selezionare **Show titles** e aggiungere il titolo 'Sessione utente iniziata' all'oggetto.

Schema degli oggetti creato usando la funzione `today()` senza script di caricamento



L'immagine qui sopra mostra dei valori di esempio per ciascuno degli oggetti creati. Ad esempio, i valori potrebbero essere i seguenti:

- Data ultimo ricaricamento: 23/06/2022
- Orario corrente: 23/06/2022
- Inizio della sessione utente: 23/06/2022

L'oggetto grafico 'Ultimo caricamento dati' utilizza un valore `timer_mode` pari a 0. Restituisce il timestamp dell'ultima volta in cui i dati sono stati ricaricati correttamente.

L'oggetto grafico 'Ora corrente' utilizza un valore `timer_mode` di 1. Questo restituisce l'ora corrente secondo l'orologio di sistema. Se il foglio o l'oggetto viene ricaricato, questo valore verrà aggiornato.

L'oggetto grafico 'Sessione utente iniziata' utilizza un valore `timer_mode` di 2. Restituisce il timestamp di quando l'applicazione è stata aperta e la sessione dell'utente è iniziata.

Esempio 3 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di saldi di prestiti, caricato in una tabella denominata `Loans`.
- Dati della tabella con campi per ID prestito, saldo all'inizio del mese e tasso di interesse semplice annuo applicato a ciascun prestito.

L'utente finale desidera un oggetto grafico che mostri, in base all'ID del prestito, gli interessi correnti che sono stati maturati per ciascun prestito nel mese in corso. Anche se l'applicazione viene ricaricata solo una volta alla settimana, l'utente vorrebbe che i risultati venissero aggiornati ogni volta che l'oggetto o l'applicazione vengono aggiornati.

Script di caricamento

```
Loans:
Load
*
Inline
[
loan_id,start_balance,rate
8188,$10000.00,0.024
8189,$15000.00,0.057
8190,$17500.00,0.024
8191,$21000.00,0.034
8192,$90000.00,0.084
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Aggiungere i seguenti campi come dimensioni:
    - `loan_id`
    - `start_balance`
3. Quindi, creare una misura per calcolare gli interessi accumulati:

```
=start_balance*(rate*(today(1)-monthstart(today(1)))/365)
```

4. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

loan_id	start_balance	=start_balance*(rate*(today(1)-monthstart(today(1)))/365)
8188	$10000.00	$16.44
8189	$15000.00	$58.56
8190	$17500.00	$28.77
8191	$21000.00	$48.90
8192	$90000.00	$517.81

Utilizzando la funzione `today()` per restituire la data odierna come unico argomento, la funzione `monthstart()` restituisce la data di inizio del mese corrente. Sottraendo tale risultato dalla data corrente, di nuovo usando la funzione `today()`, l'espressione restituisce il numero di giorni trascorsi fino a ora questo mese.

Questo valore viene quindi moltiplicato per il tasso di interesse e diviso per 365, per restituire il tasso di interesse effettivo incorso per questo periodo. Il risultato viene poi moltiplicato per il saldo iniziale del prestito per ottenere gli interessi maturati fino a ora in questo mese.

Poiché il valore 1 è usato come argomento `timer_mode` nelle funzioni `today()` all'interno dell'espressione, ogni volta che l'oggetto grafico viene aggiornato (aprendo l'applicazione, aggiornando la pagina, passando da un foglio all'altro, ecc.), la data restituita sarà quella corrente e i risultati saranno aggiornati di conseguenza.

UTC

Restituisce il Coordinated Universal Time attuale.

Sintassi:

```
UTC( )
```

Tipo di dati restituiti: duale

Esempio:

```
utc( )
```

week

Questa funzione restituisce un numero intero che rappresenta il numero della settimana corrispondente alla data inserita.

Sintassi:

```
week(timestamp [, first_week_day [, broken_weeks [, reference_day]]])
```

Tipo di dati restituiti: numero intero

Argomenti

Argomento	Descrizione
timestamp	La data o la data e ora da valutare.
first_week_day	Specifica il giorno di inizio della settimana. Se omesso, viene utilizzato il valore della variabile **FirstWeekDay**. I valori possibili per **first_week_day** sono 0 per lunedì, 1 per martedì, 2 per mercoledì, 3 per giovedì, 4 per venerdì, 5 per sabato e 6 per domenica. Per maggiori informazioni sulle variabili di sistema, vedere *FirstWeekDay (page 224)*.
broken_weeks	Se non si specifica **broken_weeks**, il valore della variabile **BrokenWeeks** verrà utilizzato per definire se le settimane sono parziali o meno.
reference_day	Se non si specifica **reference_day**, il valore della variabile **ReferenceDay** verrà utilizzato per definire quale giorno di gennaio impostare come giorno di riferimento per definire la settimana 1. Per impostazione predefinita, le funzioni Qlik Sense utilizzano 4 come giorno di riferimento. Questo significa che la settimana 1 deve contenere il 4 gennaio, vale a dire che la settimana 1 deve sempre contenere 4 quattro giorni di gennaio.

La funzione `week()` determina la settimana in cui cade la data e restituisce il numero della settimana.

In Qlik Sense, le impostazioni regionali vengono recuperate alla creazione dell'app e le impostazioni corrispondenti vengono memorizzate nello script come variabili d'ambiente. Queste vengono utilizzate per determinare il numero della settimana.

Ciò significa che la maggior parte degli sviluppatori di applicazioni europei ottiene le seguenti variabili d'ambiente, corrispondenti alla definizione ISO 8601:

```
Set FirstWeekDay =0;    // Monday as first week day
Set BrokenWeeks  =0;    // Use unbroken weeks
Set ReferenceDay =4;    // Jan 4th is always in week 1
```

Uno sviluppatore di app nordamericano ottiene spesso le seguenti variabili d'ambiente:

```
Set FirstWeekDay =6;    // Sunday as first week day
Set BrokenWeeks  =1;    // Use broken weeks
Set ReferenceDay =1;    // Jan 1st is always in week 1
```

Il primo giorno della settimana è determinato dalla variabile di sistema `FirstWeekDay`. È possibile anche modificare il primo giorno della settimana utilizzando l'argomento `first_week_day` nella funzione `week()`.

Se l'applicazione utilizza settimane interrotte, il conteggio del numero di settimane inizia il 1° gennaio e termina il giorno precedente alla variabile di sistema `FirstWeekDay`, indipendentemente dal numero di giorni trascorsi.

Se l'applicazione utilizza settimane ininterrotte, la settimana 1 può iniziare nell'anno precedente o nei primi giorni di gennaio. Dipende da come si utilizzano le variabili di ambiente `FirstWeekDay` e `ReferenceDay`.

Casi di utilizzo

La funzione `The week()` è utile per confrontare le aggregazioni per settimane. Ad esempio, potrebbe essere usata se si desidera visualizzare le vendite totali dei prodotti in base alla settimana. La funzione `week()` viene preferita a `weekname()` quando l'utente desidera che il calcolo non utilizzi necessariamente le variabili di sistema `BrokenWeeks`, `FirstWeekDay` o `ReferenceDay` dell'applicazione.

Ad esempio, può essere usata se si desidera visualizzare le vendite totali dei prodotti in base alla settimana.

Se l'applicazione utilizza settimane ininterrotte, la settimana 1 può contenere date di dicembre dell'anno precedente o escludere date di gennaio dell'anno in corso. Se l'applicazione utilizza settimane interrotte, la settimana 1 può contenere meno di sette giorni.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Gli esempi riportati di seguito presuppongono

```
Set DateFormat= 'MM/DD/YYYY';
Set FirstWeekDay=0;
Set BrokenWeeks=0;
Set ReferenceDay=4;
```

Esempi di funzioni

Esempio	Risultato
`week('12/28/2021')`	Restituisce 52.
`week(44614)`	Restituisce 8, poiché questo è il numero di serie del 22/02/2022.
`week('01/03/2021')`	Restituisce 53.
`week('01/03/2021',6)`	Restituisce 1.

Esempio 1 - Variabili di sistema predefinite

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per l'ultima settimana del 2021 e le prime due settimane del 2022 caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `week_number`, che restituisce l'anno e il numero di settimana in cui sono avvenute le transazioni.
- La creazione di un campo chiamato `week_day`, che mostra il valore del giorno della settimana per ogni data di transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=6;
SET BrokenWeeks=1;
SET ReferenceDay=0;

Transactions:
    Load
        *,
        WeekDay(date) as week_day,
        Week(date) as week_number
        ;
Load
*
Inline
[
id,date,amount
8183,12/27/2021,58.27
8184,12/28/2021,67.42
8185,12/29/2021,23.80
8186,12/30/2021,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
8198,01/11/2022,18.37
```

```
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `week_day`
- `week_number`

Tabella dei risultati

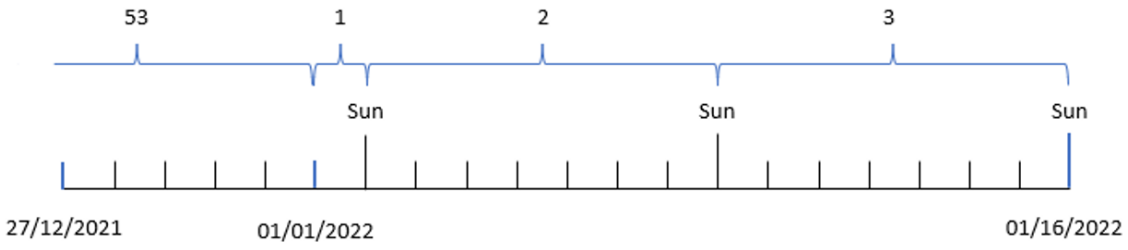
id	data	week_day	week_number
8183	12/27/2021	Mon	53
8184	12/28/2021	Tue	53
8185	12/29/2021	Wed	53
8186	12/30/2021	Thu	53
8187	12/31/2021	Fri	53
8188	01/01/2022	Sat	1
8189	01/02/2022	Sun	2
8190	01/03/2022	Mon	2
8191	01/04/2022	Tue	2
8192	01/05/2022	Wed	2
8193	01/06/2022	Thu	2
8194	01/07/2022	Fri	2
8195	01/08/2022	Sat	2
8196	01/09/2022	Sun	3
8197	01/10/2022	Mon	3
8198	01/11/2022	Tue	3
8199	01/12/2022	Wed	3
8200	01/13/2022	Thu	3
8201	01/14/2022	Fri	3

Il campo `week_number` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `week()` e trasferendo il campo `date` come argomento della funzione.

Non vengono passati altri parametri alla funzione e quindi sono attive le seguenti variabili predefinite che influenzano la funzione `week()`:

- `BrokenWeeks`: Il conteggio delle settimane inizia il 1° gennaio
- `FirstWeekDay`: Il primo giorno della settimana è la domenica

Schema della funzione `week()`, utilizzando le variabili di sistema predefinite



Poiché l'applicazione utilizza la variabile di sistema predefinita `BrokenWeeks`, la settimana 1 inizia il 1° gennaio, un sabato.

A causa della variabile di sistema `FirstWeekDay` predefinita, le settimane iniziano di domenica. La prima domenica dopo il 1° gennaio è il 2 gennaio, quando inizia la seconda settimana.

Esempio 2 – first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- La creazione di un campo, `week_number`, che restituisce l'anno e il numero di settimana in cui sono avvenute le transazioni.
- La creazione di un campo chiamato `week_day`, che mostra il valore del giorno della settimana per ogni data di transazione.

In questo esempio, vogliamo impostare l'inizio della settimana lavorativa al martedì.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=6;
SET BrokenWeeks=1;
SET ReferenceDay=0;

Transactions:
    Load
        *,
```

```
        WeekDay(date) as week_day,
        Week(date,1) as week_number
        ;
Load
*
Inline
[
id,date,amount
8183,12/27/2022,58.27
8184,12/28/2022,67.42
8185,12/29/2022,23.80
8186,12/30/2022,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
8198,01/11/2022,18.37
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `week_day`
- `week_number`

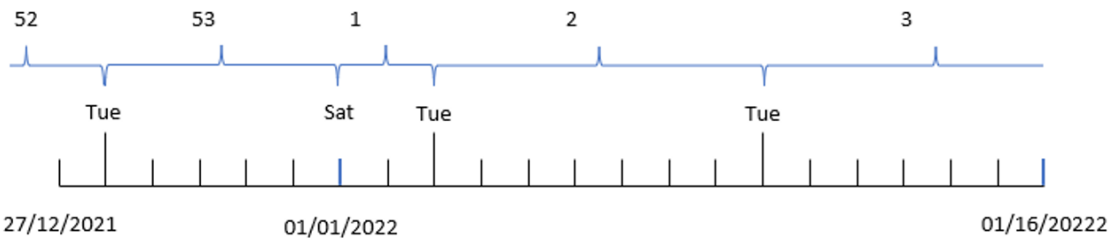
Tabella dei risultati

id	data	week_day	week_number
8183	12/27/2021	Mon	52
8184	12/28/2021	Tue	53
8185	12/29/2021	Wed	53
8186	12/30/2021	Thu	53
8187	12/31/2021	Fri	53
8188	01/01/2022	Sat	1

id	data	week_day	week_number
8189	01/02/2022	Sun	1
8190	01/03/2022	Mon	1
8191	01/04/2022	Tue	2
8192	01/05/2022	Wed	2
8193	01/06/2022	Thu	2
8194	01/07/2022	Fri	2
8195	01/08/2022	Sat	2
8196	01/09/2022	Sun	2
8197	01/10/2022	Mon	2
8198	01/11/2022	Tue	3
8199	01/12/2022	Wed	3
8200	01/13/2022	Thu	3
8201	01/14/2022	Fri	3

L'applicazione utilizza ancora le settimane parziali. Tuttavia, l'argomento `first_week_day` è stato impostato a 1 nella funzione `week()`. In questo modo si imposta il primo giorno della settimana come martedì.

Schema della funzione `week()`, esempio first_week_day



L'applicazione utilizza la variabile di sistema predefinita `BrokenWeeks`, la settimana 1 inizia il 1° gennaio, un sabato.

L'argomento `first_week_day` della funzione `week()` imposta il primo giorno della settimana a un martedì. Pertanto, la settimana 53 inizia il 28 dicembre 2021.

Tuttavia, poiché la funzione utilizza ancora le settimane parziali, la settimana 1 sarà di soli due giorni, poiché il primo martedì dopo il 1° gennaio è il 3 gennaio.

Esempio 3 – unbroken_weeks

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio.

In questo esempio, utilizziamo settimane intere.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=6;
SET BrokenWeeks=1;
SET ReferenceDay=0;

Transactions:
    Load
        *,
        WeekDay(date) as week_day,
        Week(date,6,0) as week_number
        ;
Load
*
Inline
[
id,date,amount
8183,12/27/2022,58.27
8184,12/28/2022,67.42
8185,12/29/2022,23.80
8186,12/30/2022,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
8198,01/11/2022,18.37
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `week_day`
- `week_number`

Schema della funzione `week()`, esempio di oggetto grafico

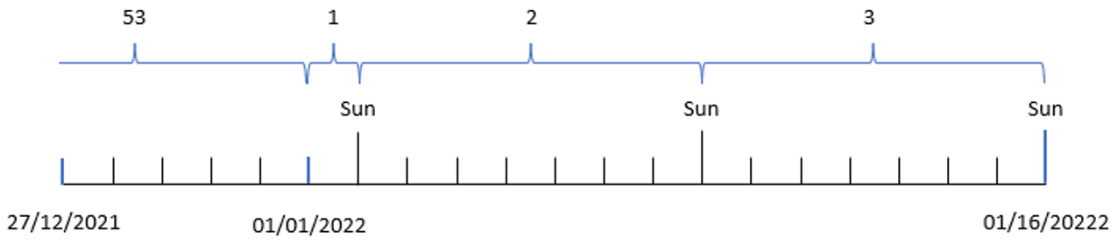


Tabella dei risultati

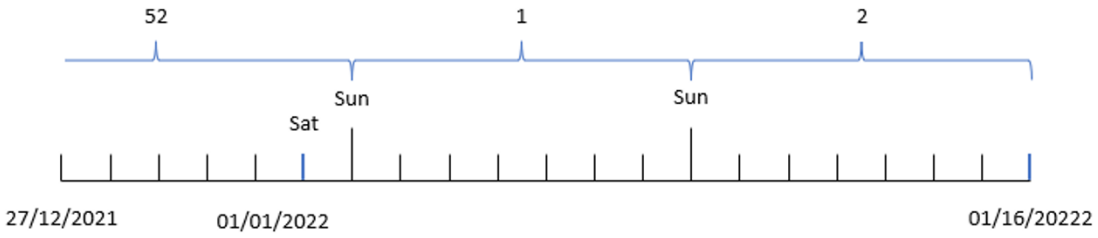
id	data	week_day	week_number
8183	12/27/2021	Mon	52
8184	12/28/2021	Tue	52
8185	12/29/2021	Wed	52
8186	12/30/2021	Thu	52
8187	12/31/2021	Fri	52
8188	01/01/2022	Sat	52
8189	01/02/2022	Sun	1
8190	01/03/2022	Mon	1
8191	01/04/2022	Tue	1
8192	01/05/2022	Wed	1
8193	01/06/2022	Thu	1
8194	01/07/2022	Fri	1
8195	01/08/2022	Sat	1
8196	01/09/2022	Sun	2
8197	01/10/2022	Mon	2
8198	01/11/2022	Tue	2

id	data	week_day	week_number
8199	01/12/2022	Wed	2
8200	01/13/2022	Thu	2
8201	01/14/2022	Fri	2

Il parametro first_week_date è impostato su 1, rendendo il martedì il primo giorno della settimana. Il parametro broken_weeks viene impostato a 0, forzando la funzione a utilizzare settimane intere. Infine, il terzo parametro imposta il reference_day su 2.

Il parametro `first_week_date` è impostato su 6, rendendo la domenica il primo giorno della settimana. Il parametro `broken_weeks` è impostato su 0, obbligando la funzione a utilizzare settimane intere.

Schema della funzione week(), esempio di utilizzo di settimane intere



Utilizzando le settimane intere, la settimana 1 non inizia necessariamente il 1° gennaio, ma deve avere un minimo di quattro giorni. Pertanto, nel set di dati, la settimana 52 si conclude sabato 1° gennaio 2022. La settimana 1 inizia con la variabile di sistema `FirstWeekDay`, domenica 2 gennaio. Questa settimana si concluderà il sabato successivo, l'8 gennaio.

Esempio 4 – reference_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del terzo esempio.
- La creazione di un campo, `week_number`, che restituisce l'anno e il numero di settimana in cui sono avvenute le transazioni.
- La creazione di un campo chiamato `week_day`, che mostra il valore del giorno della settimana per ogni data di transazione.

Inoltre, devono essere soddisfatte le seguenti condizioni:

- La settimana lavorativa inizia di martedì.
- L'azienda utilizza settimane intere.
- Il valore `reference_day` è 2. In altre parole, il numero minimo di giorni a gennaio nella settimana 1 sarà 2.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=6;
SET BrokenWeeks=1;
SET ReferenceDay=0;

Transactions:
    Load
        *,
        WeekDay(date) as week_day,
        Week(date,1,0,2) as week_number
        ;
Load
*
Inline
[
id,date,amount
8183,12/27/2022,58.27
8184,12/28/2022,67.42
8185,12/29/2022,23.80
8186,12/30/2022,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
8198,01/11/2022,18.37
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`

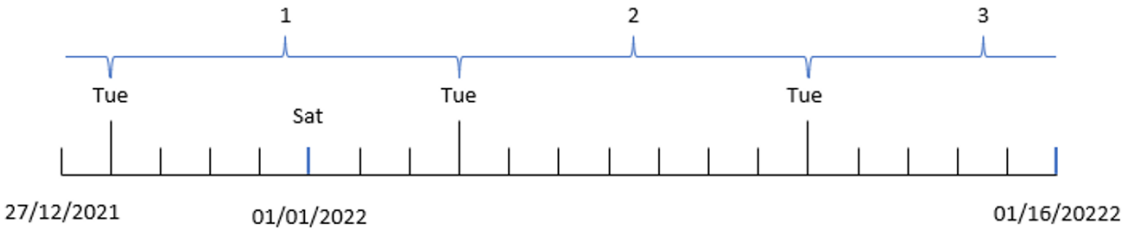
- week_day
- week_number

Tabella dei risultati

id	data	week_day	week_number
8183	12/27/2021	Mon	52
8184	12/28/2021	Tue	1
8185	12/29/2021	Wed	1
8186	12/30/2021	Thu	1
8187	12/31/2021	Fri	1
8188	01/01/2022	Sat	1
8189	01/02/2022	Sun	1
8190	01/03/2022	Mon	1
8191	01/04/2022	Tue	2
8192	01/05/2022	Wed	2
8193	01/06/2022	Thu	2
8194	01/07/2022	Fri	2
8195	01/08/2022	Sat	2
8196	01/09/2022	Sun	2
8197	01/10/2022	Mon	2
8198	01/11/2022	Tue	3
8199	01/12/2022	Wed	3
8200	01/13/2022	Thu	3
8201	01/14/2022	Fri	3

Il parametro `first_week_date` è impostato su 1, rendendo il martedì il primo giorno della settimana. Il parametro `broken_weeks` è impostato su 0, obbligando la funzione a utilizzare settimane intere. Infine, il terzo parametro imposta il parametro `reference_day` a 2.

Schema della funzione week(), esempio reference_day



Con la funzione che utilizza settimane intere e un valore `reference_day` di 2 come parametro, la settimana 1 deve includere solo due giorni di gennaio. Poiché il primo giorno feriale è il martedì, la settimana 1 inizia il 28 dicembre 2021 e si conclude lunedì 3 gennaio 2022.

Esempio 5 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce il numero della settimana viene creato come misura in un oggetto grafico.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8183,12/27/2022,58.27
8184,12/28/2022,67.42
8185,12/29/2022,23.80
8186,12/30/2022,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
```

```
8198,01/11/2022,18.37
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Aggiungere i seguenti campi come dimensioni:
   - `id`
   - `date`
3. Quindi, creare la seguente misura:
   `=week (date)`
4. Creare una misura , `week_day` che mostri il valore del giorno della settimana per ogni data di transazione:
   `=weekday(date)`

Tabella dei risultati

id	date	=week(date)	=weekday(date)
8183	12/27/2021	53	Mon
8184	12/28/2021	53	Tue
8185	12/29/2021	53	Wed
8186	12/30/2021	53	Thu
8187	12/31/2021	53	Fri
8188	01/01/2022	1	Sat
8189	01/02/2022	2	Sun
8190	01/03/2022	2	Mon
8191	01/04/2022	2	Tue
8192	01/05/2022	2	Wed
8193	01/06/2022	2	Thu
8194	01/07/2022	2	Fri
8195	01/08/2022	2	Sat
8196	01/09/2022	3	Sun
8197	01/10/2022	3	Mon
8198	01/11/2022	3	Tue

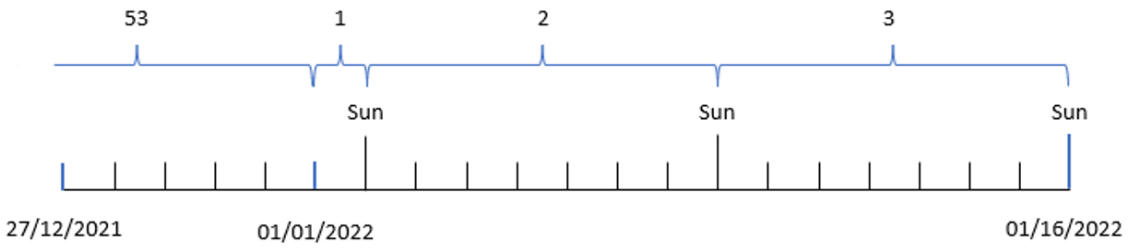
id	date	=week(date)	=weekday(date)
8199	01/12/2022	3	Wed
8200	01/13/2022	3	Thu
8201	01/14/2022	3	Fri

Il campo `week_number` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `week()` e trasferendo il campo `date` come argomento della funzione.

Non vengono passati altri parametri alla funzione e quindi sono attive le seguenti variabili predefinite che influenzano la funzione `week()`:

- `BrokenWeeks`: Il conteggio delle settimane inizia il 1° gennaio
- `FirstWeekDay`: Il primo giorno della settimana è la domenica

Schema della funzione `week()`, esempio di oggetto grafico



Poiché l'applicazione utilizza la variabile di sistema predefinita `BrokenWeeks`, la settimana 1 inizia il 1° gennaio, un sabato.

A causa della variabile di sistema `FirstWeekDay` predefinita, le settimane iniziano di domenica. La prima domenica dopo il 1° gennaio è il 2 gennaio, quando inizia la seconda settimana.

Esempio 6 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per l'ultima settimana del 2019 e le prime due settimane del 2020 caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

L'applicazione utilizza principalmente le settimane parziali nel suo dashboard. Tuttavia, l'utente finale desidera un oggetto grafico che presenti il totale delle vendite per settimana utilizzando settimane intere. Il giorno di riferimento dovrebbe essere il 2 gennaio, con le settimane che iniziano di martedì. Questo può essere ottenuto anche quando la dimensione non è disponibile nel modello dati, utilizzando la funzione `week()` come dimensione calcolata nel grafico.

Script di caricamento

```
SET BrokenWeeks=1;
SET ReferenceDay=0;
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8183,12/27/2019,58.27
8184,12/28/2019,67.42
8185,12/29/2019,23.80
8186,12/30/2019,82.06
8187,12/31/2019,40.56
8188,01/01/2020,37.23
8189,01/02/2020,17.17
8190,01/03/2020,88.27
8191,01/04/2020,57.42
8192,01/05/2020,53.80
8193,01/06/2020,82.06
8194,01/07/2020,40.56
8195,01/08/2020,53.67
8196,01/09/2020,26.63
8197,01/10/2020,72.48
8198,01/11/2020,18.37
8199,01/12/2020,45.26
8200,01/13/2020,58.23
8201,01/14/2020,18.52
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella.
2. Creare la seguente dimensione calcolata:
   `=week(date)`
3. Quindi, creare la seguente misura di aggregazione:
   `=sum(amount)`
4. Impostare la **Formattazione numero** della misura su **Denaro**.

5. Selezionare il menu **Ordinamento** e, per la dimensione calcolata, rimuovere l'ordinamento personalizzato.
6. Deselezionare le opzioni **Ordina per numero** e **Ordina alfabeticamente**.

Tabella dei risultati

week(date)	sum(amount)
52	$125.69
53	$146.42
1	$200.09
2	$347.57
3	$122.01

weekday

Questa funzione restituisce un valore duale con:

- Il nome di un giorno come definito nella variabile di ambiente **DayNames**.
- Un numero intero compreso tra 0 e 6 che corrisponde al giorno nominale della settimana (0-6).

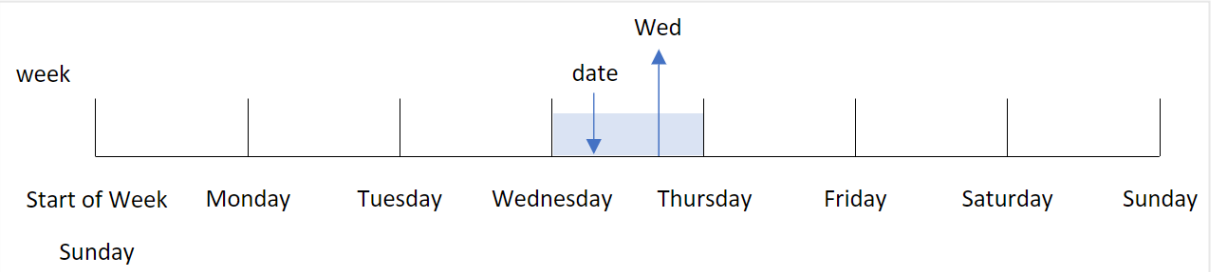
Sintassi:

```
weekday(date [,first_week_day=0])
```

Tipo di dati restituiti: duale

La funzione `weekday()` determina il giorno della settimana in cui si verifica una data. Restituisce quindi un valore stringa che rappresenta quel giorno.

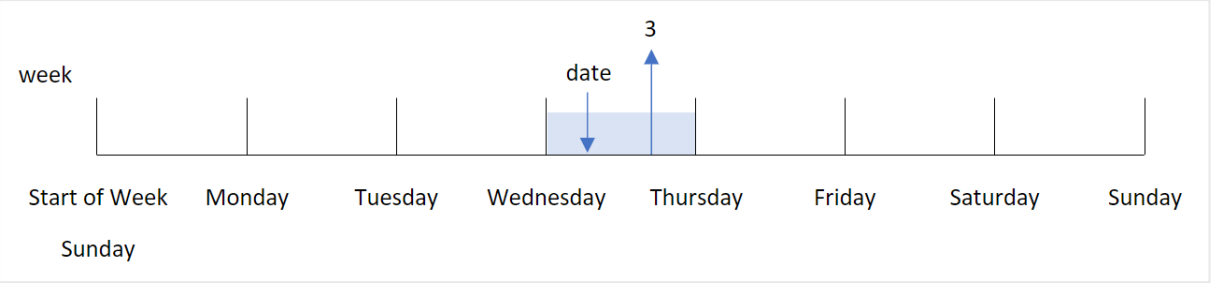
Schema della funzione `weekday()` che restituisce il nome del giorno in cui cade una data



Il risultato restituisce il valore numerico corrispondente a quel giorno della settimana (0-6), in base al giorno di inizio della settimana. Ad esempio, se il primo giorno della settimana è impostato su domenica, un mercoledì restituirà un valore numerico di `3`. Il giorno di inizio è determinato dalla variabile di sistema `FirstWeekDay` o dal parametro della funzione `first_week_day`.

È possibile utilizzare questo valore numerico come parte di un'espressione aritmetica. Ad esempio, moltiplicandolo per `1` per restituire il valore stesso.

Schema della funzione weekday() con il valore numerico del giorno visualizzato al posto del nome del giorno



Casi di utilizzo

La funzione `weekday()` è utile quando si desidera confrontare le aggregazioni per giorno della settimana. Ad esempio, se si desidera confrontare le vendite medie dei prodotti per giorno della settimana.

Queste dimensioni possono essere create nello script di caricamento utilizzando la funzione per creare un campo in una tabella del **Calendario principale**, oppure create direttamente in un grafico come misura calcolata.

Argomenti correlati

Argomenti	Interazione
FirstWeekDay (page 224)	Definisce il giorno di inizio di ogni settimana.

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
first_week_day	Specifica il giorno di inizio della settimana. Se omesso, viene utilizzato il valore della variabile **FirstWeekDay**. *FirstWeekDay (page 224)*

È possibile utilizzare i seguenti valori per impostare il giorno in cui inizia la settimana nell'argomento `first_week_day`:

valori first_week_day

Giorno	Valore
Lunedì	0
Martedì	1
Mercoledì	2
Giovedì	3
Venerdì	4
Sabato	5
Domenica	6

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Salvo ove indicato diversamente, `FirstWeekDay` è impostato su `0` in questi esempi.

Esempi di funzioni

Esempio	Risultato
`weekday('10/12/1971')`	Restituisce 'Tue' e 1.
`weekday('10/12/1971' , 6)`	Restituisce 'Tue' e 2. In questo esempio, domenica (`6`) è il primo giorno della settimana.

`SET FirstWeekDay=6;`
`...`
 	
`weekday('10/12/1971')`
	Restituisce 'Tue' e 2.

Esempio 1 - Stringa Weekday

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `FirstWeekDay` che è impostata su 6 (domenica).
- La variabile `DayNames` che è impostata per utilizzare i nomi dei giorni predefiniti.
- Un caricamento precedente che contiene la funzione `weekday()`, impostata come campo '`week_day`', e che restituisce il giorno della settimana in cui sono avvenute le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';
SET FirstWeekDay=6;

Transactions:
    Load
       *,
       WeekDay(date) as week_day
       ;
Load
*
Inline
[
id,date,amount
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.39
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `week_day`

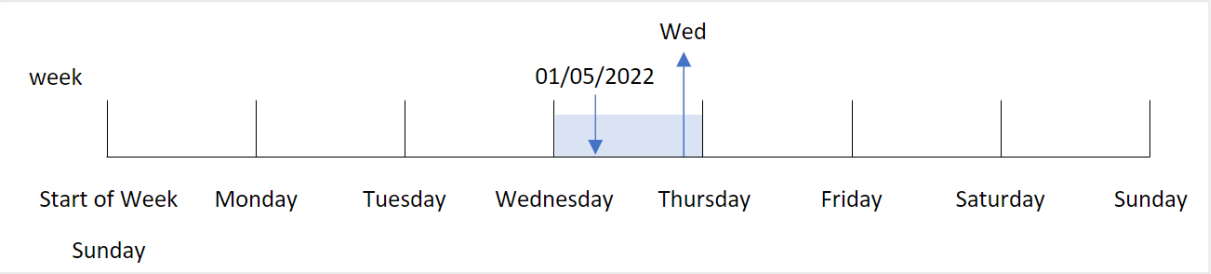
Tabella dei risultati

id	data	week_day
8188	01/01/2022	Sat
8189	01/02/2022	Sun
8190	01/03/2022	Mon
8191	01/04/2022	Tue
8192	01/05/2022	Wed
8193	01/06/2022	Thu
8194	01/07/2022	Fri

Il campo '`week_day`' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `weekday()` e trasferendo il campo data come argomento della funzione.

La funzione `weekday()` restituisce il valore della stringa del giorno della settimana, ovvero il nome del giorno della settimana impostato dalla variabile di sistema `DayNames`.

Schema della funzione `weekday()` che restituisce mercoledì come giorno della settimana per la transazione 8192



La transazione 8192 è avvenuta il 5 gennaio. La variabile di sistema `FirstWeekDay` imposta il primo giorno della settimana come domenica. La transazione della funzione `weekday()` ha avuto luogo un mercoledì e restituisce questo valore, nella forma abbreviata della variabile di sistema `DayNames`, nel campo `week_day`.

I valori del campo '`week_day`' sono allineati a destra nella colonna perché per il campo esiste un doppio risultato di numero e testo (mercoledì, 3). Per convertire il valore del campo nel suo equivalente numerico, il campo può essere avvolto all'interno della funzione `num()`. Ad esempio, nella transazione 8192, il valore del mercoledì verrà convertito nel numero 3.

Esempio 2 - first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `FirstWeekDay` che è impostata su 6 (domenica).
- La variabile `DayNames` che è impostata per utilizzare i nomi dei giorni predefiniti.
- Un caricamento precedente che contiene la funzione `weekday()`, impostata come campo '`week_day`', e che restituisce il giorno della settimana in cui sono avvenute le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';
SET FirstWeekDay=6;

Transactions:
    Load
       *,
       WeekDay(date,1) as week_day
       ;
```

```
Load
*
Inline
[
id,date,amount
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.39
];
```

Risultati

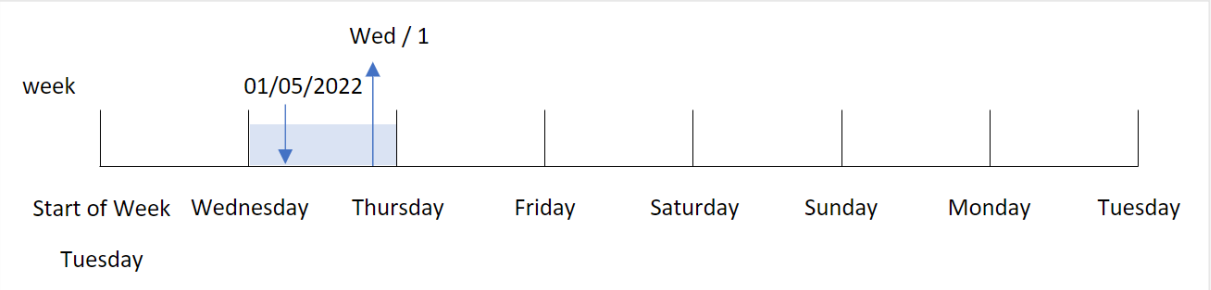
Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `week_day`

Tabella dei risultati

id	data	week_day
8188	01/01/2022	Sat
8189	01/02/2022	Sun
8190	01/03/2022	Mon
8191	01/04/2022	Tue
8192	01/05/2022	Wed
8193	01/06/2022	Thu
8194	01/07/2022	Fri

Schema della funzione `weekday()` che mostra che mercoledì ha il valore numerico duale di 1



Poiché l'argomento `first_week_day` è impostato su `1` nella funzione `weekday()`, il primo giorno della settimana è martedì. Pertanto, tutte le transazioni che avvengono di martedì avranno un valore numerico doppio di `0`.

La transazione 8192 è avvenuta il 5 gennaio. La funzione `weekday()` identifica che si tratta di un mercoledì e quindi l'espressione restituisce il valore numerico doppio di `1`.

Esempio 3 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `FirstWeekDay` che è impostata su `6` (domenica).
- La variabile `DayNames` che è impostata per utilizzare i nomi dei giorni predefiniti.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che identifica il valore del giorno della settimana viene creato come misura in un grafico dell'app.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';
SET FirstWeekDay=6;

Transactions:
Load
*
Inline
[
id,date,amount
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.39
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- date

Per calcolare il valore dei giorni feriali, creare la seguente misura:

- =weekday(date)

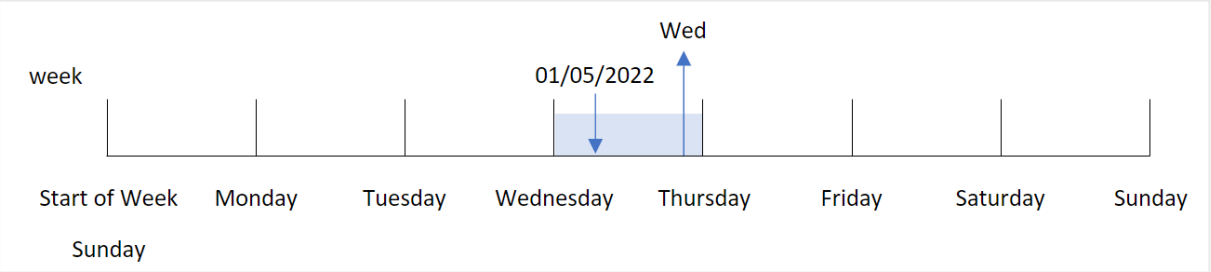
Tabella dei risultati

id	data	=weekday(date)
8188	01/01/2022	Sat
8189	01/02/2022	Sun
8190	01/03/2022	Mon
8191	01/04/2022	Tue
8192	01/05/2022	Wed
8193	01/06/2022	Thu
8194	01/07/2022	Fri

Il campo '=weekday(date)' viene creato nel grafico mediante l'utilizzo della funzione weekday() e trasferendo il campo data come argomento della funzione.

La funzione weekday() restituisce il valore della stringa del giorno della settimana, ovvero il nome del giorno della settimana impostato dalla variabile di sistema DayNames.

Schema della funzione weekday() che restituisce mercoledì come giorno della settimana per la transazione 8192



La transazione 8192 è avvenuta il 5 gennaio. La variabile di sistema FirstWeekDay imposta il primo giorno della settimana come domenica. La transazione della funzione weekday() ha avuto luogo un mercoledì e restituisce questo valore, nella forma abbreviata della variabile di sistema DayNames, nel campo =weekday(date).

Esempio 4 - Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata 'Transactions'.
- La variabile di sistema FirstWeekDay che è impostata su 6 (domenica).
- La variabile DayNames che è impostata per utilizzare i nomi dei giorni predefiniti.

L'utente finale desidera un grafico che presenti le vendite medie per giorno della settimana per le transazioni.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';
SET DayNames='Mon;Tue;Wed;Thu;Fri;Sat;Sun';
SET FirstWeekDay=6;

Transactions:
LOAD
  RecNo() AS id,
  MakeDate(2022, 1, Ceil(Rand() * 31)) as date,
  Rand() * 1000 AS amount

Autogenerate(1000);
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- =weekday(date)
- =avg(amount)

Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

weekday(date)	Avg(amount)
Sun	$536.96
Mon	$500.80
Tue	$515.63
Wed	$509.21
Thu	$482.70
Fri	$441.33
Sat	$505.22

weekend

Questa funzione restituisce un valore corrispondente a un timestamp recante l'ultimo millisecondo dell'ultimo giorno della settimana di calendario contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

Sintassi:

```
WeekEnd(timestamp [, period_no [, first_week_day ]])
```

Tipo di dati restituiti: duale

In altre parole, la funzione `weekend()` determina in quale settimana cade la data. Quindi, restituisce un timestamp, nel formato data, per l'ultimo millisecondo di quella settimana. Il primo giorno della settimana è determinato dalla variabile ambientale `FirstWeekDay`. Tuttavia, questo può essere sostituito dall'argomento `first_week_day` della funzione `weekend()`.

Argomenti

Argomento	Descrizione
timestamp	La data o la data e ora da valutare.
period_no	**shift** è un numero intero, in cui il valore 0 indica la settimana che contiene **date**. I valori negativi di shift indicano le settimane precedenti, mentre i valori positivi indicano le settimane successive.
first_week_day	Specifica il giorno di inizio della settimana. Se omesso, viene utilizzato il valore della variabile **FirstWeekDay**. I valori possibili per **first_week_day** sono 0 per lunedì, 1 per martedì, 2 per mercoledì, 3 per giovedì, 4 per venerdì, 5 per sabato e 6 per domenica. Per maggiori informazioni sulle variabili di sistema, vedere *FirstWeekDay (page 224)*.

Casi di utilizzo

La funzione `weekend()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi i giorni rimanenti della settimana per la data specificata. Ad esempio, potrebbe essere utilizzata se un utente desidera calcolare il totale degli interessi non ancora maturati durante la settimana.

Gli esempi seguenti presuppongono:

```
SET FirstWeekDay=0;
```

Esempio	Risultato
`weekend('01/10/2013')`	Restituisce 01/12/2013 23:59:59.
`weekend('01/10/2013', -1)`	Restituisce 01/05/2013 23:59:59..
`weekend('01/10/2013', 0, 1)`	Restituisce 01/14/2013 23:59:59.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi:

Se si desiderano le impostazioni ISO per le settimane e i numeri di settimana, assicurarsi di inserire nello script quanto segue:

```
Set DateFormat   ='YYYY-MM-DD';
Set FirstWeekDay =0;    // Monday as first week day
Set BrokenWeeks  =0;    //(use unbroken weeks)
Set ReferenceDay =4;    // Jan 4th is always in week 1
```

Se si desiderano le impostazioni USA, assicurarsi che nello script sia presente quanto segue:

```
Set DateFormat   ='M/D/YYYY';
Set FirstWeekDay =6;    // Sunday as first week day
Set BrokenWeeks  =1;    //(use broken weeks)
Set ReferenceDay =1;    // Jan 1st is always in week 1
```

Gli esempi sopra riportati danno come risultato quanto segue dalla funzione `weekend()`:

Esempio di funzione Weekend

Data	Fine settimana ISO	Fine settimana USA
Sab 26 dic 2020	2020-12-27	12/26/2020
Dom 27 dic 2020	2020-12-27	1/2/2021
Lun 28 dic 2020	2021-01-03	1/2/2021
Mar 29 dic 2020	2021-01-03	1/2/2021
Mer 30 dic 2020	2021-01-03	1/2/2021
Gio 31 dic 2020	2021-01-03	1/2/2021
Ven 1° gen 2021	2021-01-03	1/2/2021
Sab 2 gen 2021	2021-01-03	1/2/2021
Dom 3 gen 2021	2021-01-03	1/9/2021
Lun 4 gen 2021	2021-01-10	1/9/2021
Mar 5 gen 2021	2021-01-10	1/9/2021

Gli ultimi giorni della settimana corrispondono alla domenica nella colonna ISO e al sabato nella colonna USA.

Esempio 1 – Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `end_of_week`, che restituisce un timestamp per la fine della settimana in cui sono avvenute le transazioni.

Script di caricamento

```
SET FirstWeekDay=6;

Transactions:
    Load
        *,
        weekend(date) as end_of_week,
        timestamp(weekend(date)) as end_of_week_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
```

```
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `end_of_week`
- `end_of_week_timestamp`

Tabella dei risultati

date	end_of_week	end_of_week_timestamp
1/7/2022	01/08/2022	1/8/2022 11:59:59 PM
1/19/2022	01/22/2022	1/22/2022 11:59:59 PM
2/5/2022	02/05/2022	2/5/2022 11:59:59 PM
2/28/2022	03/05/2022	3/5/2022 11:59:59 PM
3/16/2022	03/19/2022	3/19/2022 11:59:59 PM
4/1/2022	04/02/2022	4/2/2022 11:59:59 PM
5/7/2022	05/07/2022	5/7/2022 11:59:59 PM
5/16/2022	05/21/2022	5/21/2022 11:59:59 PM
6/15/2022	06/18/2022	6/18/2022 11:59:59 PM
6/26/2022	07/02/2022	7/2/2022 11:59:59 PM
7/9/2022	07/09/2022	7/9/2022 11:59:59 PM
7/22/2022	07/23/2022	7/23/2022 11:59:59 PM
7/23/2022	07/23/2022	7/23/2022 11:59:59 PM
7/27/2022	07/30/2022	7/30/2022 11:59:59 PM
8/2/2022	08/06/2022	8/6/2022 11:59:59 PM
8/8/2022	08/13/2022	8/13/2022 11:59:59 PM
8/19/2022	08/20/2022	8/20/2022 11:59:59 PM
9/26/2022	10/01/2022	10/1/2022 11:59:59 PM
10/14/2022	10/15/2022	10/15/2022 11:59:59 PM
10/29/2022	10/29/2022	10/29/2022 11:59:59 PM

Il campo `end_of_week` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `weekend()` e trasferendo il campo data come argomento della funzione.

La funzione `weekend()` identifica in quale settimana rientra il valore della data e restituisce un timestamp per il primo millisecondo di quella settimana.

Schema della funzione `weekend()`, esempio base

La transazione 8191 è avvenuta il 5 febbraio. La variabile di sistema `FirstWeekDay` imposta il primo giorno della settimana come domenica. La funzione `weekend()` identifica che il primo sabato dopo il 5 febbraio - e quindi la fine della settimana - era il 5 febbraio. Pertanto, il valore `end_of_week` per questa transazione restituisce l'ultimo millisecondo di quel giorno, ovvero il 5 febbraio alle 23:59:59.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_week_end`, che restituisce il timestamp dell'inizio della settimana precedente la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        weekend(date,-1) as previous_week_end,
        timestamp(weekend(date,-1)) as previous_week_end_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
```

```
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_week_end`
- `previous_week_end_timestamp`

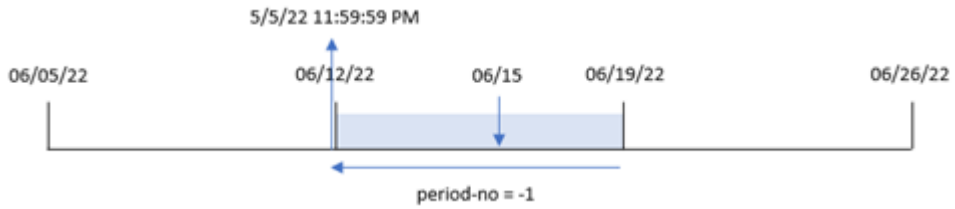
Tabella dei risultati

date	end_of_week	end_of_week_timestamp
1/7/2022	01/01/2022	1/1/2022 11:59:59 PM
1/19/2022	01/15/2022	1/15/2022 11:59:59 PM
2/5/2022	01/29/2022	1/29/2022 11:59:59 PM
2/28/2022	02/26/2022	2/26/2022 11:59:59 PM
3/16/2022	03/12/2022	3/12/2022 11:59:59 PM
4/1/2022	03/26/2022	3/26/2022 11:59:59 PM
5/7/2022	04/30/2022	4/30/2022 11:59:59 PM
5/16/2022	05/14/2022	5/14/2022 11:59:59 PM
6/15/2022	06/11/2022	6/11/2022 11:59:59 PM
6/26/2022	06/25/2022	6/25/2022 11:59:59 PM
7/9/2022	07/02/2022	7/2/2022 11:59:59 PM
7/22/2022	07/16/2022	7/16/2022 11:59:59 PM
7/23/2022	07/16/2022	7/16/2022 11:59:59 PM

date	end_of_week	end_of_week_timestamp
7/27/2022	07/23/2022	7/23/2022 11:59:59 PM
8/2/2022	07/30/2022	7/30/2022 11:59:59 PM
8/8/2022	08/06/2022	8/6/2022 11:59:59 PM
8/19/2022	08/13/2022	8/13/2022 11:59:59 PM
9/26/2022	09/24/2022	9/24/2022 11:59:59 PM
10/14/2022	10/08/2022	10/8/2022 11:59:59 PM
10/29/2022	10/22/2022	10/22/2022 11:59:59 PM

In questo caso, poiché il valore `period_no` di -1 è stato utilizzato come argomento offset nella funzione `weekend()`, la funzione per prima cosa identifica la settimana in cui sono avvenute le transazioni. Quindi, cerca la settimana precedente e identifica l'ultimo millisecondo di quella settimana.

Schema della funzione `weekend()`, esempio di period_no



La transazione 8196 è avvenuta il 15 giugno. La funzione `weekend()` identifica che la settimana è iniziata il 12 giugno. Pertanto, la settimana precedente termina l'11 giugno alle 11:59:59 PM; ciò rappresenta il valore restituito per il campo `previous_week_end`.

Esempio 3 – first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, dobbiamo impostare il martedì come primo giorno della settimana lavorativa.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        weekend(date,0,1) as end_of_week,
        timestamp(weekend(date,0,1)) as end_of_week_timestamp,
```

```
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `end_of_week`
- `end_of_week_timestamp`

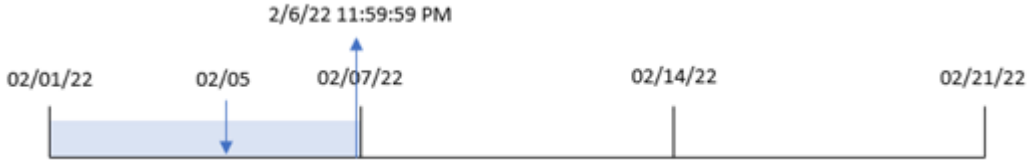
Tabella dei risultati

date	end_of_week	end_of_week_timestamp
1/7/2022	01/10/2022	1/10/2022 11:59:59 PM
1/19/2022	01/24/2022	1/24/2022 11:59:59 PM
2/5/2022	02/07/2022	2/7/2022 11:59:59 PM
2/28/2022	02/28/2022	2/28/2022 11:59:59 PM
3/16/2022	03/21/2022	3/21/2022 11:59:59 PM
4/1/2022	04/04/2022	4/4/2022 11:59:59 PM
5/7/2022	05/09/2022	5/9/2022 11:59:59 PM
5/16/2022	05/16/2022	5/16/2022 11:59:59 PM

date	end_of_week	end_of_week_timestamp
6/15/2022	06/20/2022	6/20/2022 11:59:59 PM
6/26/2022	06/27/2022	6/27/2022 11:59:59 PM
7/9/2022	07/11/2022	7/11/2022 11:59:59 PM
7/22/2022	07/25/2022	7/25/2022 11:59:59 PM
7/23/2022	07/25/2022	7/25/2022 11:59:59 PM
7/27/2022	08/01/2022	8/1/2022 11:59:59 PM
8/2/2022	08/08/2022	8/8/2022 11:59:59 PM
8/8/2022	08/08/2022	8/8/2022 11:59:59 PM
8/19/2022	08/22/2022	8/22/2022 11:59:59 PM
9/26/2022	09/26/2022	9/26/2022 11:59:59 PM
10/14/2022	10/17/2022	10/17/2022 11:59:59 PM
10/29/2022	10/31/2022	10/31/2022 11:59:59 PM

In questa istanza, dato che l'argomento `first_week_date` di 1 è utilizzato nella funzione `weekend()`, imposta il primo giorno della settimana a martedì.

Schema della funzione `weekend()`, esempio first_week_day



La transazione 8191 è avvenuta il 5 febbraio. La funzione `weekend()` identifica che il primo lunedì successivo a questa data - e quindi la fine della settimana e il valore restituito - era il 6 febbraio alle 23:59:59.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce un timestamp per la fine della settimana in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Per calcolare l'inizio della settimana in cui avviene una transazione, aggiungere le seguenti misure:

- `=weekend(date)`
- `=timestamp(weekend(date))`

Tabella dei risultati

date	=weekend(date)	=timestamp(weekend(date))
1/7/2022	01/08/2022	1/8/2022 11:59:59 PM
1/19/2022	01/22/2022	1/22/2022 11:59:59 PM
2/5/2022	02/05/2022	2/5/2022 11:59:59 PM
2/28/2022	03/05/2022	3/5/2022 11:59:59 PM
3/16/2022	03/19/2022	3/19/2022 11:59:59 PM

date	=weekend(date)	=timestamp(weekend(date))
4/1/2022	04/02/2022	4/2/2022 11:59:59 PM
5/7/2022	05/07/2022	5/7/2022 11:59:59 PM
5/16/2022	05/21/2022	5/21/2022 11:59:59 PM
6/15/2022	06/18/2022	6/18/2022 11:59:59 PM
6/26/2022	07/02/2022	7/2/2022 11:59:59 PM
7/9/2022	07/09/2022	7/9/2022 11:59:59 PM
7/22/2022	07/23/2022	7/23/2022 11:59:59 PM
7/23/2022	07/23/2022	7/23/2022 11:59:59 PM
7/27/2022	07/30/2022	7/30/2022 11:59:59 PM
8/2/2022	08/06/2022	8/6/2022 11:59:59 PM
8/8/2022	08/13/2022	8/13/2022 11:59:59 PM
8/19/2022	08/20/2022	8/20/2022 11:59:59 PM
9/26/2022	10/01/2022	10/1/2022 11:59:59 PM
10/14/2022	10/15/2022	10/15/2022 11:59:59 PM
10/29/2022	10/29/2022	10/29/2022 11:59:59 PM

La misura `end_of_week` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `weekend()` e trasferendo il campo data come argomento della funzione. La funzione `weekend()` identifica in quale settimana cade il valore della data, restituendo un timestamp per l'ultimo millisecondo di quella settimana.

Schema della funzione `weekend()`, esempio di oggetto grafico

La transazione 8191 è avvenuta il 5 febbraio. La variabile di sistema `FirstWeekDay` imposta il primo giorno della settimana come domenica. La funzione `weekend()` identifica che il primo sabato dopo il 5 febbraio - e quindi la fine della settimana - era il 5 febbraio. Pertanto, il valore `end_of_week` per questa transazione restituisce l'ultimo millisecondo di quel giorno, ovvero il 5 febbraio alle 23:59:59.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Employee_Expenses`.
- I dati consistono negli ID dei dipendenti, nei nomi dei dipendenti e nelle richieste di rimborso delle spese medie giornaliere di ciascun dipendente.

L'utente finale desidera un oggetto grafico che visualizzi, in base all'ID dipendente e al nome del dipendente, le richieste di rimborso spese stimate ancora da sostenere per il resto della settimana.

Script di caricamento

```
Employee_Expenses:
Load
*
Inline
[
employee_id,employee_name,avg_daily_claim
182,Mark, $15
183,Deryck, $12.5
184,Dexter, $12.5
185,Sydney,$27
186,Agatha,$18
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:
   - `employee_id`
   - `employee_name`
2. Quindi, creare una misura per calcolare gli interessi accumulati:
   `=(weekend(today(1))-today(1))*avg_daily_claim`
3. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

employee_id	employee_name	=(weekend(today(1))-today(1))*avg_daily_claim
182	Contrassegno	$90.00
183	Deryck	$75.00
184	Dexter	$75.00
185	Sydney	$162.00
186	Agatha	$108.00

La funzione`weekend()`, utilizzando come unico argomento la data odierna, restituisce la data finale della settimana corrente. Quindi, sottraendo la data odierna dalla data di fine settimana, l'espressione restituisce il numero di giorni rimanenti di questa settimana.

Questo valore viene quindi moltiplicato per la media delle richieste di rimborso spese giornaliere di ciascun dipendente per calcolare il valore stimato delle richieste che ogni dipendente dovrebbe presentare durante il periodo rimanente della settimana.

weekname

Questa funzione restituisce un valore che mostra l'anno e il numero della settimana con un valore numerico sottostante corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno della settimana contenente **date**.

Sintassi:

```
WeekName(date[, period_no [, first_week_day [, broken_weeks [, reference_
day]]]])
```

La funzione `weekname()` determina la settimana in cui cade la data e restituisce il numero della settimana e l'anno per quella settimana. Il primo giorno della settimana è determinato dalla variabile di sistema `FirstWeekDay`. Tuttavia, è possibile anche modificare il primo giorno della settimana utilizzando l'argomento `first_week_day` nella funzione `weekname()`.

In Qlik Sense, le impostazioni regionali vengono recuperate alla creazione dell'app e le impostazioni corrispondenti vengono memorizzate nello script come variabili d'ambiente.

Uno sviluppatore di app nordamericano ottiene spesso `Set BrokenWeeks=1;` nello script, corrispondente a settimane interrotte. Uno sviluppatore di app europeo ottiene spesso `Set BrokenWeeks=0;` nello script, corrispondente a settimane intere.

Se l'applicazione utilizza settimane interrotte, il conteggio del numero di settimane inizia il 1° gennaio e termina il giorno precedente alla variabile di sistema `FirstWeekDay`, indipendentemente dal numero di giorni trascorsi.

Tuttavia, se l'applicazione utilizza settimane ininterrotte, la settimana 1 può iniziare nell'anno precedente o nei primi giorni di gennaio. Dipende da come si utilizzano le variabili di sistema `ReferenceDay` e `FirstWeekDay`.

Esempio di funzione Weekname

Data	Nome settimana ISO	Nome settimana US
Sab 26 dic 2020	2020/52	2020/52
Dom 27 dic 2020	2020/52	2020/53
Lun 28 dic 2020	2020/53	2020/53
Mar 29 dic 2020	2020/53	2020/53
Mer 30 dic 2020	2020/53	2020/53
Gio 31 dic 2020	2020/53	2020/53

Data	Nome settimana ISO	Nome settimana US
Ven 1° gen 2021	2020/53	2021/01
Sab 2 gen 2021	2020/53	2021/01
Dom 3 gen 2021	2020/53	2021/02
Lun 4 gen 2021	2021/01	2021/02
Mar 5 gen 2021	2021/01	2021/02

Casi di utilizzo

La funzione `weekname()` è utile per confrontare le aggregazioni per settimane.

Ad esempio, può essere usata se si desidera visualizzare le vendite totali dei prodotti in base alla settimana. Per mantenere la coerenza con la variabile di ambiente `BrokenWeeks` nell'applicazione, utilizzare `weekname()` anziché `lunarweekname()`. Se l'applicazione utilizza settimane ininterrotte, la settimana 1 può contenere date di dicembre dell'anno precedente o escludere date di gennaio dell'anno in corso. Se l'applicazione utilizza settimane interrotte, la settimana 1 può contenere meno di sette giorni.

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
timestamp	La data o la data e ora da valutare.
period_no	**shift** è un numero intero, in cui il valore 0 indica la settimana che contiene **date**. I valori negativi di shift indicano le settimane precedenti, mentre i valori positivi indicano le settimane successive.
first_week_day	Specifica il giorno di inizio della settimana. Se omesso, viene utilizzato il valore della variabile **FirstWeekDay**. I valori possibili per **first_week_day** sono 0 per lunedì, 1 per martedì, 2 per mercoledì, 3 per giovedì, 4 per venerdì, 5 per sabato e 6 per domenica. Per maggiori informazioni sulle variabili di sistema, vedere *FirstWeekDay (page 224)*.
broken_weeks	Se non si specifica **broken_weeks**, il valore della variabile **BrokenWeeks** verrà utilizzato per definire se le settimane sono parziali o meno.
reference_day	Se non si specifica **reference_day**, il valore della variabile **ReferenceDay** verrà utilizzato per definire quale giorno di gennaio impostare come giorno di riferimento per definire la settimana 1. Per impostazione predefinita, le funzioni Qlik Sense utilizzano 4 come giorno di riferimento. Questo significa che la settimana 1 deve contenere il 4 gennaio, vale a dire che la settimana 1 deve sempre contenere 4 quattro giorni di gennaio.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Gli esempi riportati di seguito presuppongono:

```
Set FirstWeekDay=0;
Set BrokenWeeks=0;
Set ReferenceDay=4;
```

Esempi di funzioni

Esempio	Risultato
weekname('01/12/2013')	Restituisce 2013/02.
weekname('01/12/2013', -1)	Returns 2013/01.
weekname('01/12/2013', 0, 1)	Restituisce 2013/02.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- una serie di dati contenente un insieme di transazioni per l'ultima settimana del 2021 e le prime due settimane del 2022 caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `DateFormat` che è impostata sul formato `MM/DD/YYYY`.
- La variabile di sistema `BrokenWeeks` che è impostata su `1`.
- La variabile di sistema `FirstWeekDay` che è impostata su `6`.
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
    - La funzione `weekday()` che è impostata come campo, '`week_number`', che restituisce il numero dell'anno e della settimana in cui sono state effettuate le transazioni.

- La funzione `weekname()` che è impostata come campo denominato '`week_day`', per mostrare il valore del giorno della settimana di ogni data di transazione.

Script di caricamento

```
SET BrokenWeeks=1;
SET DateFormat='MM/DD/YYYY';
SET FirstWeekDay=6;

Transactions:
   Load
     *,
     WeekDay(date) as week_day,
     Weekname(date) as week_number
     ;
Load
*
Inline
[
id,date,amount
8183,12/27/2021,58.27
8184,12/28/2021,67.42
8185,12/29/2021,23.80
8186,12/30/2021,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
8198,01/11/2022,18.37
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `week_day`
- `week_number`

Tabella dei risultati

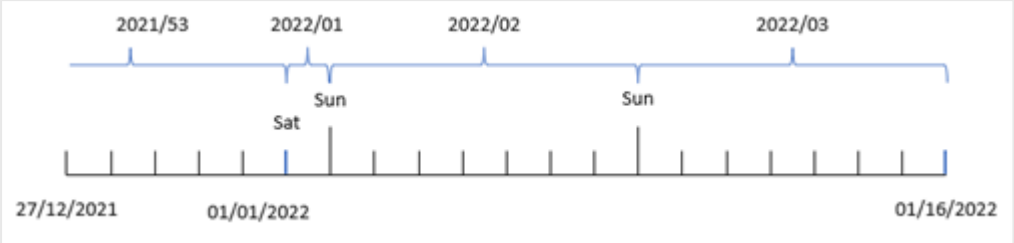
id	data	week_day	week_number
8183	12/27/2021	Mon	2021/53
8184	12/28/2021	Tue	2021/53
8185	12/29/2021	Wed	2021/53
8186	12/30/2021	Thu	2021/53
8187	12/31/2021	Fri	2021/53
8188	01/01/2022	Sat	2022/01
8189	01/02/2022	Sun	2022/02
8190	01/03/2022	Mon	2022/02
8191	01/04/2022	Tue	2022/02
8192	01/05/2022	Wed	2022/02
8193	01/06/2022	Thu	2022/02
8194	01/07/2022	Fri	2022/02
8195	01/08/2022	Sat	2022/02
8196	01/09/2022	Sun	2022/03
8197	01/10/2022	Mon	2022/03
8198	01/11/2022	Tue	2022/03
8199	01/12/2022	Wed	2022/03
8200	01/13/2022	Thu	2022/03
8201	01/14/2022	Fri	2022/03

Il campo '`week_number`' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `weekname()` e trasferendo il campo data come argomento della funzione.

La funzione `weekname()` identifica inizialmente la settimana in cui rientra il valore della data e restituisce il conteggio del numero della settimana e l'anno in cui si verifica la transazione.

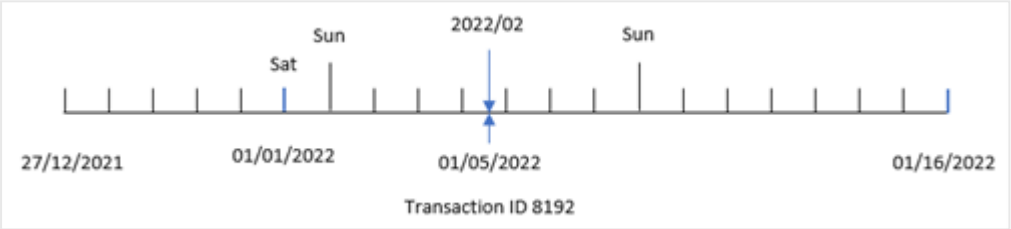
La variabile di sistema `FirstWeekDay` imposta la domenica come primo giorno della settimana. La variabile di sistema `BrokenWeeks` imposta l'applicazione in modo che utilizzi le settimane interrotte, quindi la settimana 1 inizierà il 1 gennaio.

Schema della funzione weekname() con le variabili predefinite.



La settimana 1 inizia il 1 gennaio, che è un sabato, pertanto le transazioni che si verificano in questa data restituiscono il valore `2022/01` (l'anno e il numero della settimana).

Diagramma della funzione weekname() che identifica il numero della settimana per la transazione 8192.



Poiché l'applicazione utilizza settimane interrotte e il primo giorno della settimana è la domenica, le transazioni che si verificano dal 2 all'8 gennaio restituiscono il valore `2022/02` (settimana numero 2 nel 2022). Un esempio potrebbe essere la transazione 8192 avvenuta il 5 gennaio che restituisce il valore `2022/02` per il campo '`week_number`'.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, l'attività consiste nel creare un campo, '`previous_week_number`', che restituisce l'anno e il numero della settimana precedente alla data in cui sono state eseguite le transazioni.

Aprire Editor caricamento dati, quindi aggiungere il seguente script di caricamento in una nuova scheda.

Script di caricamento

```
SET BrokenWeeks=1;
SET FirstWeekDay=6;

Transactions:
   Load
     *,
     weekname(date,-1) as previous_week_number
     ;
Load
```

```
*
Inline
[
id,date,amount
8183,12/27/2021,58.27
8184,12/28/2021,67.42
8185,12/29/2021,23.80
8186,12/30/2021,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
8198,01/11/2022,18.37
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `week_day`
- `week_number`

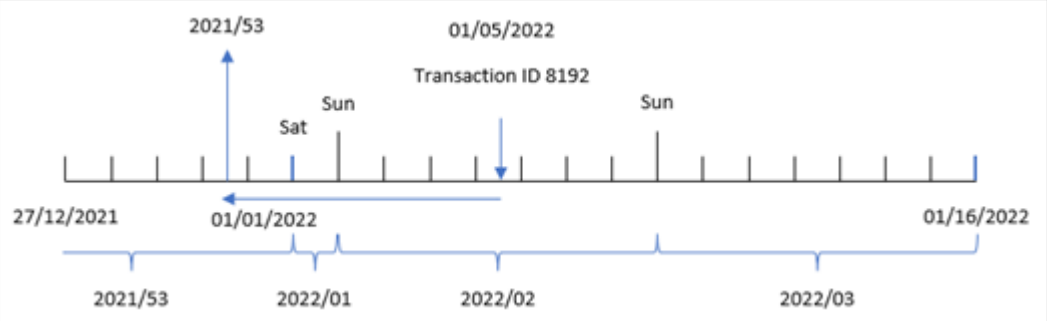
Tabella dei risultati

id	data	week_day	week_number
8183	12/27/2021	Mon	2021/52
8184	12/28/2021	Tue	2021/52
8185	12/29/2021	Wed	2021/52
8186	12/30/2021	Thu	2021/52
8187	12/31/2021	Fri	2021/52
8188	01/01/2022	Sat	2021/52
8189	01/02/2022	Sun	2021/53
8190	01/03/2022	Mon	2021/53
8191	01/04/2022	Tue	2021/53

id	data	week_day	week_number
8192	01/05/2022	Wed	2021/53
8193	01/06/2022	Thu	2021/53
8194	01/07/2022	Fri	2021/53
8195	01/08/2022	Sat	2022/01
8196	01/09/2022	Sun	2022/02
8197	01/10/2022	Mon	2022/02
8198	01/11/2022	Tue	2022/02
8199	01/12/2022	Wed	2022/02
8200	01/13/2022	Thu	2022/02
8201	01/14/2022	Fri	2022/02

Poiché il valore `period_no` di `-1` è stato utilizzato come argomento offset nella funzione `weekname()`, la funzione per prima cosa identifica la settimana in cui avvengono le transazioni. Quindi, cerca la settimana precedente e identifica il primo millisecondo di quella settimana.

Schema della funzione `weekname()` con un offset `period_no` di -1.



La transazione 8192 è avvenuta il 5 gennaio 2022. La funzione `weekname()` cerca una settimana prima, il 30 dicembre 2021, e restituisce il numero della settimana e l'anno per quella data - `2021/53`.

Esempio 3 - first_week_day

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, la politica aziendale prevede che la settimana lavorativa inizi il martedì.

Aprire Editor caricamento dati, quindi aggiungere il seguente script di caricamento in una nuova scheda.

Script di caricamento

```
SET BrokenWeeks=1;
SET DateFormat='MM/DD/YYYY';

Transactions:
   Load
     *,
     weekday(date) as week_day,
     weekname(date,0,1) as week_number
     ;
Load
*
Inline
[
id,date,amount
8183,12/27/2021,58.27
8184,12/28/2021,67.42
8185,12/29/2021,23.80
8186,12/30/2021,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
8198,01/11/2022,18.37
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

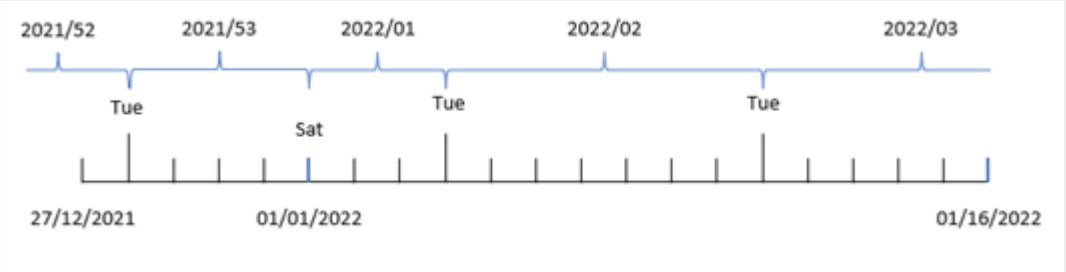
- `id`
- `date`
- `week_day`
- `week_number`

Tabella dei risultati

id	data	week_day	week_number
8183	12/27/2021	Mon	2021/52

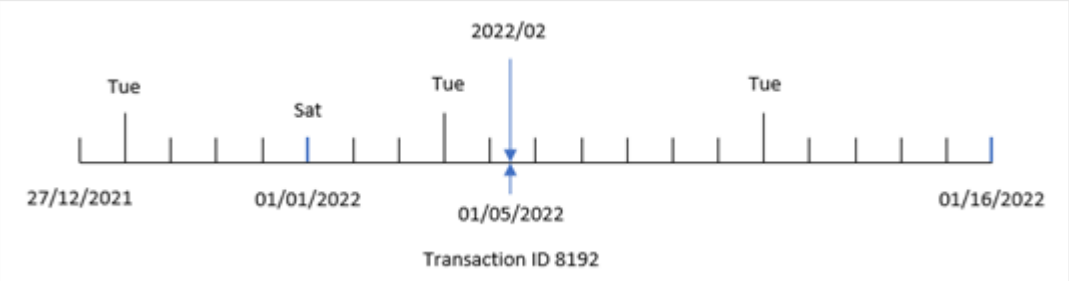
id	data	week_day	week_number
8184	12/28/2021	Tue	2021/53
8185	12/29/2021	Wed	2021/53
8186	12/30/2021	Thu	2021/53
8187	12/31/2021	Fri	2021/53
8188	01/01/2022	Sat	2022/01
8189	01/02/2022	Sun	2022/01
8190	01/03/2022	Mon	2022/01
8191	01/04/2022	Tue	2022/02
8192	01/05/2022	Wed	2022/02
8193	01/06/2022	Thu	2022/02
8194	01/07/2022	Fri	2022/02
8195	01/08/2022	Sat	2022/02
8196	01/09/2022	Sun	2022/02
8197	01/10/2022	Mon	2022/02
8198	01/11/2022	Tue	2022/03
8199	01/12/2022	Wed	2022/03
8200	01/13/2022	Thu	2022/03
8201	01/14/2022	Fri	2022/03

Schema della funzione `weekname()` con martedì come primo giorno della settimana.



Poiché l'argomento `first_week_date` di `1` viene utilizzato nella funzione `weekname()`, il martedì viene utilizzato come primo giorno della settimana. La funzione determina quindi che la settimana 53 del 2021 è iniziata martedì 28 dicembre; e, a causa dell'utilizzo da parte dell'applicazione di settimane interrotte, la settimana 1 inizia il 1 gennaio 2022 e termina l'ultimo millisecondo di lunedì 3 gennaio 2022.

Diagramma che mostra il numero della settimana della transazione 8192 con martedì come primo giorno della settimana.



La transazione 8192 è avvenuta il 5 gennaio 2022. Pertanto, utilizzando il parametro `first_week_day` di martedì, la funzione `weekname()` restituisce il valore `2022/02` per il campo '`week_number`'.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che restituisce il numero dell'anno per la fine settimana in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
SET BrokenWeeks=1;
Transactions:
Load
*
Inline
[
id,date,amount
8183,12/27/2021,58.27
8184,12/28/2021,67.42
8185,12/29/2021,23.80
8186,12/30/2021,82.06
8187,12/31/2021,40.56
8188,01/01/2022,37.23
8189,01/02/2022,17.17
8190,01/03/2022,88.27
8191,01/04/2022,57.42
8192,01/05/2022,53.80
8193,01/06/2022,82.06
8194,01/07/2022,40.56
8195,01/08/2022,53.67
8196,01/09/2022,26.63
8197,01/10/2022,72.48
8198,01/11/2022,18.37
```

```
8199,01/12/2022,45.26
8200,01/13/2022,58.23
8201,01/14/2022,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `=week_day (date)`

Per calcolare l'inizio della settimana in cui è avvenuta una transazione, creare la seguente misura:

```
=weekname(date)
```

Tabella dei risultati

id	data	=weekday(date)	=weekname(date)
8183	12/27/2021	Mon	2021/53
8184	12/28/2021	Tue	2021/53
8185	12/29/2021	Wed	2021/53
8186	12/30/2021	Thu	2021/53
8187	12/31/2021	Fri	2021/53
8188	01/01/2022	Sat	2022/01
8189	01/02/2022	Sun	2022/02
8190	01/03/2022	Mon	2022/02
8191	01/04/2022	Tue	2022/02
8192	01/05/2022	Wed	2022/02
8193	01/06/2022	Thu	2022/02
8194	01/07/2022	Fri	2022/02
8195	01/08/2022	Sat	2022/02
8196	01/09/2022	Sun	2022/03
8197	01/10/2022	Mon	2022/03
8198	01/11/2022	Tue	2022/03
8199	01/12/2022	Wed	2022/03
8200	01/13/2022	Thu	2022/03
8201	01/14/2022	Fri	2022/03

Il campo `week_number` viene creato come misura nell'oggetto grafico mediante l'utilizzo della funzione `weekname()` e trasferendo il campo data come argomento della funzione.

La funzione `weekname()` identifica inizialmente la settimana in cui rientra il valore della data e restituisce il conteggio del numero della settimana e l'anno in cui si è verificata la transazione.

La variabile di sistema imposta la domenica come primo giorno della settimana.FirstWeekDay La variabile di sistema `BrokenWeeks` imposta l'applicazione in modo che utilizzi le settimane interrotte, quindi la settimana 1 inizia il 1 gennaio.

Diagramma che mostra il numero della settimana con domenica come primo giorno della settimana.

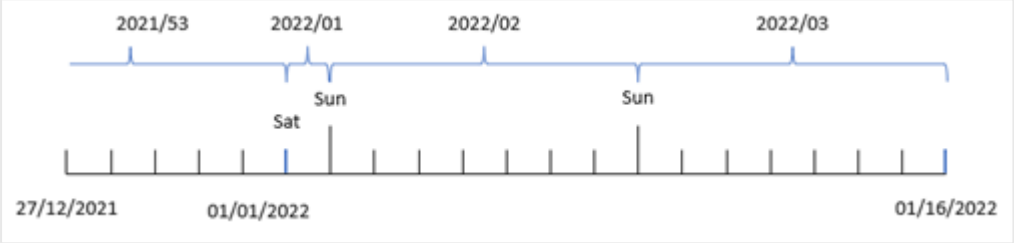
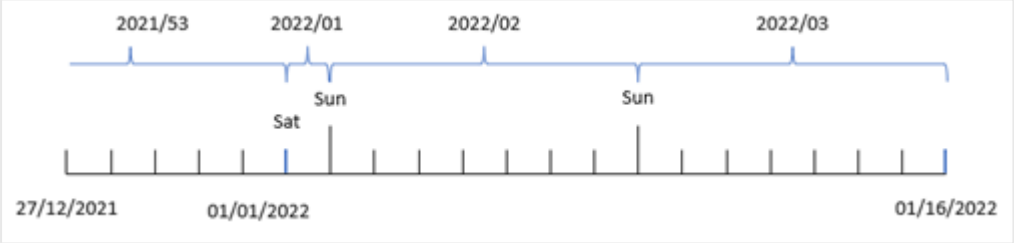


Diagramma che mostra che la transazione 8192 ha avuto luogo nella settimana numero due.



Poiché l'applicazione utilizza settimane interrotte e il primo giorno della settimana è la domenica, le transazioni che si verificano dal 2 all'8 gennaio restituiscono il valore 2022/02, ossia la settimana numero 2 del 2022. Si noti che la transazione 8192 è avvenuta il 5 gennaio e restituisce il valore 2022/02 per il campo '`week_number`'.

Esempio 5 – Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- una serie di dati contenente un insieme di transazioni per l'ultima settimana del 2019 e le prime due settimane del 2020 caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `BrokenWeeks` che è impostata su `0`.

- La variabile di sistema `ReferenceDay` che è impostata su `2`.
- La variabile di sistema che è impostata sul formato .DateFormatMM/DD/YYYY

Script di caricamento

```
SET BrokenWeeks=0;
SET ReferenceDay=2;
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8183,12/27/2019,58.27
8184,12/28/2019,67.42
8185,12/29/2019,23.80
8186,12/30/2019,82.06
8187,12/31/2019,40.56
8188,01/01/2020,37.23
8189,01/02/2020,17.17
8190,01/03/2020,88.27
8191,01/04/2020,57.42
8192,01/05/2020,53.80
8193,01/06/2020,82.06
8194,01/07/2020,40.56
8195,01/08/2020,53.67
8196,01/09/2020,26.63
8197,01/10/2020,72.48
8198,01/11/2020,18.37
8199,01/12/2020,45.26
8200,01/13/2020,58.23
8201,01/14/2020,18.52
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella.

Creare una dimensione calcolata utilizzando la seguente espressione:

```
=weekname(date)
```

Per calcolare le vendite totali, creare la seguente misura di aggregazione:

```
=sum(amount)
```

Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

weekname(date)	=sum(amount)
2019/52	$125.69
2020/01	$346.51
2020/02	$347.57
2020/03	$122.01

Per dimostrare i risultati dell'utilizzo della funzione weekname() in questo scenario, aggiungere il campo seguente come dimensione:

```
date
```

Tabella dei risultati con campo data

weekname(date)	data	=sum(amount)
2019/52	12/27/2019	$58.27
2019/52	12/28/2019	$67.42
2020/01	12/29/2019	$23.80
2020/01	12/30/2019	$82.06
2020/01	12/31/2019	$40.56
2020/01	01/01/2020	$37.23
2020/01	01/02/2020	$17.17
2020/01	01/03/2020	$88.27
2020/01	01/04/2020	$57.42
2020/02	01/05/2020	$53.80
2020/02	01/06/2020	$82.06
2020/02	01/07/2020	$40.56
2020/02	01/08/2020	$53.67
2020/02	01/09/2020	$26.63
2020/02	01/10/2020	$72.48
2020/02	01/11/2020	$18.37
2020/03	01/12/2020	$45.26
2020/03	01/13/2020	$58.23
2020/03	01/14/2020	$18.52

Poiché l'applicazione utilizza settimane ininterrotte e la settimana 1 richiede un minimo di due giorni a gennaio a causa della variabile di sistema `ReferenceDay`, la settimana 1 del 2020 include le transazioni del 29 dicembre 2019.

weekstart

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno della settimana di calendario contenente **date**. Il formato di output predefinito è il formato **DateFormat** impostato nello script.

Sintassi:

```
WeekStart(timestamp [, period_no [, first_week_day ]])
```

Tipo di dati restituiti: duale

In altre parole, la funzione `weekstart()` determina in quale settimana cade la data. Quindi, restituisce un timestamp, nel formato data, per il primo millisecondo di quella settimana. Il primo giorno della settimana è determinato dalla variabile ambientale `FirstWeekDay`. Tuttavia, questo può essere sostituito dall'argomento `first_week_day` della funzione `weekstart()`.

Argomenti

Argomento	Descrizione
timestamp	La data o la data e ora da valutare.
period_no	**shift** è un numero intero, in cui il valore 0 indica la settimana che contiene **date**. I valori negativi di shift indicano le settimane precedenti, mentre i valori positivi indicano le settimane successive.
first_week_day	Specifica il giorno di inizio della settimana. Se omesso, viene utilizzato il valore della variabile **FirstWeekDay**. I valori possibili per **first_week_day** sono 0 per lunedì, 1 per martedì, 2 per mercoledì, 3 per giovedì, 4 per venerdì, 5 per sabato e 6 per domenica. Per maggiori informazioni sulle variabili di sistema, vedere *FirstWeekDay (page 224)*.

Casi di utilizzo

La funzione `weekstart()` viene comunemente utilizzata come parte di un'espressione quando l'utente desidera che il calcolo utilizzi la frazione della settimana trascorsa finora. Ad esempio, potrebbe essere utilizzato se un utente desidera calcolare il totale dei salari guadagnati dai dipendenti nella settimana in corso.

Gli esempi seguenti presuppongono:

```
SET FirstWeekDay=0;
```

Esempi di funzioni

Esempio	Risultato
`weekstart('01/12/2013')`	Restituisce 01/07/2013.
`weekstart('01/12/2013', -1 )`	Restituisce 11/31/2012.
`weekstart('01/12/2013', 0, 1)`	Restituisce 01/08/2013.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi:

Se si desiderano le impostazioni ISO per le settimane e i numeri di settimana, assicurarsi di inserire nello script quanto segue:

```
Set DateFormat   ='YYYY-MM-DD';
Set FirstWeekDay =0;    // Monday as first week day
Set BrokenWeeks  =0;    //(use unbroken weeks)
Set ReferenceDay =4;    // Jan 4th is always in week 1
```

Se si desiderano le impostazioni USA, assicurarsi che nello script sia presente quanto segue:

```
Set DateFormat   ='M/D/YYYY';
Set FirstWeekDay =6;    // Sunday as first week day
Set BrokenWeeks  =1;    //(use broken weeks)
Set ReferenceDay =1;    // Jan 1st is always in week 1
```

Gli esempi sopra riportati danno come risultato quanto segue dalla funzione `weekstart()`:

Esempio di funzione Weekstart

Data	Inizio settimana ISO	Inizio settimana US
Sab 26 dic 2020	2020-12-21	12/20/2020
Dom 27 dic 2020	2020-12-21	12/27/2020
Lun 28 dic 2020	2020-12-28	12/27/2020
Mar 29 dic 2020	2020-12-28	12/27/2020
Mer 30 dic 2020	2020-12-28	12/27/2020
Gio 31 dic 2020	2020-12-28	12/27/2020
Ven 1° gen 2021	2020-12-28	12/27/2020
Sab 2 gen 2021	2020-12-28	12/27/2020
Dom 3 gen 2021	2020-12-28	1/3/2021
Lun 4 gen 2021	2021-01-04	1/3/2021
Mar 5 gen 2021	2021-01-04	1/3/2021

Le settimane iniziano il lunedì nella colonna ISO e la domenica nella colonna USA.

Esempio 1 – Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `start_of_week`, che restituisce un timestamp per l'inizio della settimana in cui sono avvenute le transazioni.

Script di caricamento

```
SET FirstWeekDay=6;

Transactions:
    Load
        *,
        weekstart(date) as start_of_week,
        timestamp(weekstart(date)) as start_of_week_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
```

```
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `start_of_week`
- `start_of_week_timestamp`

Tabella dei risultati

date	start_of_week	start_of_week_timestamp
1/7/2022	01/02/2022	1/2/2022 12:00:00 AM
1/19/2022	01/16/2022	1/16/2022 12:00:00 AM
2/5/2022	01/30/2022	1/30/2022 12:00:00 AM
2/28/2022	02/27/2022	2/27/2022 12:00:00 AM
3/16/2022	03/13/2022	3/13/2022 12:00:00 AM
4/1/2022	03/27/2022	3/27/2022 12:00:00 AM
5/7/2022	05/01/2022	5/1/2022 12:00:00 AM
5/16/2022	05/15/2022	5/15/2022 12:00:00 AM
6/15/2022	06/12/2022	6/12/2022 12:00:00 AM
6/26/2022	06/26/2022	6/26/2022 12:00:00 AM
7/9/2022	07/03/2022	7/3/2022 12:00:00 AM
7/22/2022	07/17/2022	7/17/2022 12:00:00 AM
7/23/2022	07/17/2022	7/17/2022 12:00:00 AM
7/27/2022	07/24/2022	7/24/2022 12:00:00 AM
8/2/2022	07/31/2022	7/31/2022 12:00:00 AM
8/8/2022	08/07/2022	8/7/2022 12:00:00 AM
8/19/2022	08/14/2022	8/14/2022 12:00:00 AM
9/26/2022	09/25/2022	9/25/2022 12:00:00 AM
10/14/2022	10/09/2022	10/9/2022 12:00:00 AM
10/29/2022	10/23/2022	10/23/2022 12:00:00 AM

Il campo `start_of_week` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `weekstart()` e trasferendo il campo data come argomento della funzione.

La funzione `weekstart()` identifica inizialmente in quale settimana cade il valore della data, restituendo un timestamp per il primo millisecondo di quella settimana.

Schema della funzione `weekstart()`, esempio senza argomenti aggiuntivi

La transazione 8191 è avvenuta il 5 febbraio. La variabile di sistema `FirstWeekDay` imposta il primo giorno della settimana come domenica. La funzione `weekstart()` identifica che la prima domenica prima del 5 febbraio - e quindi l'inizio della settimana - è stata il 30 gennaio, Pertanto, il valore `start_of_week` per questa transazione restituisce il primo millisecondo di quel giorno, che è il 30 gennaio alle 12:00:00 AM.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `previous_week_start`, che restituisce il timestamp per l'inizio del trimestre prima che fosse effettuata la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        weekstart(date,-1) as previous_week_start,
        timestamp(weekstart(date,-1)) as previous_week_start_timestamp
        ;
Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
```

```
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `previous_week_start`
- `previous_week_start_timestamp`

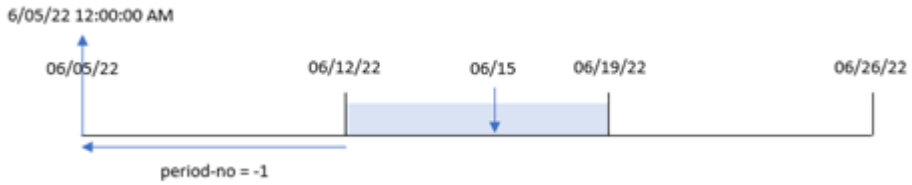
Tabella dei risultati

date	previous_week_start	previous_week_start_timestamp
1/7/2022	12/26/2021	12/26/2021 12:00:00 AM
1/19/2022	01/09/2022	1/9/2022 12:00:00 AM
2/5/2022	01/23/2022	1/23/2022 12:00:00 AM
2/28/2022	02/20/2022	2/20/2022 12:00:00 AM
3/16/2022	03/06/2022	3/6/2022 12:00:00 AM
4/1/2022	03/20/2022	3/20/2022 12:00:00 AM
5/7/2022	04/24/2022	4/24/2022 12:00:00 AM
5/16/2022	05/08/2022	5/8/2022 12:00:00 AM
6/15/2022	06/05/2022	6/5/2022 12:00:00 AM
6/26/2022	06/19/2022	6/19/2022 12:00:00 AM
7/9/2022	06/26/2022	6/26/2022 12:00:00 AM
7/22/2022	07/10/2022	7/10/2022 12:00:00 AM
7/23/2022	07/10/2022	7/10/2022 12:00:00 AM
7/27/2022	07/17/2022	7/17/2022 12:00:00 AM
8/2/2022	07/24/2022	7/24/2022 12:00:00 AM

date	previous_week_start	previous_week_start_timestamp
8/8/2022	07/31/2022	7/31/2022 12:00:00 AM
8/19/2022	08/07/2022	8/7/2022 12:00:00 AM
9/26/2022	09/18/2022	9/18/2022 12:00:00 AM
10/14/2022	10/02/2022	10/2/2022 12:00:00 AM
10/29/2022	10/16/2022	10/16/2022 12:00:00 AM

In questo caso, poiché il valore `period_no` di -1 è stato utilizzato come argomento offset nella funzione `weekstart()`, la funzione per prima cosa identifica la settimana in cui avvengono le transazioni. Quindi, cerca la settimana precedente e identifica il primo millisecondo di quella settimana.

Schema della funzione `weekstart()`, esempio di period_no



La transazione 8196 è avvenuta il 15 giugno. La funzione `weekstart()` identifica che la settimana è iniziata il 12 giugno. Pertanto, la settimana precedente è iniziata il 5 giugno alle 12:00:00; questo è il valore che viene restituito per il campo `previous_week_start`.

Esempio 3 – first_week_day

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento utilizza lo stesso set di dati e lo stesso scenario del primo esempio. Tuttavia, in questo esempio, dobbiamo impostare il martedì come primo giorno della settimana lavorativa.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        weekstart(date,0,1) as start_of_week,
        timestamp(weekstart(date,0,1)) as start_of_week_timestamp
        ;
Load
*
```

```
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `start_of_week`
- `start_of_week_timestamp`

Tabella dei risultati

date	start_of_week	start_of_week_timestamp
1/7/2022	01/04/2022	1/4/2022 12:00:00 AM
1/19/2022	01/18/2022	1/18/2022 12:00:00 AM
2/5/2022	02/01/2022	2/1/2022 12:00:00 AM
2/28/2022	02/22/2022	2/22/2022 12:00:00 AM
3/16/2022	03/15/2022	3/15/2022 12:00:00 AM
4/1/2022	03/29/2022	3/29/2022 12:00:00 AM
5/7/2022	05/03/2022	5/3/2022 12:00:00 AM
5/16/2022	05/10/2022	5/10/2022 12:00:00 AM
6/15/2022	06/14/2022	6/14/2022 12:00:00 AM

date	start_of_week	start_of_week_timestamp
6/26/2022	06/21/2022	6/21/2022 12:00:00 AM
7/9/2022	07/05/2022	7/5/2022 12:00:00 AM
7/22/2022	07/19/2022	7/19/2022 12:00:00 AM
7/23/2022	07/19/2022	7/19/2022 12:00:00 AM
7/27/2022	07/26/2022	7/26/2022 12:00:00 AM
8/2/2022	08/02/2022	8/2/2022 12:00:00 AM
8/8/2022	08/02/2022	8/2/2022 12:00:00 AM
8/19/2022	08/16/2022	8/16/2022 12:00:00 AM
9/26/2022	09/20/2022	9/20/2022 12:00:00 AM
10/14/2022	10/11/2022	10/11/2022 12:00:00 AM
10/29/2022	10/25/2022	10/25/2022 12:00:00 AM

In questa istanza, dato che l'argomento `first_week_date` di 1 è utilizzato nella funzione `weekstart()`, imposta il primo giorno della settimana a martedì.

Schema della funzione `weekstart()`, esempio first_week_day

La transazione 8191 è avvenuta il 5 febbraio. La funzione `weekstart()` identifica che il primo martedì precedente a questa data - e quindi l'inizio della settimana e il valore restituito - è stato il 1° febbraio alle 12:00:00.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che restituisce un timestamp per l'inizio della settimana in cui sono avvenute le transazioni viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:

Load
*
Inline
[
id,date,amount
8188,1/7/2022,17.17
8189,1/19/2022,37.23
8190,2/28/2022,88.27
8191,2/5/2022,57.42
8192,3/16/2022,53.80
8193,4/1/2022,82.06
8194,5/7/2022,40.39
8195,5/16/2022,87.21
8196,6/15/2022,95.93
8197,6/26/2022,45.89
8198,7/9/2022,36.23
8199,7/22/2022,25.66
8200,7/23/2022,82.77
8201,7/27/2022,69.98
8202,8/2/2022,76.11
8203,8/8/2022,25.12
8204,8/19/2022,46.23
8205,9/26/2022,84.21
8206,10/14/2022,96.24
8207,10/29/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Per calcolare l'inizio della settimana in cui avviene una transazione, sommare le seguenti misure:

- `=weekstart(date)`
- `=timestamp(weekstart(date))`

Tabella dei risultati

date	start_of_week	start_of_week_timestamp
1/7/2022	01/02/2022	1/2/2022 12:00:00 AM
1/19/2022	01/16/2022	1/16/2022 12:00:00 AM
2/5/2022	01/30/2022	1/30/2022 12:00:00 AM
2/28/2022	02/27/2022	2/27/2022 12:00:00 AM
3/16/2022	03/13/2022	3/13/2022 12:00:00 AM

date	start_of_week	start_of_week_timestamp
4/1/2022	03/27/2022	3/27/2022 12:00:00 AM
5/7/2022	05/01/2022	5/1/2022 12:00:00 AM
5/16/2022	05/15/2022	5/15/2022 12:00:00 AM
6/15/2022	06/12/2022	6/12/2022 12:00:00 AM
6/26/2022	06/26/2022	6/26/2022 12:00:00 AM
7/9/2022	07/03/2022	7/3/2022 12:00:00 AM
7/22/2022	07/17/2022	7/17/2022 12:00:00 AM
7/23/2022	07/17/2022	7/17/2022 12:00:00 AM
7/27/2022	07/24/2022	7/24/2022 12:00:00 AM
8/2/2022	07/31/2022	7/31/2022 12:00:00 AM
8/8/2022	08/07/2022	8/7/2022 12:00:00 AM
8/19/2022	08/14/2022	8/14/2022 12:00:00 AM
9/26/2022	09/25/2022	9/25/2022 12:00:00 AM
10/14/2022	10/09/2022	10/9/2022 12:00:00 AM
10/29/2022	10/23/2022	10/23/2022 12:00:00 AM

La misura `start_of_week` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `weekstart()` e trasferendo il campo `date` come argomento della funzione.

La funzione `weekstart()` identifica inizialmente in quale settimana cade il valore della data, restituendo un timestamp per il primo millisecondo di quella settimana.

Schema della funzione `weekstart()`, esempio di oggetto grafico

La transazione 8191 è avvenuta il 5 febbraio. La variabile di sistema `FirstWeekDay` imposta il primo giorno della settimana come domenica. La funzione `weekstart()` identifica che la prima domenica prima del 5 febbraio - e quindi l'inizio della settimana - era il 30 gennaio. Pertanto, il valore `start_of_week` per questa transazione restituisce il primo millisecondo di quel giorno, ovvero il 30 gennaio alle 12:00:00.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati che viene caricato in una tabella chiamata `Payroll`.
- Dati costituiti da ID dei dipendenti, nomi dei dipendenti e salario giornaliero percepito da ciascun dipendente.

I dipendenti iniziano a lavorare il lunedì e lavorano sei giorni alla settimana. La variabile di sistema `FirstWeekDay` non deve essere modificata.

L'utente finale desidera un oggetto grafico che visualizzi, in base all'ID e al nome del dipendente, i salari percepiti nella settimana fino a quel momento.

Script di caricamento

```
Payroll:
Load
*
Inline
[
employee_id,employee_name,day_rate
182,Mark, $150
183,Deryck, $125
184,Dexter, $125
185,Sydney,$270
186,Agatha,$128
];
```

Risultati

Procedere come indicato di seguito:

1. Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:
    - `employee_id`
    - `employee_name`
2. Quindi, creare una misura per calcolare i salari guadagnati nella settimana in corso: `=if(today(1)-weekstart(today(1),0,0)<7,(today(1)-weekstart(today(1),0,0))*day_rate,day_rate*6)`
3. Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

employee_id	employee_ name	=if(today(1)-weekstart(today(1),0,0)<7,(today(1)-weekstart (today(1),0,0))*day_rate,day_rate*6)
182	Contrassegno	$600.00
183	Deryck	$500.00
184	Dexter	$500.00
185	Sydney	$1080.00
186	Agatha	$512.00

La funzione `weekstart()`, utilizzando la data di oggi come primo argomento e 0 come terzo argomento, imposta il lunedì come primo giorno della settimana e restituisce la data di inizio della settimana corrente. Sottraendo tale risultato dalla data corrente, l'espressione restituisce il numero di giorni trascorsi fino ad ora questa settimana.

La condizione valuta quindi se ci sono stati più di sei giorni nella settimana. In caso di risposta affermativa, il valore `day_rate` del dipendente viene moltiplicato per 6. Altrimenti, il valore `day_rate` viene moltiplicato per il numero di giorni che si sono verificati finora nella settimana.

weekyear

Questa funzione restituisce l'anno a cui appartiene il numero della settimana in base alle variabili di ambiente. I numeri della settimana rientrano in un intervallo approssimativo compreso tra 1 e 52.

Sintassi:

```
weekyear(timestamp [, first_week_day [, broken_weeks [, reference_day]]])
```

Tipo di dati restituiti: numero intero

Argomenti

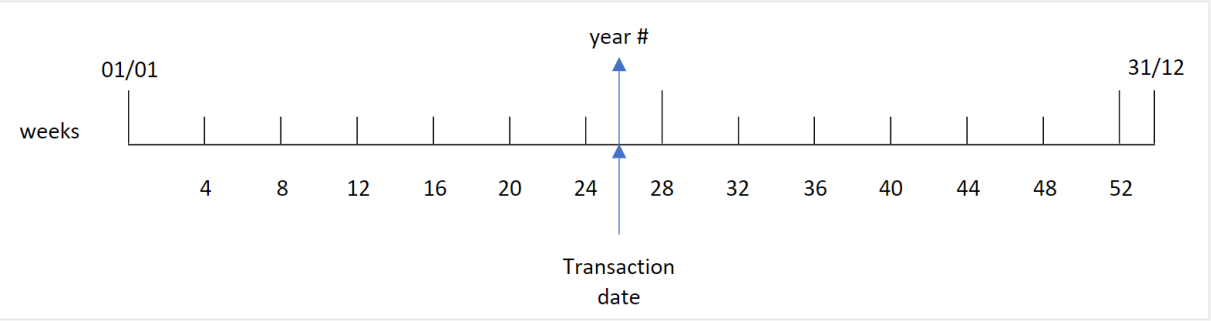
Argomento	Descrizione
timestamp	La data o la data e ora da valutare.
first_week_ day	Specifica il giorno di inizio della settimana. Se omesso, viene utilizzato il valore della variabile **FirstWeekDay**. I valori possibili per **first_week_day** sono 0 per lunedì, 1 per martedì, 2 per mercoledì, 3 per giovedì, 4 per venerdì, 5 per sabato e 6 per domenica. Per maggiori informazioni sulle variabili di sistema, vedere *FirstWeekDay (page 224)*.
broken_ weeks	Se non si specifica **broken_weeks**, il valore della variabile **BrokenWeeks** verrà utilizzato per definire se le settimane sono parziali o meno.

Argomento	Descrizione
reference_day	Se non si specifica **reference_day**, il valore della variabile **ReferenceDay** verrà utilizzato per definire quale giorno di gennaio impostare come giorno di riferimento per definire la settimana 1. Per impostazione predefinita, le funzioni Qlik Sense utilizzano 4 come giorno di riferimento. Questo significa che la settimana 1 deve contenere il 4 gennaio, vale a dire che la settimana 1 deve sempre contenere 4 quattro giorni di gennaio.

La funzione `weekyear()` determina in quale settimana dell'anno cade una data. Quindi restituisce l'anno corrispondente a quel numero di settimana.

Se `BrokenWeeks` è impostato su `0` (false), `weekyear()` restituirà lo stesso come `year()`.

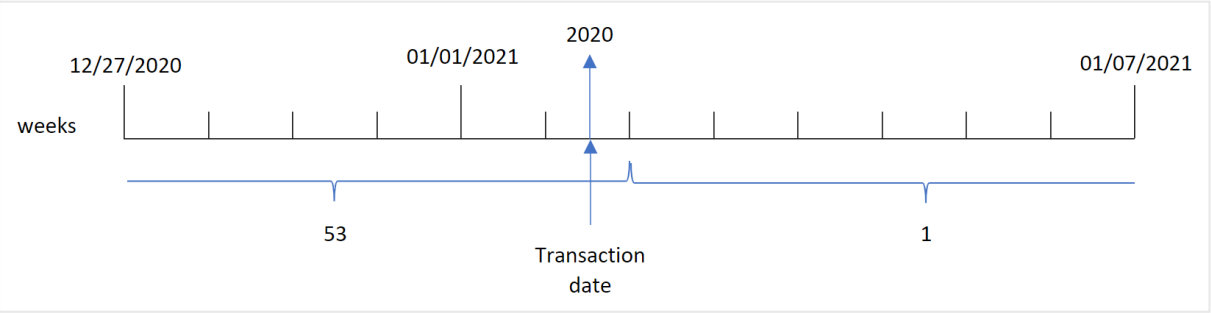
Schema dell'intervallo della funzione `weekyear()`



Tuttavia, se la variabile di sistema `BrokenWeeks` è impostata per l'utilizzo di settimane intere, la settimana 1 deve contenere solo un certo numero di giorni di gennaio, in base al valore specificato nella variabile di sistema `ReferenceDay`.

Ad esempio, se si utilizza un valore `ReferenceDay` pari a `4`, la settimana 1 deve includere almeno quattro giorni di gennaio. È possibile che la settimana 1 includa date del dicembre dell'anno precedente o che il numero della settimana finale di un anno includa date del gennaio dell'anno successivo. In situazioni come questa, la funzione `weekyear()` restituirà un valore diverso per la funzione `year()`.

Schema dell'intervallo della funzione `weekyear()` quando si utilizzano le settimane intere



Casi di utilizzo

La funzione `weekyear()` è utile quando si desidera confrontare le aggregazioni per anni. Ad esempio, può essere usata se si desidera visualizzare le vendite totali dei prodotti in base all'anno. La funzione `weekyear()` viene scelta rispetto a `year()` quando l'utente desidera mantenere la coerenza con la variabile di sistema `BrokenWeeks` nell'app.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`weekyear ('12/30/1996',0,0,4)`	Restituisce 1997, poiché la settimana 1 del 1997 inizia il 30/12/1996
`weekyear ('01/02/1997',0,0,4)`	Restituisce 1997
`weekyear ('12/28/1997',0,0,4)`	Restituisce 1997
`weekyear ('12/30/1997',0,0,4)`	Restituisce il 1998, perché la settimana 1 del 1998 inizia il 29/12/1997
`weekyear ('01/02/1999',0,0,4)`	Restituisce il 1998, perché la settimana 53 del 1998 termina il 03/01/1999

Argomenti correlati

Argomento	Interazione
week (page 1047)	Restituisce un numero intero che rappresenta il numero della settimana in base allo standard ISO 8601
year (page 1122)	Restituisce un numero intero che rappresenta l'anno in cui l'espressione viene interpretata come data in base all'interpretazione numerica standard.

Esempio 1 - Settimane parziali

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per l'ultima settimana del 2020 e la prima settimana del 2021 che viene caricato in una tabella chiamata '`Transactions`'.
- La variabile di sistema `BrokenWeeks` che è impostata su 1.
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
    - La funzione `weekyear()`, impostata come campo '`week_year`', che restituisce l'anno in cui sono avvenute le transazioni.
    - La funzione `week()`, impostata come campo '`week`', che mostra il numero di settimana di ogni data di transazione.

Script di caricamento

```
SET BrokenWeeks=1;

Transactions:
       Load
       *,
       week(date) as week,
       weekyear(date) as week_year
       ;
Load
*
Inline
[
id,date,amount
8176,12/28/2020,19.42
8177,12/29/2020,23.80
8178,12/30/2020,82.06
8179,12/31/2020,40.56
8180,01/01/2021,37.23
8181,01/02/2021,17.17
8182,01/03/2021,88.27
8183,01/04/2021,57.42
8184,01/05/2021,67.42
8185,01/06/2021,23.80
8186,01/07/2021,82.06
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- date
- week
- week_year

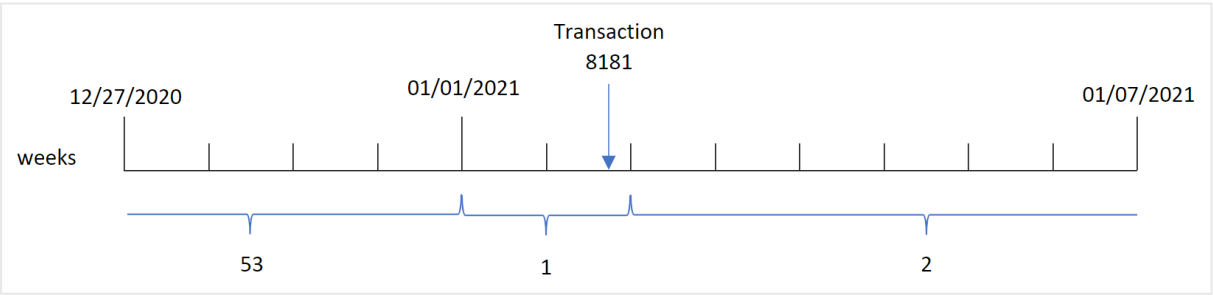
Tabella dei risultati

id	date	settimana	week_year
8176	12/28/2020	53	2020
8177	12/29/2020	53	2020
8178	12/30/2020	53	2020
8179	12/31/2020	53	2020
8180	01/01/2021	1	2021
8181	01/02/2021	1	2021
8182	01/03/2021	2	2021
8183	01/04/2021	2	2021
8184	01/05/2021	2	2021
8185	01/06/2021	2	2021
8186	01/07/2021	2	2021

Il campo 'week_year' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione weekyear() e trasferendo il campo data come argomento della funzione.

La variabile di sistema BrokenWeeks è impostata su 1, il che significa che l'app utilizza settimane parziali. La settimana 1 inizia il 1° gennaio.

Schema dell'intervallo della funzione weekyear() con l'uso di settimane parziali



La transazione 8181 ha luogo il 2 gennaio, che fa parte della settimana 1. Pertanto, restituisce un valore di 2021 per il campo 'week_year'.

Esempio 2 - Settimane intere

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per l'ultima settimana del 2020 e la prima settimana del 2021 che viene caricato in una tabella chiamata '`Transactions`'.
- La variabile di sistema `BrokenWeeks` che è impostata su `0`.
- Un'istruzione LOAD precedente che contiene i seguenti elementi:
    - La funzione `weekyear()`, impostata come campo '`week_year`', che restituisce l'anno in cui sono avvenute le transazioni.
    - La funzione `week()`, impostata come campo '`week`', che mostra il numero di settimana di ogni data di transazione.

Tuttavia, in questo esempio, la politica aziendale prevede l'utilizzo di settimane intere.

Script di caricamento

```
SET BrokenWeeks=0;

Transactions:
       Load
       *,
       week(date) as week,
       weekyear(date) as week_year
       ;
Load
*
Inline
[
id,date,amount
8176,12/28/2020,19.42
8177,12/29/2020,23.80
8178,12/30/2020,82.06
8179,12/31/2020,40.56
8180,01/01/2021,37.23
8181,01/02/2021,17.17
8182,01/03/2021,88.27
8183,01/04/2021,57.42
8184,01/05/2021,67.42
8185,01/06/2021,23.80
8186,01/07/2021,82.06
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `week`
- `week_year`

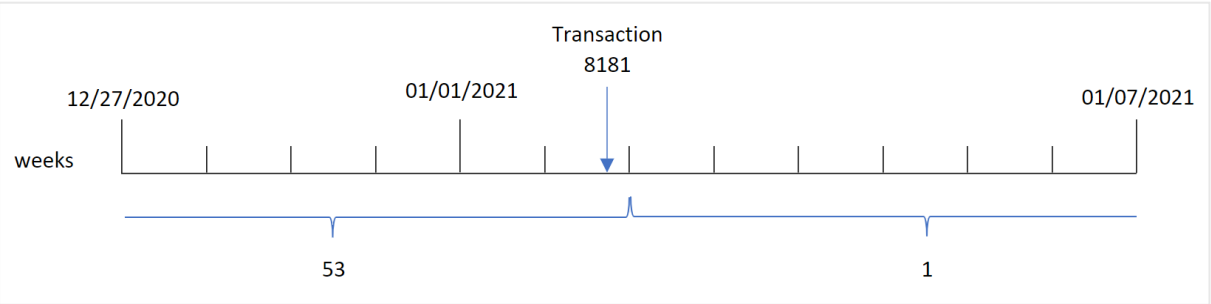
Tabella dei risultati

id	date	settimana	week_year
8176	12/28/2020	53	2020
8177	12/29/2020	53	2020
8178	12/30/2020	53	2020
8179	12/31/2020	53	2020
8180	01/01/2021	53	2020
8181	01/02/2021	53	2020
8182	01/03/2021	1	2021
8183	01/04/2021	1	2021
8184	01/05/2021	1	2021
8185	01/06/2021	1	2021
8186	01/07/2021	1	2021

La variabile di sistema `BrokenWeeks` è impostata su `0`, il che significa che l'applicazione utilizza settimane intere. Pertanto, non è necessario che la settimana 1 inizi il 1° gennaio.

La settimana 53 del 2020 continua fino alla fine del 2 gennaio 2021, mentre la settimana 1 del 2020 inizia domenica 3 gennaio 2021.

Schema dell'intervallo della funzione `weekyear()` con l'uso di settimane intere



La transazione 8181 ha luogo il 2 gennaio, che fa parte della settimana 1. Pertanto, restituisce un valore di 2021 per il campo '`week_year`'.

Esempio 3 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che restituisce il numero della settimana dell'anno in cui sono avvenute le transazioni viene creato come misura in un grafico dell'app.

Script di caricamento

```
SET BrokenWeeks=1;

Transactions:
Load
*
Inline
[
id,date,amount
8176,12/28/2020,19.42
8177,12/29/2020,23.80
8178,12/30/2020,82.06
8179,12/31/2020,40.56
8180,01/01/2021,37.23
8181,01/02/2021,17.17
8182,01/03/2021,88.27
8183,01/04/2021,57.42
8184,01/05/2021,67.42
8185,01/06/2021,23.80
8186,01/07/2021,82.06
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`

Per calcolare la settimana in cui avviene una transazione, creare la seguente misura:

- `=week(date)`

Per calcolare l'anno in cui avviene una transazione in base al numero della settimana, creare la seguente misura:

- `=weekyear(date)`

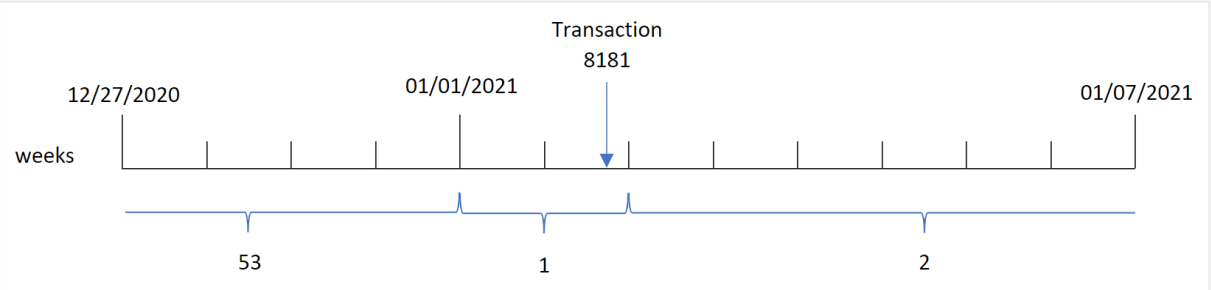
Tabella dei risultati

id	date	settimana	week_year
8176	12/28/2020	53	2020
8177	12/29/2020	53	2020
8178	12/30/2020	53	2020
8179	12/31/2020	53	2020
8180	01/01/2021	1	2021
8181	01/02/2021	1	2021
8182	01/03/2021	2	2021
8183	01/04/2021	2	2021
8184	01/05/2021	2	2021
8185	01/06/2021	2	2021
8186	01/07/2021	2	2021

Il campo 'week_year' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `weekyear()` e trasferendo il campo data come argomento della funzione.

La variabile di sistema `BrokenWeeks` è impostata su `1`, il che significa che l'applicazione utilizza settimane parziali. La settimana 1 inizia il 1° gennaio.

Schema dell'intervallo della funzione `weekyear()` con l'uso di settimane parziali



La transazione 8181 ha luogo il 2 gennaio, che fa parte della settimana 1. Pertanto, restituisce un valore di 2021 per il campo 'week_year'.

Esempio 4 - Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni per l'ultima settimana del 2020 e la prima settimana del 2021 che viene caricato in una tabella chiamata '`Transactions`'.
- La variabile di sistema `BrokenWeeks` che è impostata su `0`. Ciò significa che l'app utilizzerà settimane intere.
- La variabile di sistema `ReferenceDay` che è impostata su `2`. Ciò significa che l'anno inizierà il 2 gennaio e conterrà un minimo di due giorni a gennaio.
- La variabile di sistema `FirstWeekDay` che è impostata su `1`. Ciò significa che il primo giorno della settimana sarà il martedì.

La politica aziendale prevede l'utilizzo di settimane parziali. L'utente finale desidera un grafico che presenti le vendite totali per anno. L'app utilizza settimane intere, con la settimana 1 che contiene un minimo di due giorni a gennaio.

Script di caricamento

```
SET BrokenWeeks=0;
SET ReferenceDay=2;
SET FirstWeekDay=1;

Transactions:
Load
*
Inline
[
id,date,amount
8176,12/28/2020,19.42
8177,12/29/2020,23.80
8178,12/30/2020,82.06
8179,12/31/2020,40.56
8180,01/01/2021,37.23
8181,01/02/2021,17.17
8182,01/03/2021,88.27
8183,01/04/2021,57.42
8184,01/05/2021,67.42
8185,01/06/2021,23.80
8186,01/07/2021,82.06
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella.

Per calcolare l'anno in cui avviene una transazione in base al numero della settimana, creare la seguente misura:

- `=weekyear(date)`

Per calcolare le vendite totali, creare la seguente misura:

- `sum(amount)`

Impostare la **Formattazione numero** della misura su **Denaro**.

Tabella dei risultati

weekyear(date)	=sum(amount)
2020	19.42
2021	373.37

year

Questa funzione restituisce un numero intero che rappresenta l'anno in cui **expression** viene interpretato come data in base all'interpretazione numerica standard.

Sintassi:

```
year(expression)
```

Tipo di dati restituiti: numero intero

La funzione `year()` è disponibile sia come funzione script che come grafico. La funzione restituisce l'anno per una data particolare. È comunemente utilizzata per creare un campo anno come dimensione in un Calendario principale.

Casi di utilizzo

La funzione `year()` è utile quando si desidera confrontare le aggregazioni per anno. Ad esempio, la funzione potrebbe essere usata se si desidera visualizzare le vendite totali dei prodotti in base all'anno.

È possibile creare queste dimensioni nello script di caricamento utilizzando la funzione che consente di creare un campo in una tabella Calendario principale. In alternativa, può essere utilizzata direttamente in un grafico come dimensione calcolata.

Esempi di funzioni

Esempio	Risultato
`year( '2012-10-12' )`	restituisce 2012
`year( '35648' )`	restituisce 1997 poiché 35648 = 1997-08-06

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è

impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – set di dati DateFormat (script)

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati delle date, caricato in una tabella denominata `Master Calendar`.
- È utilizzata la variabile di sistema predefinita `DateFormat` (MM/GG/AAAA).
- Un caricamento precedente, utilizzato per creare un campo aggiuntivo, denominato `year`, mediante la funzione `year()`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Master_Calendar:

    Load
        date,
        year(date) as year
        ;
Load
date
Inline
[
date
12/28/2020
12/29/2020
12/30/2020
12/31/2020
01/01/2021
01/02/2021
01/03/2021
01/04/2021
01/05/2021
01/06/2021
01/07/2021
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- date
- year

Tabella dei risultati

date	anno
12/28/2020	2020
12/29/2020	2020
12/30/2020	2020
12/31/2020	2020
01/01/2021	2021
01/02/2021	2021
01/03/2021	2021
01/04/2021	2021
01/05/2021	2021
01/06/2021	2021
01/07/2021	2021

Esempio 2 – Date ANSI

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati delle date, caricato in una tabella denominata `Master Calendar`.
- È utilizzata la variabile di sistema predefinita `DateFormat` (MM/GG/AAAA). Tuttavia, le date incluse nel set di dati sono nel formato data standard ANSI.
- Un caricamento precedente, utilizzato per creare un campo aggiuntivo, denominato `year`, mediante la funzione `year()`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Master_Calendar:
    Load
        date,
        year(date) as year
        ;
Load
```

```
date
Inline
[
date
2020-12-28
2020-12-29
2020-12-30
2020-12-31
2021-01-01
2021-01-02
2021-01-03
2021-01-04
2021-01-05
2021-01-06
2021-01-07
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `year`

Tabella dei risultati

date	anno
2020-12-28	2020
2020-12-29	2020
2020-12-30	2020
2020-12-31	2020
2021-01-01	2021
2021-01-02	2021
2021-01-03	2021
2021-01-04	2021
2021-01-05	2021
2021-01-06	2021
2021-01-07	2021

Esempio 3 – Date non formattate

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati delle date in formato numerico, caricato in una tabella denominata `Master Calendar`.
- È utilizzata la variabile di sistema predefinita `DateFormat` (MM/GG/AAAA).
- Un caricamento precedente, utilizzato per creare un campo aggiuntivo, denominato `year`, mediante la funzione `year()`.

Viene caricata la data originale non formattata, denominata `unformatted_date`, e per maggiore chiarezza, viene utilizzato un ulteriore campo aggiuntivo, denominato `long_date`, per convertire la data numerica in un campo data formattato utilizzando la funzione `date()`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Master_Calendar:
    Load
        unformatted_date,
        date(unformatted_date) as long_date,
        year(unformatted_date) as year
        ;
Load
unformatted_date
Inline
[
unformatted_date
44868
44898
44928
44958
44988
45018
45048
45078
45008
45038
45068
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- unformatted_date
- long_date
- year

Tabella dei risultati

unformatted_date	long_date	anno
44868	11/03/2022	2022
44898	12/03/2022	2022
44928	01/02/2023	2023
44958	02/01/2023	2023
44988	03/03/2023	2023
45008	03/23/2023	2023
45018	04/02/2023	2023
45038	04/22/2023	2023
45048	05/02/2023	2023
45068	05/22/2023	2023
45078	06/01/2023	2023

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

In questo esempio, un insieme di dati relativi agli ordini effettuati viene caricato in una tabella denominata Vendite. La tabella contiene tre campi:

- id
- sales_date
- amount

Le garanzie sulle vendite dei prodotti durano due anni dalla data di vendita. L'attività consiste nel creare una misura in un grafico per determinare l'anno in cui scadrà ogni garanzia.

Script di caricamento

```
Sales:
Load
id,
sales_date,
amount
```

```
Inline
[
id,sales_date,amount
1,12/28/2020,231.24,
2,12/29/2020,567.28,
3,12/30/2020,364.28,
4,12/31/2020,575.76,
5,01/01/2021,638.68,
6,01/02/2021,785.38,
7,01/03/2021,967.46,
8,01/04/2021,287.67
9,01/05/2021,764.45,
10,01/06/2021,875.43,
11,01/07/2021,957.35
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `sales_date`.

Creare la seguente misura:

```
=year(sales_date+365*2)
```

Tabella dei risultati

sales_date	=year(sales_date+365*2)
12/28/2020	2022
12/29/2020	2022
12/30/2020	2022
12/31/2020	2022
01/01/2021	2023
01/02/2021	2023
01/03/2021	2023
01/04/2021	2023
01/05/2021	2023
01/06/2021	2023
01/07/2021	2023

I risultati di questa misura sono riportati nella tabella precedente. Per aggiungere due anni a una data, moltiplicare 365 per 2 e aggiungere il risultato alla data di vendita. Pertanto, le vendite avvenute nel 2020 hanno come anno di scadenza il 2022.

yearend

Questa funzione restituisce un valore corrispondente a un indicatore temporale recante l'ultimo millisecondo dell'ultimo giorno dell'anno contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

Sintassi:

```
YearEnd( date[, period_no[, first_month_of_year = 1]])
```

In altre parole, la funzione `yearend()` determina in quale anno cade la data. Quindi, restituisce data e ora, nel formato data, per l'ultimo millisecondo di quell'anno. Il primo mese dell'anno è, per impostazione predefinita, gennaio. Tuttavia, è possibile modificare il mese da impostare come primo utilizzando l'argomento `first_month_of_year` nella funzione `yearend()`.

> *La funzione `yearend()` non considera la variabile di sistema `FirstMonthofYear`. L'anno inizia il 1 gennaio a condizione che l'argomento `first_month_of_year` non venga utilizzato per modificarlo.*

Schema della funzione `yearend()`.

Casi di utilizzo

La funzione `yearend()` viene utilizzata come parte di un'espressione quando si desidera che il calcolo utilizzi la frazione dell'anno non ancora trascorso. Ad esempio, se si vuole calcolare l'interesse totale non ancora maturato durante l'anno.

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero, in cui il valore 0 indica l'anno che contiene **date**. I valori negativi di **period_no** indicano gli anni precedenti, mentre i valori positivi indicano gli anni successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

È possibile utilizzare i seguenti valori per impostare il primo mese dell'anno nell'argomento `first_month_of_year`:

valori first_month_of_year

Mese	Valore
Febbraio	2
March	3
Aprile	4
May	5
Giugno	6
Luglio	7
Agosto	8
Settembre	9
Ottobre	10
Novembre	11
Dicembre	12

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`yearend('10/19/2001')`	Returns 12/31/2001 23:59:59.
`yearend('10/19/2001', -1)`	Returns 12/31/2000 23:59:59.
`yearend('10/19/2001', 0, 4)`	Returns 03/31/2002 23:59:59.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022 viene caricato in una tabella denominata `Transactions`.
- Il campo della data è stato fornito nel formato della variabile di sistema `DateFormat`, ossia `(MM/DD/YYYY)`.
- Un'istruzione LOAD precedente che contiene quanto segue:
  - `yearend()` funzione che è impostata come campo `year_end`.
  - `Timestamp()` funzione che è impostata come campo `year_end_timestamp`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       yearend(date) as year_end,
       timestamp(yearend(date)) as year_end_timestamp
       ;
Load
*
Inline
[
id,date,amount
8188,01/13/2020,37.23
8189,02/26/2020,17.17
8190,03/27/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,06/06/2022,46.23
8205,07/18/2022,84.21
8206,11/14/2022,96.24
```

```
8207,12/12/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- date
- year_end
- year_end_timestamp

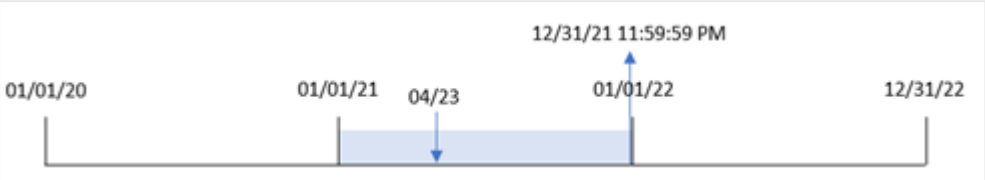
Tabella dei risultati

id	data	year_end	year_end_timestamp
8188	01/13/2020	12/31/2020	12/31/2020 11:59:59 PM
8189	02/26/2020	12/31/2020	12/31/2020 11:59:59 PM
8190	03/27/2020	12/31/2020	12/31/2020 11:59:59 PM
8191	04/16/2020	12/31/2020	12/31/2020 11:59:59 PM
8192	05/21/2020	12/31/2020	12/31/2020 11:59:59 PM
8193	08/14/2020	12/31/2020	12/31/2020 11:59:59 PM
8194	10/07/2020	12/31/2020	12/31/2020 11:59:59 PM
8195	12/05/2020	12/31/2020	12/31/2020 11:59:59 PM
8196	01/22/2021	12/31/2021	12/31/2021 11:59:59 PM
8197	02/03/2021	12/31/2021	12/31/2021 11:59:59 PM
8198	03/17/2021	12/31/2021	12/31/2021 11:59:59 PM
8199	04/23/2021	12/31/2021	12/31/2021 11:59:59 PM
8200	05/04/2021	12/31/2021	12/31/2021 11:59:59 PM
8201	06/30/2021	12/31/2021	12/31/2021 11:59:59 PM
8202	07/26/2021	12/31/2021	12/31/2021 11:59:59 PM
8203	12/27/2021	12/31/2021	12/31/2021 11:59:59 PM
8204	06/06/2022	12/31/2022	12/31/2022 11:59:59 PM
8205	07/18/2022	12/31/2022	12/31/2022 11:59:59 PM
8206	11/14/2022	12/31/2022	12/31/2022 11:59:59 PM
8207	12/12/2022	12/31/2022	12/31/2022 11:59:59 PM

Il campo 'year_end' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `yearend()` e trasferendo il campo data come argomento della funzione.

La funzione `yearend()` inizialmente identifica in quale anno rientra il valore della data e restituisce data e ora per l'ultimo millisecondo di quell'anno.

Diagramma della funzione `yearend()` con la transazione 8199 selezionata.



La transazione 8199 è avvenuta il 23 aprile 2021. La funzione `yearend()` restituisce l'ultimo millisecondo di quell'anno, ovvero il 31 dicembre alle ore 11:59:59 PM.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, l'attività consiste nel creare un campo, '`previous_year_end`', che restituisce data e ora di fine dell'anno precedente all'anno in cui ha avuto luogo la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       yearend(date,-1) as previous_year_end,
       timestamp(yearend(date,-1)) as previous_year_end_timestamp
       ;
Load
*
Inline
[
id,date,amount
8188,01/13/2020,37.23
8189,02/26/2020,17.17
8190,03/27/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
```

```
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,06/06/2022,46.23
8205,07/18/2022,84.21
8206,11/14/2022,96.24
8207,12/12/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `previous_year_end`
- `previous_ year_end_timestamp`

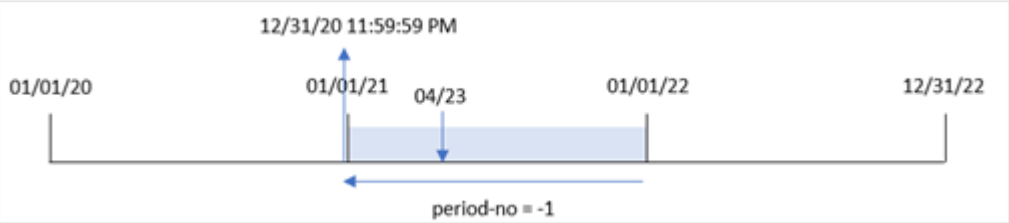
Tabella dei risultati

id	data	previous_year_end	previous_year_end_timestamp
8188	01/13/2020	12/31/2019	12/31/2019 11:59:59 PM
8189	02/26/2020	12/31/2019	12/31/2019 11:59:59 PM
8190	03/27/2020	12/31/2019	12/31/2019 11:59:59 PM
8191	04/16/2020	12/31/2019	12/31/2019 11:59:59 PM
8192	05/21/2020	12/31/2019	12/31/2019 11:59:59 PM
8193	08/14/2020	12/31/2019	12/31/2019 11:59:59 PM
8194	10/07/2020	12/31/2019	12/31/2019 11:59:59 PM
8195	12/05/2020	12/31/2019	12/31/2019 11:59:59 PM
8196	01/22/2021	12/31/2020	12/31/2020 11:59:59 PM
8197	02/03/2021	12/31/2020	12/31/2020 11:59:59 PM
8198	03/17/2021	12/31/2020	12/31/2020 11:59:59 PM
8199	04/23/2021	12/31/2020	12/31/2020 11:59:59 PM
8200	05/04/2021	12/31/2020	12/31/2020 11:59:59 PM
8201	06/30/2021	12/31/2020	12/31/2020 11:59:59 PM
8202	07/26/2021	12/31/2020	12/31/2020 11:59:59 PM
8203	12/27/2021	12/31/2020	12/31/2020 11:59:59 PM
8204	06/06/2022	12/31/2021	12/31/2021 11:59:59 PM
8205	07/18/2022	12/31/2021	12/31/2021 11:59:59 PM

id	data	previous_year_end	previous_year_end_timestamp
8206	11/14/2022	12/31/2021	12/31/2021 11:59:59 PM
8207	12/12/2022	12/31/2021	12/31/2021 11:59:59 PM

Poiché il valore `period_no` di `-1` è stato utilizzato come argomento offset nella funzione `yearend()`, la funzione per prima cosa identifica l'anno in cui avvengono le transazioni. Quindi, cerca l'anno precedente e identifica il primo millisecondo di quell'anno.

Schema della funzione `yearend()` con un valore period_no di -1.



La transazione 8199 avviene il 23 aprile 2021. La funzione `yearend()` restituisce l'ultimo millisecondo dell'anno precendente, ovvero il 31 dicembre 2020 alle ore 11:59:59 PM, per il campo '`previous_year_end`'.

Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, la politica aziendale prevede che l'anno inizi il 1 aprile.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       yearend(date,0,4) as year_end,
       timestamp(yearend(date,0,4)) as year_end_timestamp
       ;
Load
*
Inline
[
id,date,amount
8188,01/13/2020,37.23
8189,02/26/2020,17.17
8190,03/27/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
```

```
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,06/06/2022,46.23
8205,07/18/2022,84.21
8206,11/14/2022,96.24
8207,12/12/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `year_end`
- `year_end_timestamp`

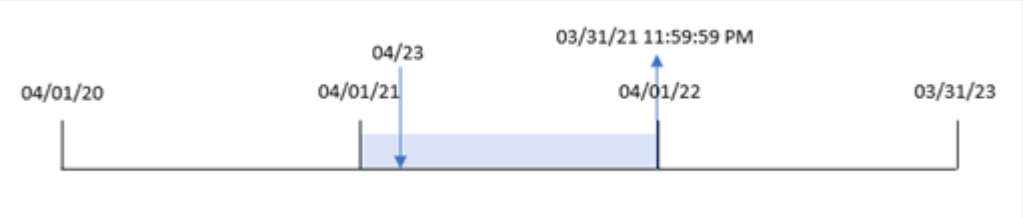
Tabella dei risultati

id	data	year_end	year_end_timestamp
8188	01/13/2020	03/31/2020	3/31/2020 11:59:59 PM
8189	02/26/2020	03/31/2020	3/31/2020 11:59:59 PM
8190	03/27/2020	03/31/2020	3/31/2020 11:59:59 PM
8191	04/16/2020	03/31/2021	3/31/2021 11:59:59 PM
8192	05/21/2020	03/31/2021	3/31/2021 11:59:59 PM
8193	08/14/2020	03/31/2021	3/31/2021 11:59:59 PM
8194	10/07/2020	03/31/2021	3/31/2021 11:59:59 PM
8195	12/05/2020	03/31/2021	3/31/2021 11:59:59 PM
8196	01/22/2021	03/31/2021	3/31/2021 11:59:59 PM
8197	02/03/2021	03/31/2021	3/31/2021 11:59:59 PM
8198	03/17/2021	03/31/2021	3/31/2021 11:59:59 PM
8199	04/23/2021	03/31/2022	3/31/2022 11:59:59 PM
8200	05/04/2021	03/31/2022	3/31/2022 11:59:59 PM

id	data	year_end	year_end_timestamp
8201	06/30/2021	03/31/2022	3/31/2022 11:59:59 PM
8202	07/26/2021	03/31/2022	3/31/2022 11:59:59 PM
8203	12/27/2021	03/31/2022	3/31/2022 11:59:59 PM
8204	06/06/2022	03/31/2023	3/31/2023 11:59:59 PM
8205	07/18/2022	03/31/2023	3/31/2023 11:59:59 PM
8206	11/14/2022	03/31/2023	3/31/2023 11:59:59 PM
8207	12/12/2022	03/31/2023	3/31/2023 11:59:59 PM

Poiché l'argomento `first_month_of_year` di `4` viene utilizzato nella funzione `yearend()`, imposta il primo giorno dell'anno come il 1 aprile e l'ultimo giorno dell'anno come il 31 marzo.

Lo schema della funzione `yearend()` con aprile come primo mese dell'anno.



La transazione 8199 avviene il 23 aprile 2021. Poiché la funzione `yearend()` imposta l'inizio dell'anno sul 1 aprile, restituisce il 31 marzo 2022 come valore '`year_end`' per la transazione.

Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che restituisce data e ora della data di fine per l'anno in cui è avvenuta una transazione viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,01/13/2020,37.23
8189,02/26/2020,17.17
8190,03/27/2020,88.27
```

```
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,06/06/2022,46.23
8205,07/18/2022,84.21
8206,11/14/2022,96.24
8207,12/12/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`

Per calcolare in quale anno è avvenuta una transazione, creare le seguenti misure:

- `=yearend(date)`
- `=timestamp(yearend(date))`

Tabella dei risultati

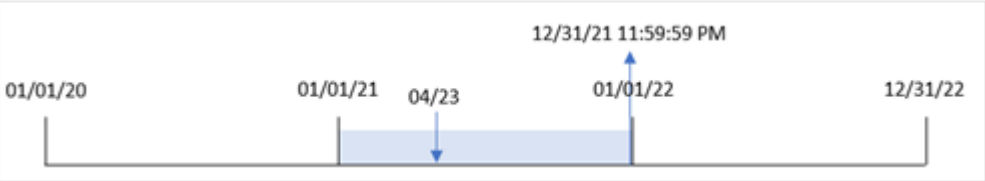
id	data	=yearend(date)	=timestamp(yearend(date))
8188	01/13/2020	12/31/2020	12/31/2020 11:59:59 PM
8189	02/26/2020	12/31/2020	12/31/2020 11:59:59 PM
8190	03/27/2020	12/31/2020	12/31/2020 11:59:59 PM
8191	04/16/2020	12/31/2020	12/31/2020 11:59:59 PM
8192	05/21/2020	12/31/2020	12/31/2020 11:59:59 PM
8193	08/14/2020	12/31/2020	12/31/2020 11:59:59 PM
8194	10/07/2020	12/31/2020	12/31/2020 11:59:59 PM
8195	12/05/2020	12/31/2020	12/31/2020 11:59:59 PM
8196	01/22/2021	12/31/2021	12/31/2021 11:59:59 PM
8197	02/03/2021	12/31/2021	12/31/2021 11:59:59 PM
8198	03/17/2021	12/31/2021	12/31/2021 11:59:59 PM

id	data	=yearend(date)	=timestamp(yearend(date))
8199	04/23/2021	12/31/2021	12/31/2021 11:59:59 PM
8200	05/04/2021	12/31/2021	12/31/2021 11:59:59 PM
8201	06/30/2021	12/31/2021	12/31/2021 11:59:59 PM
8202	07/26/2021	12/31/2021	12/31/2021 11:59:59 PM
8203	12/27/2021	12/31/2021	12/31/2021 11:59:59 PM
8204	06/06/2022	12/31/2022	12/31/2022 11:59:59 PM
8205	07/18/2022	12/31/2022	12/31/2022 11:59:59 PM
8206	11/14/2022	12/31/2022	12/31/2022 11:59:59 PM
8207	12/12/2022	12/31/2022	12/31/2022 11:59:59 PM

La misura '`end_of_year`' viene creata nell'oggetto grafico mediante l'utilizzo della funzione `yearend()` e trasferendo il campo data come argomento della funzione.

La funzione `yearend()` inizialmente identifica in quale anno rientra il valore della data, restituendo data e ora per l'ultimo millisecondo di quell'anno.

Schema della funzione `yearend()` che mostra la transazione 8199 avvenuta ad aprile.



La transazione 8199 avviene il 23 aprile 2021. La funzione `yearend()` restituisce l'ultimo millisecondo di quell'anno, ovvero il 31 dicembre alle ore 11:59:59 PM

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata '`Employee_Expenses`'. La tabella contiene i seguenti campi:
  - ID dipendenti
  - nome dipendenti
  - media delle richieste di rimborso spese giornaliero di ciascun dipendente

L'utente finale desidera un oggetto grafico che visualizzi, in base all'ID dipendente e al nome del dipendente, le richieste di rimborso spese stimate ancora da sostenere per il resto dell'anno. L'esercizio finanziario inizia a gennaio.

Script di caricamento

```
Employee_Expenses:
Load
*
Inline
[
employee_id,employee_name,avg_daily_claim
182,Mark, $15
183,Deryck, $12.5
184,Dexter, $12.5
185,Sydney,$27
186,Agatha,$18
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `employee_id`
- `employee_name`

Per calcolare una proiezione delle richieste di rimborso spese, creare la seguente misura:

```
=(yearend(today(1))-today(1))*avg_daily_claim
```

Impostare la misura **Number Formatting** su **Money**.

Tabella dei risultati

employee_id	employee_name	=(yearend(today(1))-today(1))*avg_daily_claim
182	Contrassegno	$3240.00
183	Deryck	$2700.00
184	Dexter	$2700.00
185	Sydney	$5832.00
186	Agatha	$3888.00

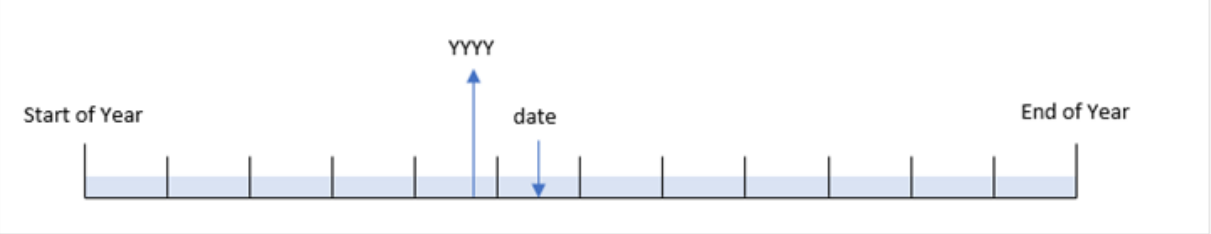
Utilizzando la data odierna come unico argomento, la funzione `yearend()` restituisce la data di fine dell'anno corrente. Quindi, sottraendo la data odierna dalla data di fine dell'anno, l'espressione restituisce il numero di giorni rimanenti per l'anno corrente.

Questo valore viene quindi moltiplicato per la media delle richieste di rimborso spese giornaliere di ciascun dipendente per calcolare il valore stimato delle richieste che ogni dipendente dovrebbe presentare durante il periodo rimanente dell'anno.

yearname

Questa funzione restituisce un anno di quattro cifre come valore di visualizzazione con un valore numerico sottostante corrispondente a un indicatore temporale recante il primo millisecondo del primo giorno dell'anno contenente **date**.

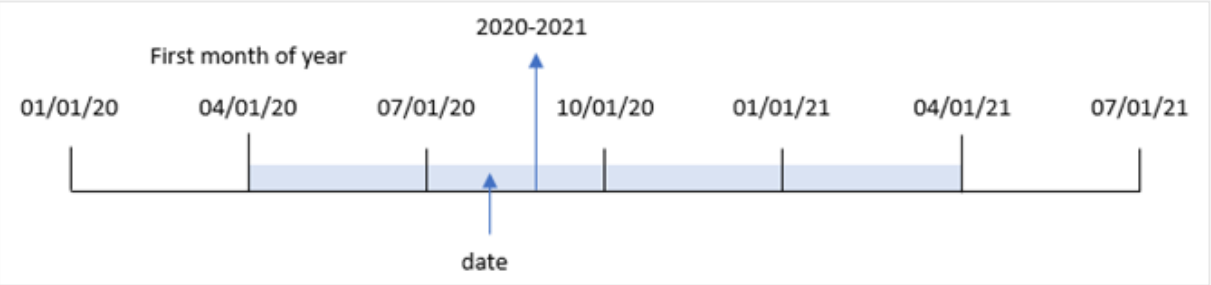
Diagramma dell'intervallo di tempo della funzione `yearname()`.



La funzione `yearname()` è diversa dalla funzione `year()` in quanto consente di compensare la data che si desidera valutare e di impostare il primo mese dell'anno.

Se il primo mese dell'anno non è gennaio, la funzione restituirà quattro cifre per i due anni nel periodo di dodici mesi che contengono la data. Ad esempio, se l'anno inizia ad aprile e la data da valutare è il giorno 30/06/2020, il risultato restituito è `2020-2021`.

Lo schema della funzione `yearname()` con aprile impostato come primo mese dell'anno.



Sintassi:

```
YearName(date[, period_no[, first_month_of_year]] )
```

Tipo di dati restituiti: duale

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero, in cui il valore 0 indica l'anno che contiene **date**. I valori negativi di **period_no** indicano gli anni precedenti, mentre i valori positivi indicano gli anni successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**. Il valore visualizzato sarà quindi una stringa che mostrerà due anni.

È possibile utilizzare i seguenti valori per impostare il primo mese dell'anno nell'argomento `first_month_of_year`:

valori first_month_of_year

Mese	Valore
Febbraio	2
March	3
Aprile	4
May	5
Giugno	6
Luglio	7
Agosto	8
Settembre	9
Ottobre	10
Novembre	11
Dicembre	12

Casi di utilizzo

La funzione `yearname()` è utile per confrontare le aggregazioni per anno. Ad esempio, può essere usata se si desidera visualizzare le vendite totali dei prodotti in base all'anno.

È possibile creare queste dimensioni nello script di caricamento utilizzando la funzione che consente di creare un campo in una tabella Calendario principale. Inoltre, possono essere create in un grafico come dimensioni calcolate.

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`yearname('10/19/2001')`	Restituisce '2001.'
`yearname('10/19/2001',-1)`	Restituisce '2000'.
`yearname('10/19/2001',0,4)`	Restituisce '2001-2002'.

Argomenti correlati

Argomento	Descrizione
year (page 1122)	Questa funzione restituisce un numero intero che rappresenta l'anno in cui l'espressione viene interpretat come data in base all'interpretazione numerica standard.

Esempio 1 - Nessun argomento aggiuntivo

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022 viene caricato in una tabella denominata '`Transactions`'.
- La variabile di sistema `DateFormat` che è impostata su '`MM/DD/YYYY`'.
- Un'istruzione LOAD precedente che utilizza `yearname()` e ne imposta il valore come campo `year_name`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
      *,
      yearname(date) as year_name
      ;
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
```

```
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `year_name`

Tabella dei risultati

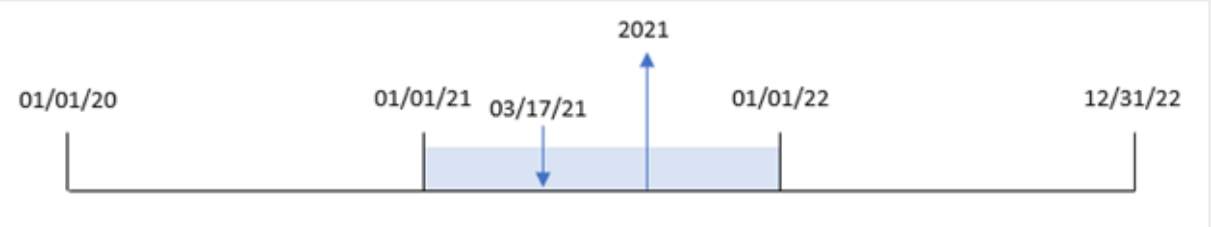
data	year_name
01/13/2020	2020
02/26/2020	2020
03/27/2020	2020
04/16/2020	2020
05/21/2020	2020
08/14/2020	2020
10/07/2020	2020
12/05/2020	2020
01/22/2021	2021
02/03/2021	2021
03/17/2021	2021
04/23/2021	2021
05/04/2021	2021
06/30/2021	2021
07/26/2021	2021

data	year_name
12/27/2021	2021
06/06/2022	2022
07/18/2022	2022
11/14/2022	2022
12/12/2022	2022

Il campo 'year_name' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `yearname()` e trasferendo il campo data come argomento della funzione.

La funzione `yearname()` identifica l'anno in cui rientra il valore della data e lo restituisce come valore anno a quattro cifre.

Diagramma della funzione `yearname()` che mostra il 2021 come valore per l'anno.



Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022 caricato in una tabella denominata 'Transazioni'.
- La variabile di sistema DateFormat impostata su 'MM/DD/YYYY'.
- Un'istruzione LOAD precedente che utilizza `yearname()` e ne imposta il valore come campo `year_name`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
      *,
      yearname(date,-1) as prior_year_name
      ;
```

```
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `prior_year_name`

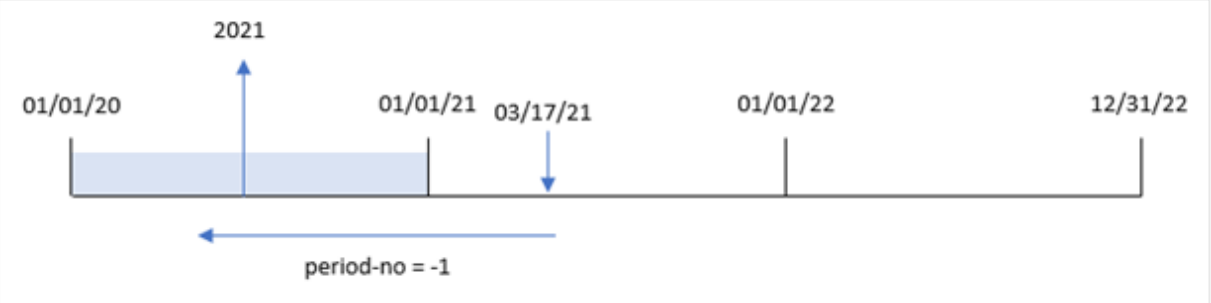
Tabella dei risultati

data	prior_year_name
01/13/2020	2019
02/26/2020	2019
03/27/2020	2019
04/16/2020	2019
05/21/2020	2019
08/14/2020	2019
10/07/2020	2019
12/05/2020	2019
01/22/2021	2020

data	prior_year_name
02/03/2021	2020
03/17/2021	2020
04/23/2021	2020
05/04/2021	2020
06/30/2021	2020
07/26/2021	2020
12/27/2021	2020
06/06/2022	2021
07/18/2022	2021
11/14/2022	2021
12/12/2022	2021

Poiché il valore `period_no` di `-1` viene utilizzato come argomento offset nella funzione `yearname()`, la funzione per prima cosa identifica l'anno in cui avvengono le transazioni. La funzione quindi si sposta di un anno prima e restituisce l'anno risultante.

Schema della funzione `yearname()` con il valore `period_no` con un impostato su `-1`.



Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- La variabile di sistema `DateFormat` che è impostata su '`MM/DD/YYYY`'.
- Un'istruzione LOAD precedente che utilizza `yearname()` e ne imposta il valore come campo `year_name`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
      *,
      yearname(date,0,4) as year_name
      ;
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `year_name`

Tabella dei risultati

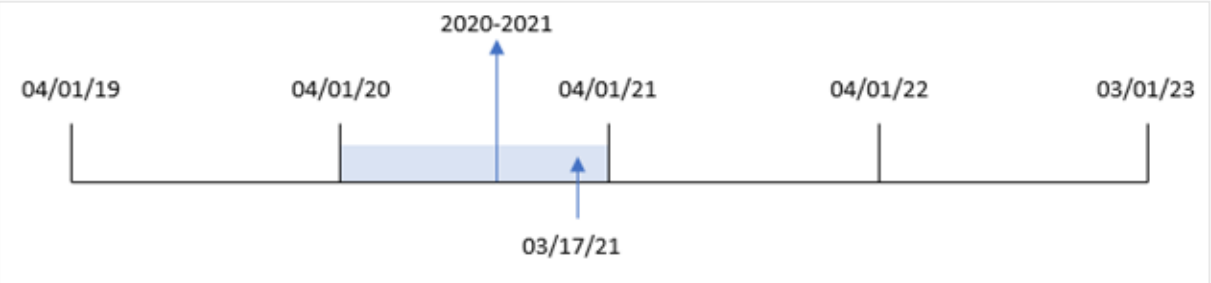
data	year_name
01/13/2020	2019-2020
02/26/2020	2019-2020
03/27/2020	2019-2020

data	year_name
04/16/2020	2020-2021
05/21/2020	2020-2021
08/14/2020	2020-2021
10/07/2020	2020-2021
12/05/2020	2020-2021
01/22/2021	2020-2021
02/03/2021	2020-2021
03/17/2021	2020-2021
04/23/2021	2021-2022
05/04/2021	2021-2022
06/30/2021	2021-2022
07/26/2021	2021-2022
12/27/2021	2021-2022
06/06/2022	2022-2023
07/18/2022	2022-2023
11/14/2022	2022-2023
12/12/2022	2022-2023

Poiché l'argomento `first_month_of_year` di `4` viene utilizzato nella funzione `yearname()`, l'inizio dell'anno va dal 1 gennaio al 1 aprile. Pertanto, ogni periodo di dodici mesi attraversa due anni civili e la funzione `yearname()` restituisce i due anni con formato a quattro cifre anni per le date valutate.

La transazione 8198 avviene il 17 marzo 2021. La funzione `yearname()` imposta l'inizio dell'anno il 1 aprile e la fine il 30 marzo. Pertanto, la transazione 8198 è avvenuta nel periodo dell'anno dal 1 aprile 2020 al 30 marzo 2021. Di conseguenza, la funzione `yearname()` restituisce il valore 2020-2021.

Lo schema della funzione `yearname()` con marzo impostato come primo mese dell'anno.



Esempio 4 - Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- La variabile di sistema `DateFormat` che è impostata su '`MM/DD/YYYY`'.

Tuttavia, il campo che restituisce l'anno in cui è avvenuta la transazione viene creato come misura in un oggetto grafico.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione:

```
date
```

Per calcolare il campo 'year_name', creare questa misura:

```
=yearname(date)
```

Tabella dei risultati

data	=yearname(date)
01/13/2020	2020
02/26/2020	2020
03/27/2020	2020
04/16/2020	2020
05/21/2020	2020
08/14/2020	2020
10/07/2020	2020
12/05/2020	2020
01/22/2021	2021
02/03/2021	2021
03/17/2021	2021
04/23/2021	2021
05/04/2021	2021
06/30/2021	2021
07/26/2021	2021
12/27/2021	2021
06/06/2022	2022
07/18/2022	2022
11/14/2022	2022
12/12/2022	2022

La misura 'year_name' viene creata nell'oggetto grafico utilizzando la funzione `yearname()` e trasferendo il campo data come argomento della funzione.

La funzione `yearname()` identifica l'anno in cui rientra il valore della data e lo restituisce come valore anno a quattro cifre.

Diagramma della funzione `yearname()` con 2021 come valore per l'anno.

Esempio 5 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Lo stesso set di dati dal primo esempio.
- La variabile di sistema `DateFormat` che è impostata su '`MM/DD/YYYY`'.

L'utente finale desidera un grafico che presenta le vendite totali per trimestre per le transazioni. Utilizzare la funzione `yearname()` come dimensione calcolata per creare questo grafico quando la dimensione `yearname()` non è disponibile nel modello dati.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,'01/13/2020',37.23
8189,'02/26/2020',17.17
8190,'03/27/2020',88.27
8191,'04/16/2020',57.42
8192,'05/21/2020',53.80
8193,'08/14/2020',82.06
8194,'10/07/2020',40.39
8195,'12/05/2020',87.21
8196,'01/22/2021',95.93
8197,'02/03/2021',45.89
8198,'03/17/2021',36.23
8199,'04/23/2021',25.66
8200,'05/04/2021',82.77
8201,'06/30/2021',69.98
8202,'07/26/2021',76.11
8203,'12/27/2021',25.12
```

```
8204,'06/06/2022',46.23
8205,'07/18/2022',84.21
8206,'11/14/2022',96.24
8207,'12/12/2022',67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella.

Per confrontare le aggregazioni per anno, creare la seguente dimensione calcolata:

```
=yearname(date)
```

Creare questa misura:

```
=sum(amount)
```

Impostare la misura **Number Formatting** su **Money**.

Tabella dei risultati

yearname(date)	=sum(amount)
2020	$463.55
2021	$457.69
2022	$294.35

yearstart

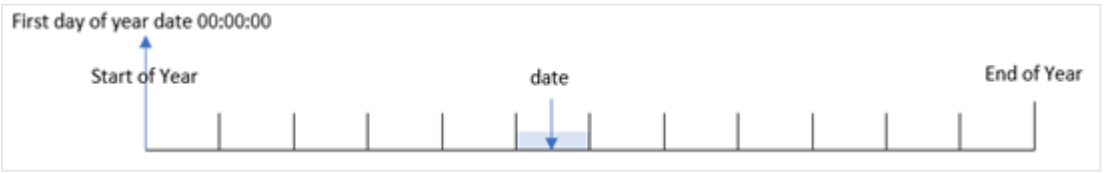
Questa funzione restituisce un indicatore temporale corrispondente all'inizio del primo giorno dell'anno contenente **date**. Il formato di output predefinito sarà il formato **DateFormat** impostato nello script.

Sintassi:

```
YearStart(date[, period_no[, first_month_of_year]])
```

In altre parole, la funzione `yearstart()` determina in quale anno cade la data. Quindi, restituisce data e ora, nel formato data, per il primo millisecondo di quell'anno. Il primo mese dell'anno è, per impostazione predefinita, gennaio. Tuttavia, è possibile modificare il mese da impostare come primo utilizzando l'argomento `first_month_of_year` nella funzione `yearstart()`.

Diagramma della funzione `yearstart()` che mostra l'intervallo di tempo che la funzione può coprire.



Casi di utilizzo

La funzione `yearstart()` viene utilizzata come parte di un'espressione quando si desidera che il calcolo utilizzi la frazione dell'anno trascorso fin'ora. Ad esempio, se si desidera calcolare l'interesse accumulato in un anno a oggi.

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
date	La data o la data e ora da valutare.
period_no	**period_no** è un numero intero, in cui il valore 0 indica l'anno che contiene **date**. I valori negativi di **period_no** indicano gli anni precedenti, mentre i valori positivi indicano gli anni successivi.
first_month_of_year	Se si intende utilizzare anni (fiscali) che non iniziano a gennaio, indicare un valore compreso tra 2 e 12 in **first_month_of_year**.

I mesi seguenti possono essere utilizzati in `first_month_of_year argument`:

valori first_month_of_year

Month	Value
February	2
March	3
April	4
May	5
June	6
July	7
August	8
September	9
October	10
November	11
December	12

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema,

a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempi di funzioni

Esempio	Risultato
`yearstart('10/19/2001')`	Returns 01/01/2001 00:00:00.
`yearstart('10/19/2001',-1)`	Returns 01/01/2000 00:00:00.
`yearstart('10/19/2001',0,4)`	Returns 04/01/2001 00:00:00.

Esempio 1 – Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022 viene caricato in una tabella denominata '`Transactions`'.
- Il campo della data è stato fornito nel formato della variabile di sistema `DateFormat`, ossia `MM/DD/YYYY`.
- Un'istruzione LOAD precedente che contiene quanto segue:
  - `yearstart()` funzione che è impostata come campo `year_start`.
  - La funzione `Timestamp()` impostata come campo `year_start_timestamp`.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       yearstart(date) as year_start,
       timestamp(yearstart(date)) as year_start_timestamp
       ;
Load
*
Inline
```

```
[
id,date,amount
8188,01/13/2020,37.23
8189,02/26/2020,17.17
8190,03/27/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,06/06/2022,46.23
8205,07/18/2022,84.21
8206,11/14/2022,96.24
8207,12/12/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`
- `year_start`
- `year_start_timestamp`

Tabella dei risultati

id	date	year_start	year_start_timestamp
8188	01/13/2020	01/01/2020	1/1/2020 12:00:00 AM
8189	02/26/2020	01/01/2020	1/1/2020 12:00:00 AM
8190	03/27/2020	01/01/2020	1/1/2020 12:00:00 AM
8191	04/16/2020	01/01/2020	1/1/2020 12:00:00 AM
8192	05/21/2020	01/01/2020	1/1/2020 12:00:00 AM
8193	08/14/2020	01/01/2020	1/1/2020 12:00:00 AM
8194	10/07/2020	01/01/2020	1/1/2020 12:00:00 AM
8195	12/05/2020	01/01/2020	1/1/2020 12:00:00 AM
8196	01/22/2021	01/01/2021	1/1/2020 12:00:00 AM

id	date	year_start	year_start_timestamp
8197	02/03/2021	01/01/2021	1/1/2020 12:00:00 AM
8198	03/17/2021	01/01/2021	1/1/2021 12:00:00 AM
8199	04/23/2021	01/01/2021	1/1/2021 12:00:00 AM
8200	05/04/2021	01/01/2021	1/1/2021 12:00:00 AM
8201	06/30/2021	01/01/2021	1/1/2021 12:00:00 AM
8202	07/26/2021	01/01/2021	1/1/2021 12:00:00 AM
8203	12/27/2021	01/01/2021	1/1/2021 12:00:00 AM
8204	06/06/2022	01/01/2022	1/1/2022 12:00:00 AM
8205	07/18/2022	01/01/2022	1/1/2022 12:00:00 AM
8206	11/14/2022	01/01/2022	1/1/2022 12:00:00 AM
8207	12/12/2022	01/01/2022	1/1/2022 12:00:00 AM

Il campo '`year_start`' viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `yearstart()` e trasferendo il campo data come argomento della funzione.

La funzione `yearstart()` inizialmente identifica in quale anno rientra il valore della data e restituisce data e ora per il primo millisecondo di quell'anno.

Schema della funzione `yearstart()` e della transazione 8199.

La transazione 8199 è avvenuta il 23 aprile 2021. La funzione `yearstart()` restituisce il primo millisecondo di quell'anno, ovvero il 1 gennaio alle ore 12:00:00 AM.

Esempio 2 – period_no

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, l'attività consiste nel creare un campo, '`previous_year_start`', che restituisce data e ora di inizio dell'anno precedente all'anno in cui ha avuto luogo la transazione.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       yearstart(date,-1) as previous_year_start,
       timestamp(yearstart(date,-1)) as previous_year_start_timestamp
       ;
Load
*
Inline
[
id,date,amount
8188,01/13/2020,37.23
8189,02/26/2020,17.17
8190,03/27/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,06/06/2022,46.23
8205,07/18/2022,84.21
8206,11/14/2022,96.24
8207,12/12/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- id
- date
- previous_year_start
- previous_ year_start_timestamp

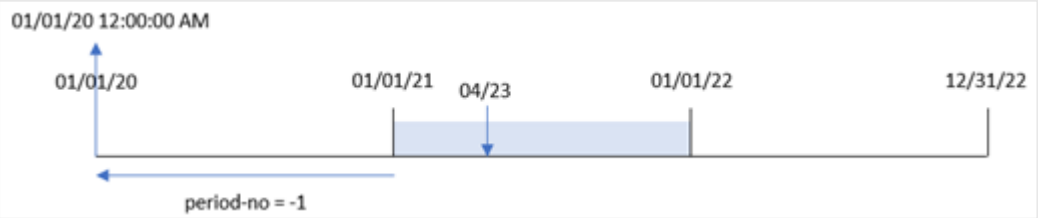
Tabella dei risultati

id	date	previous_year_start	previous_year_start_timestamp
8188	01/13/2020	01/01/2019	1/1/2019 12:00:00 AM

id	date	previous_year_start	previous_year_start_timestamp
8189	02/26/2020	01/01/2019	1/1/2019 12:00:00 AM
8190	03/27/2020	01/01/2019	1/1/2019 12:00:00 AM
8191	04/16/2020	01/01/2019	1/1/2019 12:00:00 AM
8192	05/21/2020	01/01/2019	1/1/2019 12:00:00 AM
8193	08/14/2020	01/01/2019	1/1/2019 12:00:00 AM
8194	10/07/2020	01/01/2019	1/1/2019 12:00:00 AM
8195	12/05/2020	01/01/2019	1/1/2019 12:00:00 AM
8196	01/22/2021	01/01/2020	1/1/2020 12:00:00 AM
8197	02/03/2021	01/01/2020	1/1/2020 12:00:00 AM
8198	03/17/2021	01/01/2020	1/1/2020 12:00:00 AM
8199	04/23/2021	01/01/2020	1/1/2020 12:00:00 AM
8200	05/04/2021	01/01/2020	1/1/2020 12:00:00 AM
8201	06/30/2021	01/01/2020	1/1/2020 12:00:00 AM
8202	07/26/2021	01/01/2020	1/1/2020 12:00:00 AM
8203	12/27/2021	01/01/2020	1/1/2020 12:00:00 AM
8204	06/06/2022	01/01/2021	1/1/2021 12:00:00 AM
8205	07/18/2022	01/01/2021	1/1/2021 12:00:00 AM
8206	11/14/2022	01/01/2021	1/1/2021 12:00:00 AM
8207	12/12/2022	01/01/2021	1/1/2021 12:00:00 AM

In questa istanza, poiché il valore `period_no` di `-1` viene utilizzato come argomento offset nella funzione `yearstart()`, la funzione per prima cosa identifica l'anno in cui avvengono le transazioni. Quindi, cerca l'anno precedente e identifica l'ultimo millisecondo di quell'anno.

Schema della funzione `yearstart()` con un valore `period_no` di -1.



La transazione 8199 è avvenuta il 23 aprile 2021. La funzione `yearstart()` restituisce il primo millisecondo dell'anno precendente, ovvero il 1 gennaio 2020 alle ore 12:00:00 AM, per il campo '`previous_year_start`'.

Esempio 3 – first_month_of_year

Script di caricamento e risultati

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, la politica aziendale prevede che l'anno inizi il 1 aprile.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
       *,
       yearstart(date,0,4) as year_start,
       timestamp(yearstart(date,0,4)) as year_start_timestamp
       ;
Load
*
Inline
[
id,date,amount
8188,01/13/2020,37.23
8189,02/26/2020,17.17
8190,03/27/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,06/06/2022,46.23
8205,07/18/2022,84.21
8206,11/14/2022,96.24
8207,12/12/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

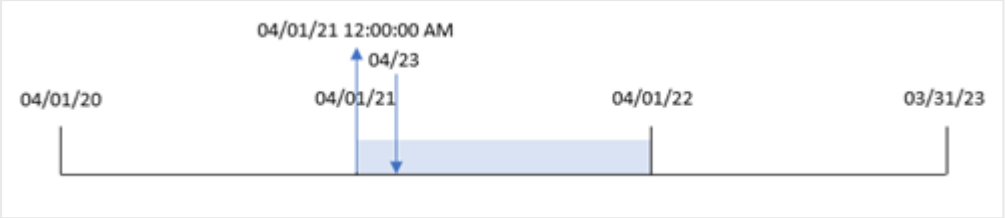
- id
- date
- year_start
- year_start_timestamp

Tabella dei risultati

id	date	year_start	year_start_timestamp
8188	01/13/2020	04/01/2019	4/1/2019 12:00:00 AM
8189	02/26/2020	04/01/2019	4/1/2019 12:00:00 AM
8190	03/27/2020	04/01/2019	4/1/2019 12:00:00 AM
8191	04/16/2020	04/01/2020	4/1/2020 12:00:00 AM
8192	05/21/2020	04/01/2020	4/1/2020 12:00:00 AM
8193	08/14/2020	04/01/2020	4/1/2020 12:00:00 AM
8194	10/07/2020	04/01/2020	4/1/2020 12:00:00 AM
8195	12/05/2020	04/01/2020	4/1/2020 12:00:00 AM
8196	01/22/2021	04/01/2020	4/1/2020 12:00:00 AM
8197	02/03/2021	04/01/2020	4/1/2020 12:00:00 AM
8198	03/17/2021	04/01/2020	4/1/2020 12:00:00 AM
8199	04/23/2021	04/01/2021	4/1/2021 12:00:00 AM
8200	05/04/2021	04/01/2021	4/1/2021 12:00:00 AM
8201	06/30/2021	04/01/2021	4/1/2021 12:00:00 AM
8202	07/26/2021	04/01/2021	4/1/2021 12:00:00 AM
8203	12/27/2021	04/01/2021	4/1/2021 12:00:00 AM
8204	06/06/2022	04/01/2022	4/1/2022 12:00:00 AM
8205	07/18/2022	04/01/2022	4/1/2022 12:00:00 AM
8206	11/14/2022	04/01/2022	4/1/2022 12:00:00 AM
8207	12/12/2022	04/01/2022	4/1/2022 12:00:00 AM

In questa istanza, poiché l'argomento `first_month_of_year` di `4` viene utilizzato nella funzione `yearstart ()`, imposta il primo giorno dell'anno come il 1 aprile e l'ultimo giorno dell'anno come il 31 marzo.

Lo schema della funzione `yearstart()` con il primo mese impostato su aprile.



La transazione 8199 è avvenuta il 23 aprile 2021. Poiché la funzione `yearstart()` imposta l'inizio dell'anno sul 1 aprile, lo restituisce come valore '`year_start`' per la transazione.

Esempio 4 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati è invariato e viene caricato nell'applicazione. Il calcolo che restituisce data e ora della data di inizio per l'anno in cui è avvenuta una transazione viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,01/13/2020,37.23
8189,02/26/2020,17.17
8190,03/27/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,06/06/2022,46.23
8205,07/18/2022,84.21
8206,11/14/2022,96.24
8207,12/12/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `id`
- `date`

Per calcolare in quale anno è avvenuta una transazione, creare le seguenti misure:

- `=yearstart(date)`
- `=timestamp(yearstart(date))`

Tabella dei risultati

id	date	=yearstart(date)	=timestamp(yearstart(date))
8188	06/06/2022	01/01/2022	1/1/2022 12:00:00 AM
8189	07/18/2022	01/01/2022	1/1/2022 12:00:00 AM
8190	11/14/2022	01/01/2022	1/1/2022 12:00:00 AM
8191	12/12/2022	01/01/2022	1/1/2022 12:00:00 AM
8192	01/22/2021	01/01/2021	1/1/2021 12:00:00 AM
8193	02/03/2021	01/01/2021	1/1/2021 12:00:00 AM
8194	03/17/2021	01/01/2021	1/1/2021 12:00:00 AM
8195	04/23/2021	01/01/2021	1/1/2021 12:00:00 AM
8196	05/04/2021	01/01/2021	1/1/2021 12:00:00 AM
8197	06/30/2021	01/01/2021	1/1/2021 12:00:00 AM
8198	07/26/2021	01/01/2021	1/1/2021 12:00:00 AM
8199	12/27/2021	01/01/2021	1/1/2021 12:00:00 AM
8200	01/13/2020	01/01/2020	1/1/2020 12:00:00 AM
8201	02/26/2020	01/01/2020	1/1/2020 12:00:00 AM
8202	03/27/2020	01/01/2020	1/1/2020 12:00:00 AM
8203	04/16/2020	01/01/2020	1/1/2020 12:00:00 AM
8204	05/21/2020	01/01/2020	1/1/2020 12:00:00 AM
8205	08/14/2020	01/01/2020	1/1/2020 12:00:00 AM
8206	10/07/2020	01/01/2020	1/1/2020 12:00:00 AM
8207	12/05/2020	01/01/2020	1/1/2020 12:00:00 AM

La misura '`start_of_year`' viene creata nell'oggetto grafico mediante l'utilizzo della funzione `yearstart()` e trasferendo il campo data come argomento della funzione.

La funzione `yearstart()` inizialmente identifica in quale anno rientra il valore della data e restituisce data e ora per il primo millisecondo di quell'anno.

Schema della funzione `yearstart()` e della transazione 8199.

La transazione 8199 è avvenuta il 23 aprile 2021. La funzione `yearstart()` restituisce il primo millisecondo di quell'anno, ovvero il 1 gennaio alle ore 12:00:00 AM.

Esempio 5 – Scenario

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati caricato in una tabella denominata '`Loans`'. La tabella contiene i seguenti campi:
  - Loan IDs.
  - Il saldo all'inizio dell'anno.
  - Il tasso di interesse semplice applicato su ciascun prestito ogni anno.

L'utente finale desidera un oggetto grafico che mostri, in base all'ID del prestito, gli interessi correnti che sono stati maturato per ciascun prestito nell'anno in corso.

Script di caricamento

```
Loans:
Load
*
Inline
[
loan_id,start_balance,rate
8188,$10000.00,0.024
8189,$15000.00,0.057
8190,$17500.00,0.024
8191,$21000.00,0.034
8192,$90000.00,0.084
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- loan_id
- start_balance

Per calcolare gli interessi maturati, creare la seguente misura:

```
=start_balance*(rate*(today(1)-yearstart(today(1)))/365)
```

Impostare la misura **Number Formatting** su **Money**.

Tabella dei risultati

loan_id	start_balance	=start_balance*(rate*(today(1)-yearstart(today(1)))/365)
8188	$10000.00	$39.73
8189	$15000.00	$339.66
8190	$17500.00	$166.85
8191	$21000.00	$283.64
8192	$90000.00	$3003.29

Utilizzando la data odierna come unico argomento, la funzione `yearstart()` restituisce la data di inizio dell'anno corrente. Sottraendo tale risultato dalla data corrente, l'espressione restituisce il numero di giorni trascorsi fino ad ora quest'anno.

Questo valore viene quindi moltiplicato per il tasso di interesse e diviso per 365, per restituire il tasso di interesse effettivo per questo periodo. Il tasso di interesse effettivo per il periodo viene quindi moltiplicato per il saldo iniziale del prestito, per restituire gli interessi maturati finoranel corso di quest'anno.

yeartodate

Questa funzione stabilisce se l'indicatore temporale di input ricade all'interno dell'anno in cui lo script è stato caricato per l'ultima volta e restituisce True in caso affermativo e False in caso negativo.

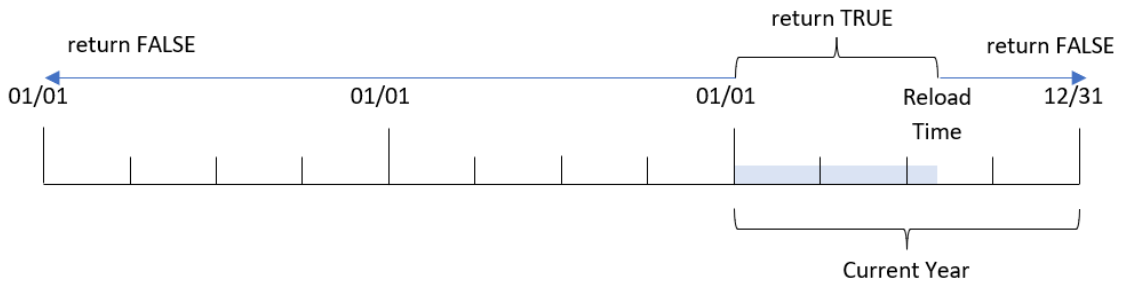
Sintassi:

```
YearToDate(timestamp[ , yearoffset [ , firstmonth [ , todaydate] ] ])
```

Tipo di dati restituiti: Booleano

> *In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.*

Schema esemplificativo della funzione `yeartodate()`



Se non si utilizza alcun parametro opzionale, l'anno rispetto alla data attuale sarà qualsiasi data che rientra in un anno di calendario a partire dal 1° gennaio fino all'ultima data di esecuzione dello script inclusa.

In altre parole, la funzione `yeartodate()`, quando viene attivata senza parametri aggiuntivi, viene utilizzata per valutare un timestamp e restituire un risultato booleano in base al fatto che la data si sia verificata nell'anno solare fino alla data di ricaricamento inclusa.

Tuttavia, è anche possibile sostituire la data di inizio dell'anno utilizzando l'argomento `firstmonth`, nonché fare confronti con gli anni precedenti o successivi utilizzando l'argomento `yearoffset`.

Infine, nei casi di insiemi di dati cronologici, la funzione `yeartodate()` fornisce un parametro per impostare `todaydate`, che confronterà invece il timestamp con l'anno solare fino alla data fornita nell'argomento `todaydate`.

Argomenti

Argomento	Descrizione
timestamp	Il timestamp da valutare, ad esempio '12/10/2012'.
yearoffset	Specificando un **yearoffset**, **yeartodate** restituisce True per lo stesso periodo di un altro anno. Un **yearoffset** negativo indica un anno precedente, mentre un differimento positivo indica un anno successivo. È possibile ottenere l'anno alla data odierna più recente specificando yearoffset = -1. Se omesso, viene utilizzato 0.
firstmonth	Se si specifica un valore **firstmonth** compreso tra 1 e 12 (1 se omesso), l'inizio dell'anno potrà essere spostato in avanti al primo giorno di qualsiasi mese. Se, ad esempio, si intende utilizzare un anno fiscale che inizi il 1° maggio, specificare **firstmonth** = 5. Il valore 1 indica un anno fiscale che inizia il 1° gennaio, mentre il valore 12 indica un anno fiscale che inizia il 1° dicembre.
todaydate	Se si specifica un valore **todaydate** (timestamp dell'ultima esecuzione dello script, se omesso), è possibile spostare il giorno utilizzato come limite superiore del periodo.

Casi di utilizzo

La funzione `yeartodate()` restituisce un risultato booleano. In genere, questo tipo di funzione viene utilizzato come condizione in un'espressione if. Questo restituisce un'aggregazione o un calcolo che dipende dal fatto che la data valutata si sia verificata nell'anno fino all'ultima data di ricaricamento dell'applicazione.

Ad esempio, la funzione YearToDate() può essere utilizzata per identificare tutte le apparecchiature prodotte fino a quel momento nell'anno corrente.

Gli esempi seguenti presuppongono che la data dell'ultimo ricaricamento sia il 18/11/2011.

Esempi di funzioni

Esempio	Risultato
`yeartodate( '11/18/2010')`	restituisce False
`yeartodate( '02/01/2011')`	restituisce True
`yeartodate( '11/18/2011')`	restituisce True
`yeartodate( '11/19/2011')`	restituisce False
`yeartodate( '11/19/2011', 0, 1, '12/31/2011')`	restituisce True
`yeartodate( '11/18/2010', -1)`	restituisce True
`yeartodate( '11/18/2011', -1)`	restituisce False
`yeartodate( '04/30/2011', 0, 5)`	restituisce False
`yeartodate( '05/01/2011', 0, 5)`	restituisce True

Impostazioni locali

Se non diversamente specificato, gli esempi di questo argomento utilizzano il seguente formato di data: MM/GG/AAAA. Il formato della data viene specificato nell'istruzione `SET DateFormat` nello script di caricamento dei dati. La formattazione predefinita della data potrebbe essere diversa nel proprio sistema, a causa delle impostazioni regionali e di altri fattori. È possibile modificare i formati degli esempi seguenti in base alle proprie esigenze. In alternativa, è possibile modificare i formati nel proprio script di caricamento per adattarli a questi esempi.

Le impostazioni regionali predefinite delle app si basano sulle impostazioni regionali del sistema del computer o del server in cui risulta installato Qlik Sense. Se il server Qlik Sense a cui si accede è impostato in Svezia, l'editor caricamento dati utilizzerà le impostazioni regionali svedesi per date, ora e valuta. Queste impostazioni di formato regionale non riguardano la lingua visualizzata nell'interfaccia utente Qlik Sense. Qlik Sense verrà visualizzato nella stessa lingua del browser utilizzato.

Esempio 1 – Esempio di base

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).
- La creazione di un campo, `year_to_date`, che determina quali transazioni sono state effettuate nell'anno solare fino alla data dell'ultimo ricaricamento.

Al momento in cui scriviamo, la data è il 26 aprile 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        yeartodate(date) as year_to_date
        ;
Load
*
Inline
[
id,date,amount
8188,01/10/2020,37.23
8189,02/28/2020,17.17
8190,04/09/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,02/02/2022,46.23
8205,02/26/2022,84.21
8206,03/07/2022,96.24
8207,03/11/2022,67.67
];
```

Risultati

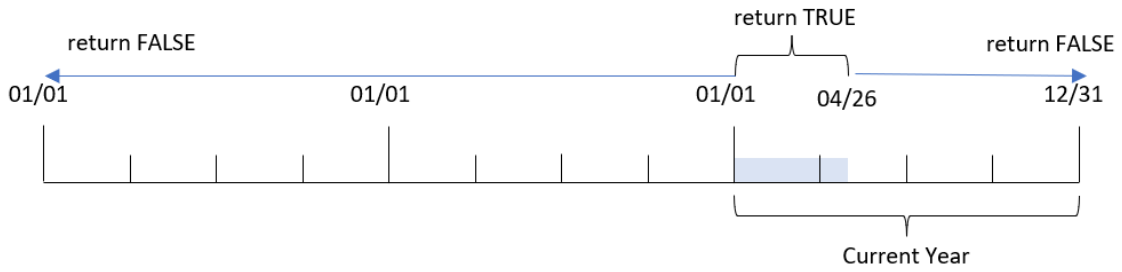
Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `year_to_date`

Tabella dei risultati

date	year_to_date
01/10/2020	0
02/28/2020	0
04/09/2020	0
04/16/2020	0
05/21/2020	0
08/14/2020	0
10/07/2020	0
12/05/2020	0
01/22/2021	0
02/03/2021	0
03/17/2021	0
04/23/2021	0
05/04/2021	0
06/30/2021	0
07/26/2021	0
12/27/2021	0
02/02/2022	-1
02/26/2022	-1
03/07/2022	-1
03/11/2022	-1

Schema della funzione `yeartodate()`, esempio base



Il campo `year_to_date` viene creato nell'istruzione LOAD precedente mediante l'uso della funzione `yeartodate()` e trasferendo il campo `date` come argomento della funzione.

Poiché non vengono passati altri parametri alla funzione, la funzione `yeartodate()`identifica inizialmente la data di ricaricamento e quindi i confini dell'anno solare in corso (a partire dal 1° gennaio) che restituirà un risultato booleano di TRUE.

Pertanto, qualsiasi transazione effettuata tra il 1° gennaio e il 26 aprile, data di ricaricamento, restituirà un risultato booleano di TRUE. Qualsiasi transazione che si verifichi prima dell'inizio del 2022 restituirà un risultato booleano di FALSE.

Esempio 2 – yearoffset

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `two_years_prior`, che determina quali transazioni hanno avuto luogo due anni prima dell'anno solare in corso.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        yeartodate(date,-2) as two_years_prior
        ;
Load
*
Inline
[
id,date,amount
8188,01/10/2020,37.23
8189,02/28/2020,17.17
8190,04/09/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
```

```
8204,02/02/2022,46.23
8205,02/26/2022,84.21
8206,03/07/2022,96.24
8207,03/11/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

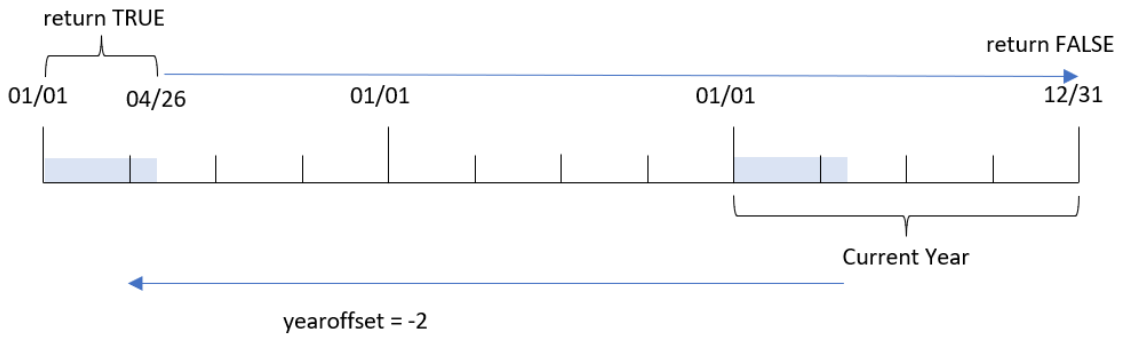
- `date`
- `two_years_prior`

Tabella dei risultati

date	two_years_prior
01/10/2020	-1
02/28/2020	-1
04/09/2020	-1
04/16/2020	-1
05/21/2020	0
08/14/2020	0
10/07/2020	0
12/05/2020	0
01/22/2021	0
02/03/2021	0
03/17/2021	0
04/23/2021	0
05/04/2021	0
06/30/2021	0
07/26/2021	0
12/27/2021	0
02/02/2022	0
02/26/2022	0
03/07/2022	0
03/11/2022	0

Utilizzando -2 come argomento `yearoffset` della funzione `yeartodate()`, questa sposta i confini del segmento dell'anno solare di confronto di ben due anni. Inizialmente, il segmento di anno corrisponde a un periodo compreso tra il 1° gennaio e il 26 aprile 2022. L'argomentazione `yearoffset` compensa poi questo segmento con quello di due anni prima. La data limite sarà quindi compresa tra il 1° gennaio e il 26 aprile 2020.

Schema della funzione `yeartodate()`, esempio di yearoffset



Pertanto, qualsiasi transazione effettuata tra il 1° gennaio e il 26 aprile 2020 restituirà un risultato booleano di TRUE. Tutte le transazioni che appaiono prima o dopo questo segmento restituiranno FALSE.

Esempio 3 – firstmonth

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `year_to_date`, che determina quali transazioni sono state effettuate nell'anno solare fino alla data dell'ultimo ricaricamento.

In questo esempio, abbiamo impostato l'inizio dell'anno fiscale al 1° luglio.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        yeartodate(date,0,7) as year_to_date
        ;
Load
*
Inline
```

```
[
id,date,amount
8188,01/10/2020,37.23
8189,02/28/2020,17.17
8190,04/09/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,02/02/2022,46.23
8205,02/26/2022,84.21
8206,03/07/2022,96.24
8207,03/11/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `year_to_date`

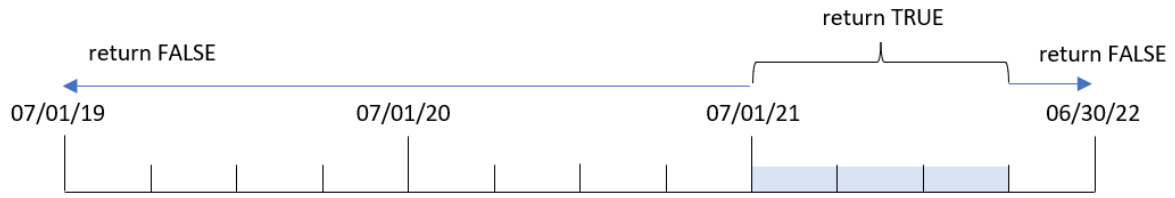
Tabella dei risultati

date	year_to_date
01/10/2020	0
02/28/2020	0
04/09/2020	0
04/16/2020	0
05/21/2020	0
08/14/2020	0
10/07/2020	0
12/05/2020	0
01/22/2021	0
02/03/2021	0
03/17/2021	0

date	year_to_date
04/23/2021	0
05/04/2021	0
06/30/2021	0
07/26/2021	-1
12/27/2021	-1
02/02/2022	-1
02/26/2022	-1
03/07/2022	-1
03/11/2022	-1

In questa istanza, poiché l'argomento `firstmonth` di 7 viene utilizzato nella funzione `yeartodate()`, imposta il primo giorno dell'anno al 1° luglio e l'ultimo giorno dell'anno al 30 giugno.

Schema della funzione `yeartodate()`, esempio firstmonth



Pertanto, qualsiasi transazione effettuata tra il 1° luglio 2021 e il 26 aprile 2022, data di ricaricamento, restituirà un risultato booleano di `TRUE`. Qualsiasi transazione effettuata prima del 1° luglio 2021 restituirà un risultato booleano di `FALSE`.

Esempio 4 – todaydate

Script di caricamento e risultati

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Vengono utilizzati lo stesso set di dati e lo stesso scenario del primo esempio.
- La creazione di un campo, `year_to_date`, che determina quali transazioni sono state effettuate nell'anno solare fino alla data dell'ultimo ricaricamento.

Tuttavia, in questo esempio, dobbiamo identificare tutte le transazioni avvenute nell'anno solare fino al 1° marzo 2022 incluso.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
    Load
        *,
        yeartodate(date, 0, 1, '03/01/2022') as year_to_date
;
Load
*
Inline
[
id,date,amount
8188,01/10/2020,37.23
8189,02/28/2020,17.17
8190,04/09/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,02/02/2022,46.23
8205,02/26/2022,84.21
8206,03/07/2022,96.24
8207,03/11/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere tali campi come dimensioni:

- `date`
- `year_to_date`

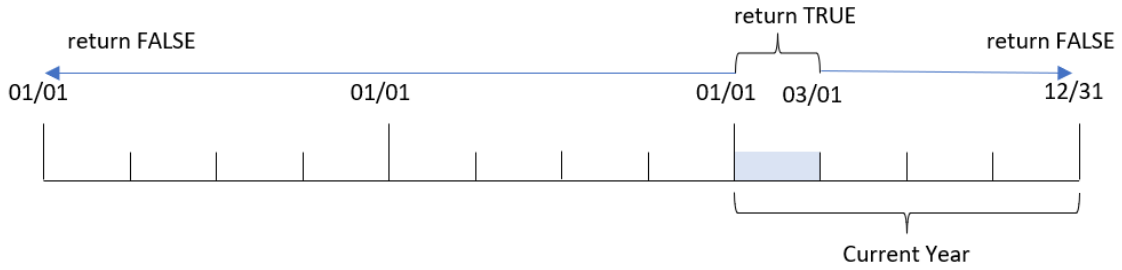
Tabella dei risultati

date	year_to_date
01/10/2020	0
02/28/2020	0

date	year_to_date
04/09/2020	0
04/16/2020	0
05/21/2020	0
08/14/2020	0
10/07/2020	0
12/05/2020	0
01/22/2021	0
02/03/2021	0
03/17/2021	0
04/23/2021	0
05/04/2021	0
06/30/2021	0
07/26/2021	0
12/27/2021	0
02/02/2022	-1
02/26/2022	-1
03/07/2022	0
03/11/2022	0

In questo caso, poiché la funzione `yeartodate()` utilizza l'argomento `todaydate` di 01/03/2022, imposta il limite finale del segmento dell'anno solare di confronto al 1° marzo 2022. È fondamentale fornire il parametro `firstmonth` (compreso tra 1 e 12), altrimenti la funzione restituirà risultati null.

Schema della funzione `yeartodate()`, esempio di utilizzo dell'argomento todaydate



Pertanto, qualsiasi transazione che si verifichi tra il 1° gennaio 2022 e il 1° marzo 2022, con il parametro `todaydate`, restituirà un risultato booleano di `TRUE`. Qualsiasi transazione avvenuta prima del 1° gennaio 2022 o dopo il 1° marzo 2022 restituirà un risultato booleano di `FALSE`.

Esempio 5 – Esempio di oggetto grafico

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene lo stesso set di dati e lo stesso scenario del primo esempio.

Tuttavia, in questo esempio, il set di dati invariato viene caricato nell'applicazione. Il calcolo che determina quali transazioni hanno avuto luogo nell'anno solare fino alla data dell'ultimo ricaricamento viene creato come misura in un oggetto grafico dell'applicazione.

Script di caricamento

```
Transactions:
Load
*
Inline
[
id,date,amount
8188,01/10/2020,37.23
8189,02/28/2020,17.17
8190,04/09/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,02/02/2022,46.23
8205,02/26/2022,84.21
8206,03/07/2022,96.24
8207,03/11/2022,67.67
];
```

Risultati

Caricare i dati e aprire un foglio. Creare una nuova tabella e aggiungere questo campo come dimensione: `date`.

Aggiungere la misura seguente:

```
=yeartodate(date)
```

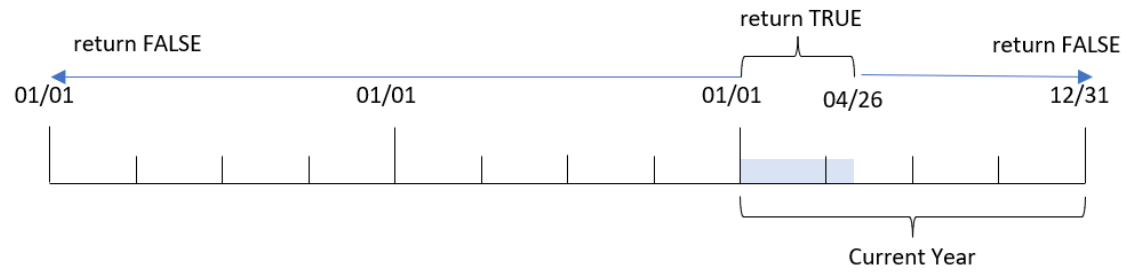
Tabella dei risultati

date	=yeartodate(date)
01/10/2020	0
02/28/2020	0
04/09/2020	0
04/16/2020	0
05/21/2020	0
08/14/2020	0
10/07/2020	0
12/05/2020	0
01/22/2021	0
02/03/2021	0
03/17/2021	0
04/23/2021	0
05/04/2021	0
06/30/2021	0
07/26/2021	0
12/27/2021	0
02/02/2022	-1
02/26/2022	-1
03/07/2022	-1
03/11/2022	-1

La misura `year_to_date` viene creata nell'oggetto grafico mediante l'utilizzo della funzione `yeartodate()` e trasferendo il campo `date` come argomento della funzione.

Poiché non vengono passati altri parametri alla funzione, la funzione `yeartodate()`identifica inizialmente la data di ricaricamento e quindi i confini dell'anno solare in corso (a partire dal 1° gennaio) che restituirà un risultato booleano di `TRUE`.

Schema della funzione `yeartodate()`, esempio di utilizzo dell'oggetto grafico



Qualsiasi transazione effettuata tra il 1° gennaio e il 26 aprile, data di ricaricamento, restituirà un risultato booleano di `TRUE`. Qualsiasi transazione che si verifichi prima dell'inizio del 2022 restituirà un risultato booleano di `FALSE`.

Esempio 6 – Scenario

Script di caricamento ed espressione del grafico

Panoramica

Aprire l'editor caricamento dati e aggiungere lo script di caricamento sotto in una nuova scheda.

Lo script di caricamento contiene:

- Un set di dati contenente un insieme di transazioni tra il 2020 e il 2022, caricato in una tabella denominata `Transactions`.
- Il campo della data fornito nel formato della variabile di sistema `DateFormat` (MM/GG/AAAA).

L'utente finale desidera un oggetto KPI che presenti le vendite totali per il periodo equivalente nel 2021 come l'anno corrente fino ad oggi, all'ultimo orario di ricaricamento.

Al momento in cui scriviamo, la data è il 16 giugno 2022.

Script di caricamento

```
SET DateFormat='MM/DD/YYYY';

Transactions:
Load
*
Inline
[
id,date,amount
8188,01/10/2020,37.23
8189,02/28/2020,17.17
8190,04/09/2020,88.27
8191,04/16/2020,57.42
8192,05/21/2020,53.80
8193,08/14/2020,82.06
8194,10/07/2020,40.39
```

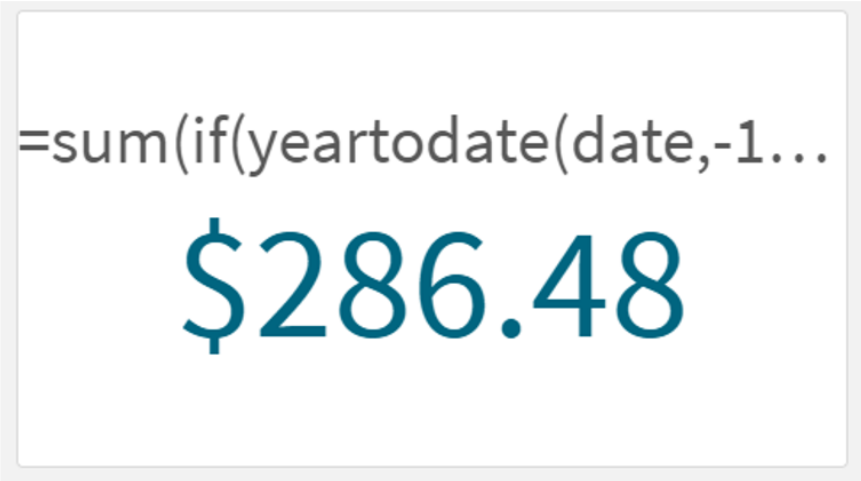
```
8195,12/05/2020,87.21
8196,01/22/2021,95.93
8197,02/03/2021,45.89
8198,03/17/2021,36.23
8199,04/23/2021,25.66
8200,05/04/2021,82.77
8201,06/30/2021,69.98
8202,07/26/2021,76.11
8203,12/27/2021,25.12
8204,02/02/2022,46.23
8205,02/26/2022,84.21
8206,03/07/2022,96.24
8207,03/11/2022,67.67
];
```

Risultati

Procedere come indicato di seguito:

1. Creare un oggetto KPI.
2. Creare la seguente misura di aggregazione per calcolare le vendite totali:
   `=sum(if(yeartodate(date,-1),amount,0))`
3. Impostare la **Formattazione numero** della misura su **Denaro**.

Grafico `yeartodate()` KPI per il 2021



La funzione `yeartodate()` restituisce un valore booleano quando si valutano le date di ciascun ID transazione. Poiché il ricaricamento è avvenuto il 16 giugno 2022, la funzione `yeartodate` segmenta il periodo dell'anno tra il 01/01/2022 e il 16/06/2022. Tuttavia, poiché nella funzione `period_no` è stato utilizzato il valore -1, questi confini vengono spostati all'anno precedente. Pertanto, per qualsiasi transazione che si verifichi tra il 01/01/2021 e il 06/16/2021, la funzione `yeartodate()` restituisce un valore booleano di `TRUE` e somma l'importo.

5.8 Funzioni esponenziali e logaritmiche

In questa sezione vengono descritte le funzioni correlate ai calcoli esponenziali e logaritmici. Tutte le funzioni possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche.

Nelle seguenti funzioni i parametri sono espressioni dove **x** e **y** devono essere interpretati come numeri reali valutati.

exp

La funzione esponenziale naturale, e^x, che utilizza l'algoritmo naturale **e** come base. Il risultato è un numero positivo.

```
exp(x )
```

Esempi e risultati:

`exp(3)` restituisce 20,085.

log

Il logaritmo naturale di **x**. La funzione viene definita solo se **x**> 0. Il risultato è un numero.

```
log(x )
```

Esempi e risultati:

`log(3)` restituisce 1,0986

log10

Il logaritmo comune (base 10) di **x**. La funzione viene definita solo se **x**> 0. Il risultato è un numero.

```
log10(x )
```

Esempi e risultati:

`log10(3)` restituisce 0,4771

pow

Restituisce **x** alla potenza di **y**. Il risultato è un numero.

```
pow(x,y )
```

Esempi e risultati:

`pow(3, 3)` restituisce 27

sqr

x quadrato (**x** alla potenza di 2). Il risultato è un numero.

```
sqr (x )
```

Esempi e risultati:

`sqr(3)` restituisce 9

sqrt
Radice quadrata di **x**. La funzione viene definita solo se **x** >= 0. Il risultato è un numero positivo.

```
sqrt(x )
```

Esempi e risultati:

`sqrt(3)` restituisce 1,732

5.9 Funzioni di campo

Queste funzioni possono essere utilizzate solo nelle espressioni grafiche.

Le funzioni di campo restituiscono numeri interi o stringhe che identificano aspetti differenti delle selezioni dei campi.

Funzioni di conteggio

GetAlternativeCount
GetAlternativeCount() viene utilizzato per trovare il numero di valori alternativi (grigio chiaro) nel campo identificato.

```
GetAlternativeCount - funzione per grafici (field_name)
```

GetExcludedCount
GetExcludedCount() trova il numero di valori distinti esclusi nel campo identificato. I valori esclusi includono i campi alternativi (grigio chiaro), esclusi (grigio scuro) e selezionati esclusi (grigio scuro con segno di spunta).

```
GetExcludedCount - funzione per grafici (page 1186)(field_name)
```

GetNotSelectedCount
Questa funzione grafica restituisce il numero di valori non selezionati nel campo denominato **fieldname**. Affinché questa funzione risulti pertinente, il campo deve essere in modalità And.

```
GetNotSelectedCount - funzione per grafici(fieldname [,
includeexcluded=false])
```

GetPossibleCount
GetPossibleCount() viene utilizzato per trovare il numero di valori possibili nel campo identificato. Se il campo identificato include selezioni, i campi selezionati (verdi) vengono conteggiati. In caso contrario, vengono conteggiati i valori associati (bianco).

```
GetPossibleCount - funzione per grafici(field_name)
```

GetSelectedCount

GetSelectedCount() trova il numero di valori selezionati (verdi) in un campo.

```
GetSelectedCount - funzione per grafici (field_name [, include_excluded])
```

Funzioni di campo e di selezione

GetCurrentSelections

GetCurrentSelections() restituisce un elenco delle selezioni attuali all'interno dell'app. Se invece le selezioni vengono effettuate usando una stringa di ricerca all'interno di una casella di ricerca, **GetCurrentSelections()** restituisce la stringa di ricerca.

```
GetCurrentSelections - funzione per grafici([record_sep [,tag_sep [,value_sep
[,max_values]]]])
```

GetFieldSelections

GetFieldSelections() restituisce una **stringa** con le selezioni attuali in un campo.

```
GetFieldSelections - funzione per grafici ( field_name [, value_sep [, max_
values]])
```

GetObjectDimension

GetObjectDimension() restituisce il nome della dimensione. **Index** è un numero intero facoltativo che indica la dimensione da restituire.

```
GetObjectDimension - funzione per grafici ([index])
```

GetObjectField

GetObjectField() restituisce il nome della dimensione. **Index** è un numero intero opzionale che indica la dimensione da restituire.

```
GetObjectField - funzione per grafici ([index])
```

GetObjectMeasure

GetObjectMeasure() restituisce il nome della misura. **Index** è un numero intero opzionale che indica la misura da restituire.

```
GetObjectMeasure - funzione per grafici ([index])
```

GetAlternativeCount - funzione per grafici

GetAlternativeCount() viene utilizzato per trovare il numero di valori alternativi (grigio chiaro) nel campo identificato.

Sintassi:

```
GetAlternativeCount (field_name)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Il campo contenente la scala di dati da misurare.

Esempi e risultati:

L'esempio seguente utilizza il campo **First name** caricato in una casella di filtro.

Esempi e risultati

Esempi	Risultati
Presupponendo che **John** sia selezionato in **First name**. `GetAlternativeCount ([First name])`	4 in quanto in **First name** vi sono 4 valori univoci ed esclusi (grigio).
Presupponendo che **John** e **Peter** siano selezionati. `GetAlternativeCount ([First name])`	3 in quanto in **First name** vi sono 3 valori univoci ed esclusi (grigio).
Presupponendo che in **First name** non sia selezionato alcun valore. `GetAlternativeCount ([First name])`	0 in quanto non vi sono selezioni.

Dati utilizzati nell'esempio:

```
Names:
LOAD * inline [
First name|Last name|Initials|Has cellphone
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC|No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');
```

GetCurrentSelections - funzione per grafici

GetCurrentSelections() restituisce un elenco delle selezioni attuali all'interno dell'app. Se invece le selezioni vengono effettuate usando una stringa di ricerca all'interno di una casella di ricerca, **GetCurrentSelections()** restituisce la stringa di ricerca.

Se vengono utilizzate opzioni, è necessario specificare record_sep. Per specificare una nuova riga, impostare **record_sep** su **chr(13)&chr(10)**.

Se vengono selezionati tutti i valori meno due o tutti i valori meno uno, verrà utilizzato rispettivamente il formato NOT x,y' o 'NOT y'. Se si selezionano tutti i valori e il conteggio dei valori è superiore a max_ values, verrà restituito il testo ALL.

Sintassi:

```
GetCurrentSelections ([record_sep [, tag_sep [, value_sep [, max_values [, state_name]]]]])
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomenti	Descrizione
record_sep	Il separatore da inserire tra i record del campo. Il valore predefinito è <CR><LF> che significa una nuova linea.
tag_sep	Il separatore da inserire tra il tag del nome di campo e i valori del campo. Il valore predefinito è ': '.
value_sep	Il separatore da inserire tra i valori di campo. Il valore predefinito è ', '.
max_values	Il numero massimo di valori di campo da elencare singolarmente. Se si seleziona un numero maggiore di valori, verrà utilizzato il formato 'x di y valori'. Il valore predefinito è 6.
state_name	Il nome di uno stato alternato che è stato scelto per la specifica visualizzazione. Se viene utilizzato l'argomento **state_name**, saranno prese in considerazione solo le selezioni associate al nome dello stato specificato.

Esempi e risultati:

Nel seguente esempio sono utilizzati due campi caricati in caselle di filtro differenti, una per il nome **First name** e un'altra per **Initials**.

Esempi e risultati

Esempi	Risultati
Presupponendo che **John** sia selezionato in **First name**. `GetCurrentSelections ()`	'First name: John'
Presupponendo che **John** e **Peter** siano selezionati in **First name**. `GetCurrentSelections ()`	'First name: John, Peter'
Presupponendo che **John** e **Peter** siano selezionati in **First name** e **JA** sia selezionato in **Initials**. `GetCurrentSelections ()`	'First name: John, Peter Initials: JA'

Esempi	Risultati
Presupponendo che **John** sia selezionato in **First name** e che **JA** sia selezionato in **Initials**. `GetCurrentSelections ( chr(13)&chr(10) , ' = ' )`	'First name = John Initials = JA'
Presupponendo che si siano selezionati tutti i nomi tranne Sue in **First name** e non sia stata effettuata alcuna selezione in **Initials**. `GetCurrentSelections (chr(13)&chr(10), '=', ',' ,3)`	'First name=NOT Sue'

Dati utilizzati nell'esempio:

```
Names:
LOAD * inline [
First name|Last name|Initials|Has cellphone
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC|No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');
```

GetExcludedCount - funzione per grafici

GetExcludedCount() trova il numero di valori distinti esclusi nel campo identificato. I valori esclusi includono i campi alternativi (grigio chiaro), esclusi (grigio scuro) e selezionati esclusi (grigio scuro con segno di spunta).

Sintassi:

```
GetExcludedCount (field_name)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomenti	Descrizione
field_name	Il campo contenente la scala di dati da misurare.

Esempi e risultati:

Nell'esempio seguente sono utilizzati tre campi caricati in caselle di filtro diverse, una per **First name**, una per **Last name** e una per **Initials**.

Esempi e risultati

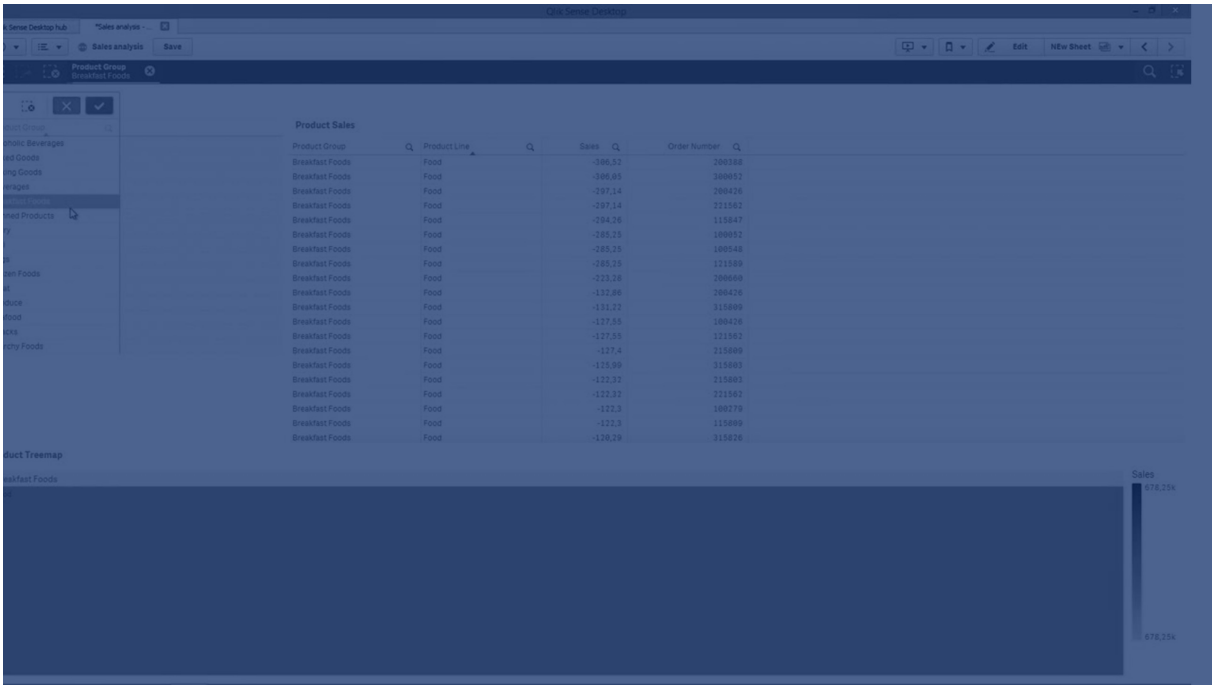
Esempi	Risultati
Se in **First name** non sono selezionati valori.	`GetExcludedCount (Initials) = 0` Non vi sono selezioni.
Se **John** è selezionato in **First name**.	`GetExcludedCount (Initials) = 5` Vi sono 5 valori esclusi in **Initials** con colore grigio scuro. La sesta cella (JA) sarà bianca in quanto è associata alla selezione John in **First name**.
Se **John** e **Peter** sono selezionati.	`GetExcludedCount (Initials) = 3` John è associato a 1 valore e Peter è associato a 2 valori, in **Initials**.
Se sono selezionati **John** e **Peter** in **First name** e quindi è selezionato **Franc** in **Last name**.	`GetExcludedCount ([First name]) = 4` Vi sono 4 valori esclusi in **First name** con colore grigio scuro. **GetExcludedCount()** viene valutato per i campi con valori esclusi, compresi i campi alternativi e selezionati esclusi.
Se sono selezionati **John** e **Peter** in **First name** e quindi sono selezionati **Franc** e **Anderson** in **Last name**.	`GetExcludedCount (Initials) = 4` Vi sono 4 valori esclusi in **Initials** con colore grigio scuro. Le altre due celle (JA e PF) saranno bianche in quanto associate alle selezioni John e Peter in **First name**.
Se sono selezionati **John** e **Peter** in **First name** e quindi sono selezionati **Franc** e **Anderson** in **Last name**.	`GetExcludedCount ([Last name]) = 4` Vi sono 4 valori esclusi in **Initials**. Devonshire è di colore grigio chiaro mentre Brown, Carr e Elliot sono di colore grigio scuro.

Dati utilizzati nell'esempio:

```
Names:
LOAD * inline [
First name|Last name|Initials|Has cellphone
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC|No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');
```

GetFieldSelections - funzione per grafici

GetFieldSelections() restituisce una **stringa** con le selezioni attuali in un campo.



Se si selezionano tutti i valori meno due o tutti i valori meno uno, verrà utilizzato rispettivamente il formato 'NOT x,y' o il formato 'NOT y'. Se si selezionano tutti i valori e il conteggio dei valori è superiore a max_ values, verrà restituito il testo ALL.

Sintassi:

```
GetFieldSelections ( field_name [, value_sep [, max_values [, state_name]]])
```

Tipo di dati restituiti: stringa

Restituisci formati stringa

Formato	Descrizione
'a, b, c'	Se il numero dei valori selezionati è max_values o meno, la stringa restituita è un elenco dei valori selezionati. I valori sono separati con value_sep come delimitatore.
'NOT a, b, c'	Se il numero dei valori non selezionati è max_values o meno, la stringa restituita è un elenco dei valori non selezionati con NOT come prefisso. I valori sono separati con value_sep come delimitatore.
'x of y'	x = il numero di valori selezionati y = il numero totale di valori Questo viene restituito quando max_values < x < (y - max_values).
'ALL'	Restituito se tutti i valori risultano selezionati.
'-'	Restituito se nessun valore risulta selezionato.
<search string>	Se si è effettuato una selezione usando la ricerca, la stringa di ricerca viene restituita.

Argomenti:

Argomenti

Argomenti	Descrizione
field_name	Il campo contenente la scala di dati da misurare.
value_sep	Il separatore da inserire tra i valori di campo. Il valore predefinito è ', '.
max_values	Il numero massimo di valori di campo da elencare singolarmente. Se si seleziona un numero maggiore di valori, verrà utilizzato il formato 'x di y valori'. Il valore predefinito è 6.
state_name	Il nome di uno stato alternato che è stato scelto per la specifica visualizzazione. Se viene utilizzato l'argomento **state_name**, saranno prese in considerazione solo le selezioni associate al nome dello stato specificato.

Esempi e risultati:

L'esempio seguente utilizza il campo **First name** caricato in una casella di filtro.

Esempi e risultati

Esempi	Risultati
Presupponendo che **John** sia selezionato in **First name**. `GetFieldSelections ([First name])`	'John'
Presupponendo che **John** e **Peter** siano selezionati. `GetFieldSelections ([First name])`	'John,Peter'
Presupponendo che **John** e **Peter** siano selezionati. `GetFieldSelections ([First name],'; ')`	'John; Peter'
Presupponendo che **John**, **Sue**, **Mark** siano selezionati in **First name**. `GetFieldSelections ([First name],';',2)`	'NOT Jane;Peter', perché il valore 2 è dichiarato come il valore dell'argomento max_values. In caso contrario, il risultato sarebbe John; Sue; Mark.

Dati utilizzati nell'esempio:

```
Names:
LOAD * inline [
First name|Last name|Initials|Has cellphone
```

```
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC|No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');
```

GetNotSelectedCount - funzione per grafici

Questa funzione grafica restituisce il numero di valori non selezionati nel campo denominato **fieldname**. Affinché questa funzione risulti pertinente, il campo deve essere in modalità And.

Sintassi:

```
GetNotSelectedCount(fieldname [, includeexcluded=false])
```

Argomenti:

Argomenti

Argomento	Descrizione
fieldname	Il nome del campo da valutare.
includeexcluded	Se **includeexcluded** viene dichiarato come True, il conteggio includerà i valori selezionati esclusi dalle selezioni in un altro campo.

Esempi:

```
GetNotSelectedCount( Country )
GetNotSelectedCount( Country, true )
```

GetObjectDimension - funzione per grafici

GetObjectDimension() restituisce il nome della dimensione. **Index** è un numero intero facoltativo che indica la dimensione da restituire.

Non è possibile utilizzare questa funzione in un grafico nelle posizioni seguenti: titolo, sottotitolo, piè di pagina, espressione linea di riferimento.

Non è possibile fare riferimento al nome di una dimensione o misura in un altro oggetto usando l'Object ID.

Sintassi:

```
GetObjectDimension ([index])
```

Esempio:

```
GetObjectDimension(1)
```

Esempio: espressione del grafico

Tabella Qlik Sense che mostra esempi della funzione GetObjectDimension in un'espressione del grafico

transaction_date	customer_id	transaction_quantity	=GetObjectDimension ()	=GetObjectDimension (0)	=GetObjectDimension (1)
2018/08/30	049681	13	transaction_date	transaction_date	customer_id
2018/08/30	203521	6	transaction_date	transaction_date	customer_id
2018/08/30	203521	21	transaction_date	transaction_date	customer_id

Se si desidera restituire il nome di una misura, usare invece la funzione **GetObjectMeasure**.

GetObjectField - funzione per grafici

GetObjectField() restituisce il nome della dimensione. **Index** è un numero intero opzionale che indica la dimensione da restituire.

Non è possibile utilizzare questa funzione in un grafico nelle posizioni seguenti: titolo, sottotitolo, piè di pagina, espressione linea di riferimento.

Non è possibile fare riferimento al nome di una dimensione o misura in un altro oggetto usando l'Object ID.

Sintassi:

```
GetObjectField ([index])
```

Esempio:

```
GetObjectField(1)
```

Esempio: espressione del grafico

Tabella Qlik Sense che mostra esempi della funzione GetObjectField in un'espressione del grafico.

transaction_date	customer_id	transaction_quantity	=GetObjectField ()	=GetObjectField (0)	=GetObjectField (1)
2018/08/30	049681	13	transaction_date	transaction_date	customer_id
2018/08/30	203521	6	transaction_date	transaction_date	customer_id
2018/08/30	203521	21	transaction_date	transaction_date	customer_id

Se si desidera restituire il nome di una misura, usare invece la funzione **GetObjectMeasure**.

GetObjectMeasure - funzione per grafici

GetObjectMeasure() restituisce il nome della misura. **Index** è un numero intero opzionale che indica la misura da restituire.

Non è possibile utilizzare questa funzione in un grafico nelle posizioni seguenti: titolo, sottotitolo, piè di pagina, espressione linea di riferimento.

Non è possibile fare riferimento al nome di una dimensione o misura in un altro oggetto usando l'Object ID.

Sintassi:

```
GetObjectMeasure ([index])
```

Esempio:

```
GetObjectMeasure(1)
```

Esempio: espressione del grafico

Tabella Qlik Sense che mostra esempi della funzione GetObjectMeasure in un'espressione del grafico

customer_id	sum (transaction_quantity)	Avg (transaction_quantity)	=GetObjectMeasure ()	=GetObjectMeasure(0)	=GetObjectMeasure(1)
49681	13	13	sum(transaction_quantity)	sum(transaction_quantity)	Avg(transaction_quantity)
203521	27	13.5	sum(transaction_quantity)	sum(transaction_quantity)	Avg(transaction_quantity)

Se si desidera restituire il nome di una dimensione, usare invece la funzione **GetObjectField**.

GetPossibleCount - funzione per grafici

GetPossibleCount() viene utilizzato per trovare il numero di valori possibili nel campo identificato. Se il campo identificato include selezioni, i campi selezionati (verdi) vengono conteggiati. In caso contrario, vengono conteggiati i valori associati (bianco). .

Per i campi con selezioni, **GetPossibleCount()** restituisce il numero di campi selezionati (verdi).

Tipo di dati restituiti: numero intero

Sintassi:

```
GetPossibleCount (field_name)
```

Argomenti:

Argomenti

Argomenti	Descrizione
field_name	Il campo contenente la scala di dati da misurare.

Esempi e risultati:

Nel seguente esempio sono utilizzati due campi caricati in caselle di filtro differenti, una per il nome **First name** e un'altra per **Initials**.

Esempi e risultati

Esempi	Risultati
Presupponendo che **John** sia selezionato in **First name**. `GetPossibleCount ([Initials])`	1 in quanto in Initials è presente un valore è associato con la selezione, **John**, in **First name**.
Presupponendo che **John** sia selezionato in **First name**. `GetPossibleCount ([First name])`	1 in quanto vi è una 1 selezione , **John**, in **First name**.
Presupponendo che **Peter** sia selezionato in **First name**. `GetPossibleCount ([Initials])`	2 in quanto Peter è associato con 2 valori in **Initials**.
Presupponendo che in **First name** non sia selezionato alcun valore. `GetPossibleCount ([First name])`	5 in quanto non vi sono selezioni e vi sono 5 valori univoci in **First name**.
Presupponendo che in **First name** non sia selezionato alcun valore. `GetPossibleCount ([Initials])`	6 in quanto non vi sono selezioni e vi sono 6 valori univoci in **Initials**.

Dati utilizzati nell'esempio:

```
Names:
LOAD * inline [
First name|Last name|Initials|Has cellphone
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC|No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');
```

GetSelectedCount - funzione per grafici

GetSelectedCount() trova il numero di valori selezionati (verdi) in un campo.

Sintassi:

```
GetSelectedCount (field_name [, include_excluded [, state_name]])
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomenti	Descrizione
field_name	Il campo contenente la scala di dati da misurare.
include_ excluded	Se impostato su **True()**, il conteggio includerà i valori selezionati, i quali sono attualmente esclusi dalle selezioni in altri campi. Se invece è False o omesso, questi valori non verranno inclusi.
state_name	Il nome di uno stato alternato che è stato scelto per la specifica visualizzazione. Se viene utilizzato l'argomento **state_name**, saranno prese in considerazione solo le selezioni associate al nome dello stato specificato.

Esempi e risultati:

Nel seguente esempio, sono utilizzati tre campi caricati in caselle di filtro differenti, una per il nome **First name**, una per **Initials** e una per **Has cellphone**.

Esempi e risultati

Esempi	Risultati
Presupponendo che **John** sia selezionato in **First name**. `GetSelectedCount ([First name])`	1 in quanto in **First name** è selezionato un valore.
Presupponendo che **John** sia selezionato in **First name**. `GetSelectedCount ([Initials])`	0 in quanto in **Initials** non è selezionato alcun valore.
Senza alcuna selezione in **First name**, selezionare tutti i valori in **Initials** quindi, selezionare il valore **Yes** in **Has cellphone**. `GetSelectedCount ([Initials], True ())`	6. Sebbene tutte le selezioni con MC e PD di **Initials** abbiano **Has cellphone** impostato su **No**, il risultato è sempre 6, perché l'argomento include_excluded è impostato su True().

Dati utilizzati nell'esempio:

```
Names:
LOAD * inline [
First name|Last name|Initials|Has cellphone
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC|No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');
```

5.10 Funzioni di file

Le funzioni di file (disponibili solo nelle espressioni di script) restituiscono informazioni sul file tabellare in corso di lettura. Queste funzioni restituiranno un valore NULL per tutte le sorgenti dati tranne che per i file tabella (eccezione: **ConnectString( )**).

Prospetto delle funzioni di file

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Attribute

Questa funzione di script restituisce il valore dei metatag di diversi file multimediali come testo. Sono supportati i formati di file seguenti: MP3, WMA, WMV, PNG e JPG. Se il file **filename** non esiste, il formato di file non è supportato oppure non è presente il metatag **attributename**, verrà restituito NULL.

```
Attribute (filename, attributename)
```

ConnectString

La funzione **ConnectString()** restituisce il nome della connessione dati attiva per le connessioni ODBC o OLE DB. La funzione restituisce una stringa vuota se non è stata eseguita alcuna istruzione **connect** o dopo un'istruzione **disconnect**.

```
ConnectString ()
```

FileBaseName

La funzione **FileBaseName** restituisce una stringa contenente il nome del file tabella in corso di lettura, senza percorso o estensione.

```
FileBaseName ()
```

FileDir

La funzione **FileDir** restituisce una stringa contenente il percorso della directory del file tabella in corso di lettura.

```
FileDir ()
```

FileExtension

La funzione **FileExtension** restituisce una stringa contenente l'estensione del file tabella in corso di lettura.

```
FileExtension ()
```

FileName

La funzione **FileName** restituisce una stringa contenente il nome del file tabella in corso di lettura, senza percorso ma includendo l'estensione.

```
FileName ()
```

FilePath

La funzione **FilePath** restituisce una stringa contenente il percorso completo del file tabella in corso di lettura.

```
FilePath ()
```

FileSize

La funzione **FileSize** restituisce un valore intero contenente le dimensioni in byte del file filename oppure, se non viene specificato alcun filename, del file tabella in corso di lettura.

```
FileSize ()
```

FileTime

La funzione **FileTime** restituisce un indicatore temporale in UTC per la data e l'ora dell'ultima modifica del file filename. Se non viene specificato alcun filename, la funzione farà riferimento al file tabella in corso di lettura.

```
FileTime ([ filename ])
```

GetFolderPath

La funzione **GetFolderPath** restituisce il valore della funzione Microsoft Windows *SHGetFolderPath*. Questa funzione utilizza come input il nome di una cartella Microsoft Windows e restituisce il percorso completo della cartella.

```
GetFolderPath ()
```

QvdCreateTime

Questa funzione di script restituisce l'intestazione XML relativa alla data e ora da un file QVD, se disponibile, altrimenti restituisce NULL. Nella data e ora, l'ora è fornita in UTC.

```
QvdCreateTime (filename)
```

QvdFieldName

La funzione script restituisce il nome del numero campo **fieldno** in un file QVD. Se il campo non esiste, viene restituito NULL.

```
QvdFieldName (filename , fieldno)
```

QvdNoOfFields

Questa funzione dello script restituisce il numero di campi all'interno di file QVD.

```
QvdNoOfFields (filename)
```

QvdNoOfRecords

Questa funzione dello script restituisce il numero di record attualmente presente in un file QVD.

```
QvdNoOfRecords (filename)
```

QvdTableName

Questa funzione di script restituisce il nome della tabella memorizzata in un file QVD.

```
QvdTableName (filename)
```

Attribute

Questa funzione di script restituisce il valore dei metatag di diversi file multimediali come testo. Sono supportati i formati di file seguenti: MP3, WMA, WMV, PNG e JPG. Se il file **filename** non esiste, il formato di file non è supportato oppure non è presente il metatag **attributename**, verrà restituito NULL.

Sintassi:

```
Attribute(filename, attributename)
```

È possibile leggere un elevato numero di metatag. Negli esempi di questo argomento viene mostrato quali tag è possibile leggere per i relativi tipi di file supportati.

> *È possibile leggere solo i metatag salvati nel file in base alla specifica pertinente, ad esempio ID2v3 per file MP3 o EXIF per file JPG, non metainformazioni salvate in **Esplora file Windows**.*

Argomenti:

Argomenti

Argomento	Descrizione
filename	Il nome di un file multimediale comprensivo del percorso, se necessario, come connessione dati di una cartella. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:\data* • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** *data*
attributename	Il nome di un metatag.

Negli esempi viene utilizzata la funzione **GetFolderPath** per trovare i percorsi dei file multimediali. Poiché la funzione **GetFolderPath** è supportata solo nella modalità legacy, è necessario sostituire i riferimenti a **GetFolderPath** con un percorso di connessione dei dati lib:// quando si utilizza questa funzione in modalità standard o in Qlik Sense SaaS.

Restrizione dell'accesso al file system (page 1482)

Example 1: File MP3

Lo script legge tutti i metatag MP3 possibili nella cartella *MyMusic*.

```
// Script to read MP3 meta tags for each vExt in 'mp3' for each vFoundFile in filelist(
GetFolderPath('MyMusic') & '\*.'& vExt ) FileList: LOAD FileLongName,     subfield
(FileLongName,'\',-1) as FileShortName,     num(FileSize(FileLongName),'# ### ### ###',',','
') as FileSize,     FileTime(FileLongName) as FileTime,      // ID3v1.0 and ID3v1.1 tags
Attribute(FileLongName, 'Title') as Title,     Attribute(FileLongName, 'Artist') as Artist,
 Attribute(FileLongName, 'Album') as Album,     Attribute(FileLongName, 'Year') as Year,
Attribute(FileLongName, 'Comment') as Comment,     Attribute(FileLongName, 'Track') as Track,
   Attribute(FileLongName, 'Genre') as Genre,

     // ID3v2.3 tags     Attribute(FileLongName, 'AENC') as AENC, // Audio encryption
Attribute(FileLongName, 'APIC') as APIC, // Attached picture     Attribute(FileLongName,
'COMM') as COMM, // Comments     Attribute(FileLongName, 'COMR') as COMR, // Commercial frame
   Attribute(FileLongName, 'ENCR') as ENCR, // Encryption method registration     Attribute
(FileLongName, 'EQUA') as EQUA, // Equalization     Attribute(FileLongName, 'ETCO') as ETCO,
// Event timing codes     Attribute(FileLongName, 'GEOB') as GEOB, // General encapsulated
object     Attribute(FileLongName, 'GRID') as GRID, // Group identification registration
Attribute(FileLongName, 'IPLS') as IPLS, // Involved people list     Attribute(FileLongName,
'LINK') as LINK, // Linked information     Attribute(FileLongName, 'MCDI') as MCDI, // Music
CD identifier     Attribute(FileLongName, 'MLLT') as MLLT, // MPEG location lookup table
Attribute(FileLongName, 'OWNE') as OWNE, // Ownership frame     Attribute(FileLongName,
'PRIV') as PRIV, // Private frame     Attribute(FileLongName, 'PCNT') as PCNT, // Play counter
    Attribute(FileLongName, 'POPM') as POPM, // Popularimeter

     Attribute(FileLongName, 'POSS') as POSS, // Position synchronisation frame     Attribute
(FileLongName, 'RBUF') as RBUF, // Recommended buffer size     Attribute(FileLongName, 'RVAD')
as RVAD, // Relative volume adjustment     Attribute(FileLongName, 'RVRB') as RVRB, // Reverb
   Attribute(FileLongName, 'SYLT') as SYLT, // Synchronized lyric/text     Attribute
(FileLongName, 'SYTC') as SYTC, // Synchronized tempo codes     Attribute(FileLongName,
'TALB') as TALB, // Album/Movie/Show title     Attribute(FileLongName, 'TBPM') as TBPM, // BPM
(beats per minute)     Attribute(FileLongName, 'TCOM') as TCOM, // Composer     Attribute
(FileLongName, 'TCON') as TCON, // Content type     Attribute(FileLongName, 'TCOP') as TCOP,
// Copyright message     Attribute(FileLongName, 'TDAT') as TDAT, // Date     Attribute
(FileLongName, 'TDLY') as TDLY, // Playlist delay

     Attribute(FileLongName, 'TENC') as TENC, // Encoded by     Attribute(FileLongName,
'TEXT') as TEXT, // Lyricist/Text writer     Attribute(FileLongName, 'TFLT') as TFLT, // File
type     Attribute(FileLongName, 'TIME') as TIME, // Time     Attribute(FileLongName, 'TIT1')
as TIT1, // Content group description     Attribute(FileLongName, 'TIT2') as TIT2, //
Title/songname/content description     Attribute(FileLongName, 'TIT3') as TIT3, //
Subtitle/Description refinement     Attribute(FileLongName, 'TKEY') as TKEY, // Initial key
 Attribute(FileLongName, 'TLAN') as TLAN, // Language(s)     Attribute(FileLongName, 'TLEN')
as TLEN, // Length     Attribute(FileLongName, 'TMED') as TMED, // Media type
```

```
    Attribute(FileLongName, 'TOAL') as TOAL, // Original album/movie/show title    Attribute
(FileLongName, 'TOFN') as TOFN, // Original filename    Attribute(FileLongName, 'TOLY') as
TOLY, // Original lyricist(s)/text writer(s)    Attribute(FileLongName, 'TOPE') as TOPE, //
Original artist(s)/performer(s)    Attribute(FileLongName, 'TORY') as TORY, // Original
release year    Attribute(FileLongName, 'TOWN') as TOWN, // File owner/licensee    Attribute
(FileLongName, 'TPE1') as TPE1, // Lead performer(s)/Soloist(s)    Attribute(FileLongName,
'TPE2') as TPE2, // Band/orchestra/accompaniment

    Attribute(FileLongName, 'TPE3') as TPE3, // Conductor/performer refinement    Attribute
(FileLongName, 'TPE4') as TPE4, // Interpreted, remixed, or otherwise modified by
Attribute(FileLongName, 'TPOS') as TPOS, // Part of a set    Attribute(FileLongName, 'TPUB')
as TPUB, // Publisher    Attribute(FileLongName, 'TRCK') as TRCK, // Track number/Position in
set    Attribute(FileLongName, 'TRDA') as TRDA, // Recording dates    Attribute
(FileLongName, 'TRSN') as TRSN, // Internet radio station name    Attribute(FileLongName,
'TRSO') as TRSO, // Internet radio station owner

    Attribute(FileLongName, 'TSIZ') as TSIZ, // Size    Attribute(FileLongName, 'TSRC') as
TSRC, // ISRC (international standard recording code)    Attribute(FileLongName, 'TSSE') as
TSSE, // Software/Hardware and settings used for encoding    Attribute(FileLongName, 'TYER')
as TYER, // Year    Attribute(FileLongName, 'TXXX') as TXXX, // User defined text information
frame    Attribute(FileLongName, 'UFID') as UFID, // Unique file identifier    Attribute
(FileLongName, 'USER') as USER, // Terms of use    Attribute(FileLongName, 'USLT') as USLT,
// Unsychronized lyric/text transcription    Attribute(FileLongName, 'WCOM') as WCOM, //
Commercial information    Attribute(FileLongName, 'WCOP') as WCOP, // Copyright/Legal
information

    Attribute(FileLongName, 'WOAF') as WOAF, // Official audio file webpage    Attribute
(FileLongName, 'WOAR') as WOAR, // Official artist/performer webpage    Attribute
(FileLongName, 'WOAS') as WOAS, // Official audio source webpage    Attribute(FileLongName,
'WORS') as WORS, // Official internet radio station homepage    Attribute(FileLongName,
'WPAY') as WPAY, // Payment    Attribute(FileLongName, 'WPUB') as WPUB, // Publishers
official webpage    Attribute(FileLongName, 'WXXX') as WXXX; // User defined URL link frame
LOAD @1:n as FileLongName Inline "$(vFoundFile)" (fix, no labels); Next vFoundFile Next vExt
```

Example 2: JPEG

Lo script legge tutti i metatag EXIF possibili dai file JPG nella cartella *MyPictures*.

```
// Script to read Jpeg Exif meta tags for each vExt in 'jpg', 'jpeg', 'jpe', 'jfif', 'jif',
'jfi' for each vFoundFile in filelist( GetFolderPath('MyPictures') & '\*.'& vExt )

 FileList: LOAD FileLongName,    subfield(FileLongName,'\',-1) as FileShortName,    num
(FileSize(FileLongName),'# ### ### ###',',',' ') as FileSize,    FileTime(FileLongName) as
FileTime,    // ************  Exif Main (IFD0) Attributes  ************    Attribute
(FileLongName, 'ImageWidth') as ImageWidth,    Attribute(FileLongName, 'ImageLength') as
ImageLength,    Attribute(FileLongName, 'BitsPerSample') as BitsPerSample,    Attribute
(FileLongName, 'Compression') as Compression,

    //  examples: 1=uncompressed, 2=CCITT, 3=CCITT 3, 4=CCITT 4,

    //5=LZW, 6=JPEG (old style), 7=JPEG, 8=Deflate, 32773=PackBits RLE,    Attribute
(FileLongName, 'PhotometricInterpretation') as PhotometricInterpretation,

    //  examples: 0=WhiteIsZero, 1=BlackIsZero, 2=RGB, 3=Palette, 5=CMYK, 6=YCbCr,
Attribute(FileLongName, 'ImageDescription') as ImageDescription,    Attribute(FileLongName,
```

```
'Make') as Make,     Attribute(FileLongName, 'Model') as Model,     Attribute(FileLongName,
'StripOffsets') as StripOffsets,     Attribute(FileLongName, 'Orientation') as Orientation,

    //  examples: 1=TopLeft, 2=TopRight, 3=BottomRight, 4=BottomLeft,

    // 5=LeftTop, 6=RightTop, 7=RightBottom, 8=LeftBottom,      Attribute(FileLongName,
'SamplesPerPixel') as SamplesPerPixel,     Attribute(FileLongName, 'RowsPerStrip') as
RowsPerStrip,     Attribute(FileLongName, 'StripByteCounts') as StripByteCounts,     Attribute
(FileLongName, 'XResolution') as XResolution,     Attribute(FileLongName, 'YResolution') as
YResolution,     Attribute(FileLongName, 'PlanarConfiguration') as PlanarConfiguration,

    //  examples: 1=chunky format, 2=planar format,      Attribute(FileLongName,
'ResolutionUnit') as ResolutionUnit,

    //  examples: 1=none, 2=inches, 3=centimeters,      Attribute(FileLongName,
'TransferFunction') as TransferFunction,     Attribute(FileLongName, 'Software') as Software,
   Attribute(FileLongName, 'DateTime') as DateTime,     Attribute(FileLongName, 'Artist') as
Artist,     Attribute(FileLongName, 'HostComputer') as HostComputer,     Attribute
(FileLongName, 'WhitePoint') as WhitePoint,     Attribute(FileLongName,
'PrimaryChromaticities') as PrimaryChromaticities,     Attribute(FileLongName,
'YCbCrCoefficients') as YCbCrCoefficients,     Attribute(FileLongName, 'YCbCrSubSampling') as
YCbCrSubSampling,     Attribute(FileLongName, 'YCbCrPositioning') as YCbCrPositioning,

    //  examples: 1=centered, 2=co-sited,      Attribute(FileLongName, 'ReferenceBlackWhite')
as ReferenceBlackWhite,     Attribute(FileLongName, 'Rating') as Rating,     Attribute
(FileLongName, 'RatingPercent') as RatingPercent,     Attribute(FileLongName,
'ThumbnailFormat') as ThumbnailFormat,

    //  examples: 0=Raw Rgb, 1=Jpeg,      Attribute(FileLongName, 'Copyright') as Copyright,
  Attribute(FileLongName, 'ExposureTime') as ExposureTime,     Attribute(FileLongName,
'FNumber') as FNumber,     Attribute(FileLongName, 'ExposureProgram') as ExposureProgram,

    //  examples: 0=Not defined, 1=Manual, 2=Normal program, 3=Aperture priority, 4=Shutter
priority,

    // 5=Creative program, 6=Action program, 7=Portrait mode, 8=Landscape mode, 9=Bulb,
Attribute(FileLongName, 'ISOSpeedRatings') as ISOSpeedRatings,     Attribute(FileLongName,
'TimeZoneOffset') as TimeZoneOffset,     Attribute(FileLongName, 'SensitivityType') as
SensitivityType,

    //  examples: 0=Unknown, 1=Standard output sensitivity (SOS), 2=Recommended exposure index
(REI),

    // 3=ISO speed, 4=Standard output sensitivity (SOS) and Recommended exposure index (REI),

    //5=Standard output sensitivity (SOS) and ISO Speed, 6=Recommended exposure index (REI)
and ISO Speed,

    // 7=Standard output sensitivity (SOS) and Recommended exposure index (REI) and ISO speed,
     Attribute(FileLongName, 'ExifVersion') as ExifVersion,     Attribute(FileLongName,
'DateTimeOriginal') as DateTimeOriginal,     Attribute(FileLongName, 'DateTimeDigitized') as
DateTimeDigitized,     Attribute(FileLongName, 'ComponentsConfiguration') as
ComponentsConfiguration,
```

```
    //  examples: 1=Y, 2=Cb, 3=Cr, 4=R, 5=G, 6=B,      Attribute(FileLongName,
'CompressedBitsPerPixel') as CompressedBitsPerPixel,     Attribute(FileLongName,
'ShutterSpeedValue') as ShutterSpeedValue,     Attribute(FileLongName, 'ApertureValue') as
ApertureValue,     Attribute(FileLongName, 'BrightnessValue') as BrightnessValue, //
examples: -1=Unknown,      Attribute(FileLongName, 'ExposureBiasValue') as ExposureBiasValue,
   Attribute(FileLongName, 'MaxApertureValue') as MaxApertureValue,     Attribute
(FileLongName, 'SubjectDistance') as SubjectDistance,

    //  examples: 0=Unknown, -1=Infinity,      Attribute(FileLongName, 'MeteringMode') as
MeteringMode,

    //  examples: 0=Unknown, 1=Average, 2=CenterWeightedAverage, 3=Spot,

    // 4=MultiSpot, 5=Pattern, 6=Partial, 255=Other,      Attribute(FileLongName,
'LightSource') as LightSource,

    //  examples: 0=Unknown, 1=Daylight, 2=Fluorescent, 3=Tungsten, 4=Flash, 9=Fine weather,

    // 10=Cloudy weather, 11=Shade, 12=Daylight fluorescent,

    // 13=Day white fluorescent, 14=Cool white fluorescent,

    // 15=White fluorescent, 17=Standard light A, 18=Standard light B, 19=Standard light C,

    // 20=D55, 21=D65, 22=D75, 23=D50, 24=ISO studio tungsten, 255=other light source,
Attribute(FileLongName, 'Flash') as Flash,     Attribute(FileLongName, 'FocalLength') as
FocalLength,     Attribute(FileLongName, 'SubjectArea') as SubjectArea,     Attribute
(FileLongName, 'MakerNote') as MakerNote,     Attribute(FileLongName, 'UserComment') as
UserComment,     Attribute(FileLongName, 'SubSecTime') as SubSecTime,

     Attribute(FileLongName, 'SubsecTimeOriginal') as SubsecTimeOriginal,     Attribute
(FileLongName, 'SubsecTimeDigitized') as SubsecTimeDigitized,     Attribute(FileLongName,
'XPTitle') as XPTitle,     Attribute(FileLongName, 'XPComment') as XPComment,

     Attribute(FileLongName, 'XPAuthor') as XPAuthor,     Attribute(FileLongName,
'XPKeywords') as XPKeywords,     Attribute(FileLongName, 'XPSubject') as XPSubject,
Attribute(FileLongName, 'FlashpixVersion') as FlashpixVersion,     Attribute(FileLongName,
'ColorSpace') as ColorSpace, //  examples: 1=sRGB, 65535=Uncalibrated,      Attribute
(FileLongName, 'PixelXDimension') as PixelXDimension,     Attribute(FileLongName,
'PixelYDimension') as PixelYDimension,     Attribute(FileLongName, 'RelatedSoundFile') as
RelatedSoundFile,

     Attribute(FileLongName, 'FocalPlaneXResolution') as FocalPlaneXResolution,     Attribute
(FileLongName, 'FocalPlaneYResolution') as FocalPlaneYResolution,     Attribute(FileLongName,
'FocalPlaneResolutionUnit') as FocalPlaneResolutionUnit,

    //  examples: 1=None, 2=Inch, 3=Centimeter,      Attribute(FileLongName, 'ExposureIndex')
as ExposureIndex,     Attribute(FileLongName, 'SensingMethod') as SensingMethod,

    //  examples: 1=Not defined, 2=One-chip color area sensor, 3=Two-chip color area sensor,

    // 4=Three-chip color area sensor, 5=Color sequential area sensor,

    // 7=Trilinear sensor, 8=Color sequential linear sensor,      Attribute(FileLongName,
'FileSource') as FileSource,
```

```
    //  examples: 0=Other, 1=Scanner of transparent type,

    // 2=Scanner of reflex type, 3=Digital still camera,      Attribute(FileLongName,
'SceneType') as SceneType,

    //  examples: 1=A directly photographed image,      Attribute(FileLongName, 'CFAPattern')
as CFAPattern,     Attribute(FileLongName, 'CustomRendered') as CustomRendered,

    //  examples: 0=Normal process, 1=Custom process,      Attribute(FileLongName,
'ExposureMode') as ExposureMode,

    //  examples: 0=Auto exposure, 1=Manual exposure, 2=Auto bracket,      Attribute
(FileLongName, 'WhiteBalance') as WhiteBalance,

    //  examples: 0=Auto white balance, 1=Manual white balance,      Attribute(FileLongName,
'DigitalZoomRatio') as DigitalZoomRatio,     Attribute(FileLongName, 'FocalLengthIn35mmFilm')
as FocalLengthIn35mmFilm,     Attribute(FileLongName, 'SceneCaptureType') as SceneCaptureType,

    //  examples: 0=Standard, 1=Landscape, 2=Portrait, 3=Night scene,      Attribute
(FileLongName, 'GainControl') as GainControl,

    //  examples: 0=None, 1=Low gain up, 2=High gain up, 3=Low gain down, 4=High gain down,
  Attribute(FileLongName, 'Contrast') as Contrast,

    //  examples: 0=Normal, 1=Soft, 2=Hard,      Attribute(FileLongName, 'Saturation') as
Saturation,

    //  examples: 0=Normal, 1=Low saturation, 2=High saturation,      Attribute(FileLongName,
'Sharpness') as Sharpness,

    //  examples: 0=Normal, 1=Soft, 2=Hard,      Attribute(FileLongName,
'SubjectDistanceRange') as SubjectDistanceRange,

    //  examples: 0=Unknown, 1=Macro, 2=Close view, 3=Distant view,      Attribute
(FileLongName, 'ImageUniqueID') as ImageUniqueID,     Attribute(FileLongName,
'BodySerialNumber') as BodySerialNumber,     Attribute(FileLongName, 'CMNT_GAMMA') as CMNT_
GAMMA,     Attribute(FileLongName, 'PrintImageMatching') as PrintImageMatching,     Attribute
(FileLongName, 'OffsetSchema') as OffsetSchema,

     // ************   Interoperability Attributes   ************     Attribute(FileLongName,
'InteroperabilityIndex') as InteroperabilityIndex,     Attribute(FileLongName,
'InteroperabilityVersion') as InteroperabilityVersion,     Attribute(FileLongName,
'InteroperabilityRelatedImageFileFormat') as InteroperabilityRelatedImageFileFormat,
Attribute(FileLongName, 'InteroperabilityRelatedImageWidth') as
InteroperabilityRelatedImageWidth,     Attribute(FileLongName,
'InteroperabilityRelatedImageLength') as InteroperabilityRelatedImageLength,     Attribute
(FileLongName, 'InteroperabilityColorSpace') as InteroperabilityColorSpace,

    //  examples: 1=sRGB, 65535=Uncalibrated,      Attribute(FileLongName,
'InteroperabilityPrintImageMatching') as InteroperabilityPrintImageMatching,     //
************   GPS Attributes   ************     Attribute(FileLongName, 'GPSVersionID') as
GPSVersionID,     Attribute(FileLongName, 'GPSLatitudeRef') as GPSLatitudeRef,     Attribute
(FileLongName, 'GPSLatitude') as GPSLatitude,     Attribute(FileLongName, 'GPSLongitudeRef')
as GPSLongitudeRef,     Attribute(FileLongName, 'GPSLongitude') as GPSLongitude,     Attribute
(FileLongName, 'GPSAltitudeRef') as GPSAltitudeRef,
```

```
    //  examples: 0=Above sea level, 1=Below sea level,      Attribute(FileLongName,
'GPSAltitude') as GPSAltitude,     Attribute(FileLongName, 'GPSTimeStamp') as GPSTimeStamp,
 Attribute(FileLongName, 'GPSSatellites') as GPSSatellites,     Attribute(FileLongName,
'GPSStatus') as GPSStatus,     Attribute(FileLongName, 'GPSMeasureMode') as GPSMeasureMode,
 Attribute(FileLongName, 'GPSDOP') as GPSDOP,     Attribute(FileLongName, 'GPSSpeedRef') as
GPSSpeedRef,

     Attribute(FileLongName, 'GPSSpeed') as GPSSpeed,     Attribute(FileLongName,
'GPSTrackRef') as GPSTrackRef,     Attribute(FileLongName, 'GPSTrack') as GPSTrack,
Attribute(FileLongName, 'GPSImgDirectionRef') as GPSImgDirectionRef,     Attribute
(FileLongName, 'GPSImgDirection') as GPSImgDirection,     Attribute(FileLongName,
'GPSMapDatum') as GPSMapDatum,     Attribute(FileLongName, 'GPSDestLatitudeRef') as
GPSDestLatitudeRef,

     Attribute(FileLongName, 'GPSDestLatitude') as GPSDestLatitude,     Attribute
(FileLongName, 'GPSDestLongitudeRef') as GPSDestLongitudeRef,     Attribute(FileLongName,
'GPSDestLongitude') as GPSDestLongitude,     Attribute(FileLongName, 'GPSDestBearingRef') as
GPSDestBearingRef,     Attribute(FileLongName, 'GPSDestBearing') as GPSDestBearing,
Attribute(FileLongName, 'GPSDestDistanceRef') as GPSDestDistanceRef,

     Attribute(FileLongName, 'GPSDestDistance') as GPSDestDistance,     Attribute
(FileLongName, 'GPSProcessingMethod') as GPSProcessingMethod,     Attribute(FileLongName,
'GPSAreaInformation') as GPSAreaInformation,     Attribute(FileLongName, 'GPSDateStamp') as
GPSDateStamp,     Attribute(FileLongName, 'GPSDifferential') as GPSDifferential;

    //  examples: 0=No correction, 1=Differential correction,  LOAD @1:n as FileLongName
Inline "$(vFoundFile)" (fix, no labels); Next vFoundFile Next vExt
```

Example 3: File multimediali di Windows

Lo script legge tutti i metatag WMA/WMV ASF possibili nella cartella *MyMusic*.

```
/ Script to read WMA/WMV ASF meta tags for each vExt in 'asf', 'wma', 'wmv' for each
vFoundFile in filelist( GetFolderPath('MyMusic') & '\*.'& vExt )

 FileList: LOAD FileLongName,     subfield(FileLongName,'\',-1) as FileShortName,     num
(FileSize(FileLongName),'# ### ### ###',',',' ') as FileSize,     FileTime(FileLongName) as
FileTime,      Attribute(FileLongName, 'Title') as Title,     Attribute(FileLongName,
'Author') as Author,     Attribute(FileLongName, 'Copyright') as Copyright,     Attribute
(FileLongName, 'Description') as Description,

     Attribute(FileLongName, 'Rating') as Rating,     Attribute(FileLongName, 'PlayDuration')
as PlayDuration,     Attribute(FileLongName, 'MaximumBitrate') as MaximumBitrate,
Attribute(FileLongName, 'WMFSDKVersion') as WMFSDKVersion,     Attribute(FileLongName,
'WMFSDKNeeded') as WMFSDKNeeded,     Attribute(FileLongName, 'IsVBR') as IsVBR,     Attribute
(FileLongName, 'ASFLeakyBucketPairs') as ASFLeakyBucketPairs,

     Attribute(FileLongName, 'PeakValue') as PeakValue,     Attribute(FileLongName,
'AverageLevel') as AverageLevel; LOAD @1:n as FileLongName Inline "$(vFoundFile)" (fix, no
labels); Next vFoundFile Next vExt
```

Example 4: PNG

Lo script legge tutti i metatag PNG possibili nella cartella *MyPictures*.

```
// Script to read PNG meta tags for each vExt in 'png' for each vFoundFile in filelist(
GetFolderPath('MyPictures') & '\*.'& vExt )

 FileList: LOAD FileLongName,     subfield(FileLongName,'\',-1) as FileShortName,     num
(FileSize(FileLongName),'# ### ### ###',',',' ') as FileSize,     FileTime(FileLongName) as
FileTime,      Attribute(FileLongName, 'Comment') as Comment,

     Attribute(FileLongName, 'Creation Time') as Creation_Time,     Attribute(FileLongName,
'Source') as Source,     Attribute(FileLongName, 'Title') as Title,     Attribute
(FileLongName, 'Software') as Software,     Attribute(FileLongName, 'Author') as Author,
Attribute(FileLongName, 'Description') as Description,

     Attribute(FileLongName, 'Copyright') as Copyright; LOAD @1:n as FileLongName Inline
"$(vFoundFile)" (fix, no labels); Next vFoundFile Next vExt
```

ConnectString

La funzione **ConnectString()** restituisce il nome della connessione dati attiva per le connessioni ODBC o OLE DB. La funzione restituisce una stringa vuota se non è stata eseguita alcuna istruzione **connect** o dopo un'istruzione **disconnect**.

Sintassi:

```
ConnectString()
```

Esempi e risultati:

Esempi di script

Esempio	Risultato

`LIB CONNECT TO 'Tutorial ODBC';`
`ConnectString:`
`Load ConnectString() as ConnectString AutoGenerate 1;`
	Restituisce 'Tutorial ODBC' nel campo ConnectString.
	
	In questo esempio si presuppone che sia disponibile una connessione dati denominata Tutorial ODBC.

FileBaseName

La funzione **FileBaseName** restituisce una stringa contenente il nome del file tabella in corso di lettura, senza percorso o estensione.

Sintassi:

```
FileBaseName()
```

Esempi e risultati:

Esempi di script

Esempio	Risultato
`LOAD *, filebasename( ) as X from C:\UserFiles\abc.txt`	Restituisce 'abc' nel campo X in ogni record letto.

FileDir

La funzione **FileDir** restituisce una stringa contenente il percorso della directory del file tabella in corso di lettura.

Sintassi:

```
FileDir()
```

Questa funzione supporta solo le connessioni dati della cartella in modalità standard.

Esempi e risultati:

Esempi di script

Esempio	Risultato
`Load *, filedir( ) as X from C:\UserFiles\abc.txt`	Restituisce 'C:\UserFiles' nel campo X in ogni record letto.

FileExtension

La funzione **FileExtension** restituisce una stringa contenente l'estensione del file tabella in corso di lettura.

Sintassi:

```
FileExtension()
```

Esempi e risultati:

Esempi di script

Esempio	Risultato
`LOAD *, FileExtension( ) as X from C:\UserFiles\abc.txt`	Restituisce 'txt' nel campo X in ogni record letto.

FileName

La funzione **FileName** restituisce una stringa contenente il nome del file tabella in corso di lettura, senza percorso ma includendo l'estensione.

Sintassi:

```
FileName()
```

Esempi e risultati:

Esempi di script

Esempio	Risultato
`LOAD *, FileName( ) as X from C:\UserFiles\abc.txt`	Restituisce 'abc.txt' nel campo X in ogni record letto.

FilePath

La funzione **FilePath** restituisce una stringa contenente il percorso completo del file tabella in corso di lettura.

Sintassi:

```
FilePath()
```

Questa funzione supporta solo le connessioni dati della cartella in modalità standard.

Esempi e risultati:

Esempi di script

Esempio	Risultato
`Load *, FilePath( ) as X from C:\UserFiles\abc.txt`	Restituisce 'C:\UserFiles\abc.txt' nel campo X in ogni record letto.

FileSize

La funzione **FileSize** restituisce un valore intero contenente le dimensioni in byte del file filename oppure, se non viene specificato alcun filename, del file tabella in corso di lettura.

Sintassi:

```
FileSize([filename])
```

Argomenti:

Argomenti

Argomento	Descrizione
filename	Il nome di un file, con il relativo percorso se necessario, come connessione dati di una cartella o di un file Web. Se non si specifica il nome di un file, viene utilizzato il file tabella attualmente letto. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:\data* • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** *data* • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio:** *http://www.qlik.com*

Esempi e risultati:

Esempi di script

Esempio	Risultato
`LOAD *, FileSize( ) as X from abc.txt;`	Restituisce la dimensione del file specificato (abc.txt) come un numero intero nel campo X in ogni record letto.
`FileSize( 'lib://DataFiles/xyz.xls' )`	Restituisce la dimensione del file xyz.xls.

FileTime

La funzione **FileTime** restituisce un indicatore temporale in UTC per la data e l'ora dell'ultima modifica del file filename. Se non viene specificato alcun filename, la funzione farà riferimento al file tabella in corso di lettura.

Sintassi:

```
FileTime([ filename ])
```

Argomenti:

Argomenti

Argomento	Descrizione
filename	Il nome di un file, con il relativo percorso se necessario, come connessione dati di una cartella o di un file Web. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:\data* • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** *data* • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio:** *http://www.qlik.com*

Esempi e risultati:

Esempi di script

Esempio	Risultato
`LOAD *, FileTime( ) as X from abc.txt;`	Restituisce la data e l'ora dell'ultima modifica del file (abc.txt) come indicazione data/ora nel campo X in ogni record letto.
FileTime('xyz.xls')	Restituisce l'indicazione di data e ora dell'ultima modifica del file xyz.xls.

GetFolderPath

La funzione **GetFolderPath** restituisce il valore della funzione Microsoft Windows *SHGetFolderPath*. Questa funzione utilizza come input il nome di una cartella Microsoft Windows e restituisce il percorso completo della cartella.

Questa funzione non è supportata in modalità standard. .

Sintassi:

```
GetFolderPath(foldername)
```

Argomenti:

Argomenti

Argomento	Descrizione
foldername	Nome della cartella Microsoft Windows. Il nome della cartella non deve contenere spazi. Occorre rimuovere qualsiasi spazio presente nel nome della cartella in Windows Explorer. Esempi: *MyMusic* *MyDocuments*

Esempi e risultati:

Lo scopo di questo esempio è ottenere i percorsi delle seguenti cartelle Microsoft Windows: *MyMusic*, *MyPictures* e *Windows*. Aggiungere lo script di esempio all'app e ricaricarla.

```
LOAD
 GetFolderPath('MyMusic') as MyMusic,
 GetFolderPath('MyPictures') as MyPictures,
 GetFolderPath('Windows') as Windows
AutoGenerate 1;
```

Una volta ricaricata l'app, i campi *MyMusic*, *MyPictures* e *Windows* vengono aggiunti al modello dati. Ogni campo contiene il percorso della cartella definita nell'input. Ad esempio:

- *C:\Users\smu\Music* for the folder *MyMusic*
- *C:\Users\smu\Pictures* for the folder *MyPictures*
- *C:\Windows* for the folder *Windows*

QvdCreateTime

Questa funzione di script restituisce l'intestazione XML relativa alla data e ora da un file QVD, se disponibile, altrimenti restituisce NULL. Nella data e ora, l'ora è fornita in UTC.

Sintassi:

```
QvdCreateTime(filename)
```

Argomenti:

Argomenti

Argomento	Descrizione
filename	Il nome di un file QVD, includendo il percorso, se necessario come connessione dati della cartella o Web. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:\data* • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** *data* • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio:** *http://www.qlik.com*

Esempio:

```
QvdCreateTime('MyFile.qvd')

QvdCreateTime('C:\MyDir\MyFile.qvd')

QvdCreateTime('lib://DataFiles/MyFile.qvd')
```

QvdFieldName

La funzione script restituisce il nome del numero campo **fieldno** in un file QVD. Se il campo non esiste, viene restituito NULL.

Sintassi:

```
QvdFieldName(filename , fieldno)
```

Argomenti:

Argomenti

Argomento	Descrizione
filename	Il nome di un file QVD, includendo il percorso, se necessario come connessione dati della cartella o Web. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:**data*\ • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** *data*\ • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio:** *http://www.qlik.com*
fieldno	Il numero del campo all'interno della tabella contenuta nel file QVD.

Esempi:

```
QvdFieldName ('MyFile.qvd', 5)

QvdFieldName ('C:\MyDir\MyFile.qvd', 5)

QvdFieldName ('lib://DataFiles/MyFile.qvd', 5)
```

Tutti e tre gli esempi restituiscono il nome del quinto campo della tabella contenuta nel file QVD.

QvdNoOfFields

Questa funzione dello script restituisce il numero di campi all'interno di file QVD.

Sintassi:

```
QvdNoOfFields(filename)
```

Argomenti:

Argomenti

Argomento	Descrizione
filename	Il nome di un file QVD, includendo il percorso, se necessario come connessione dati della cartella o Web. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:\data* • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** *data* • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio:** *http://www.qlik.com*

Esempi:

```
QvdNoOfFields ('MyFile.qvd')

QvdNoOfFields ('C:\MyDir\MyFile.qvd')

QvdNoOfFields ('lib://DataFiles/MyFile.qvd')
```

QvdNoOfRecords

Esempio: Questa funzione dello script restituisce il numero di record attualmente presente in un file QVD.

Sintassi:

```
QvdNoOfRecords(filename)
```

Argomenti:

Argomenti

Argomento	Descrizione
filename	Il nome di un file QVD, includendo il percorso, se necessario come connessione dati della cartella o Web. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:\data* • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** *data* • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio:** *http://www.qlik.com*

Esempi:

```
QvdNoOfRecords ('MyFile.qvd')

QvdNoOfRecords ('C:\MyDir\MyFile.qvd')

QvdNoOfRecords ('lib://DataFiles/MyFile.qvd')
```

QvdTableName

Questa funzione di script restituisce il nome della tabella memorizzata in un file QVD.

Sintassi:

```
QvdTableName(filename)
```

Argomenti:

Argomenti

Argomento	Descrizione
filename	Il nome di un file QVD, includendo il percorso, se necessario come connessione dati della cartella o Web. **Esempio:** *'lib://Table Files/'* Nella modalità di creazione degli script legacy sono supportati anche i seguenti formati di percorso: • assoluto **Esempio:** *c:\data* • relativo alla directory di lavoro dell'app Qlik Sense **Esempio:** *data* • indirizzo dell'URL (HTTP o FTP), che punta a una posizione in Internet o su una Intranet **Esempio:** *http://www.qlik.com*

Esempi:

```
QvdTableName ('MyFile.qvd')
QvdTableName ('C:\MyDir\MyFile.qvd')
QvdTableName ('lib://data\MyFile.qvd')
```

5.11 Funzioni finanziarie

Le funzioni finanziarie possono essere utilizzate nello script di caricamento dei dati e nelle espressioni grafiche per calcolare i pagamenti e i tassi di interesse.
Per tutti gli argomenti, le uscite sono rappresentate da numeri negativi. Il denaro in entrata viene rappresentato da numeri positivi.
Di seguito è riportato un elenco degli argomenti utilizzati nelle funzioni finanziarie (tranne quelli che iniziano con **range**-).

*Per tutte le funzioni finanziarie è importante essere coerenti quando si specificano le unità per **rate** e **nper**. Se si eseguono pagamenti mensili su un prestito quinquennale al tasso di interesse annuale del 6%, utilizzare 0,005 (6%/12) per **rate** e 60 (5*12) per **nper**. Se si effettuano pagamenti annuali sullo stesso prestito, utilizzare il 6% per **rate** e il 5 per **nper**.*

Panoramica sulle funzioni finanziarie

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

FV

Questa funzione restituisce il valore futuro di un investimento basato su pagamenti periodici e costanti e un tasso di interesse annuale semplice.

```
FV (rate, nper, pmt [ ,pv [ , type ] ])
```

nPer

Questa funzione restituisce il numero dei periodi per un investimento basato su pagamenti periodici e costanti con un tasso di interesse costante.

```
nPer (rate, pmt, pv [ ,fv [ , type ] ])
```

Pmt

Questa funzione restituisce il pagamento di un prestito basato su versamenti periodici e costanti e un tasso di interesse costante. Non può essere modificato per tutta la durata dell'annualità. Un pagamento viene indicato con un numero negativo, ad esempio, -20.

```
Pmt (rate, nper, pv [ ,fv [ , type ] ] )
```

PV

Questa funzione restituisce il valore corrente di un investimento.

```
PV (rate, nper, pmt [ ,fv [ , type ] ])
```

Rate

Questa funzione restituisce il tasso di interesse per periodo di un anno. Il risultato presenta un formato numerico predefinito **Fix** con due decimali e %.

```
Rate (nper, pmt , pv [ ,fv [ , type ] ])
```

BlackAndSchole

Il modello Black and Scholes è un modello matematico per gli strumenti derivati del mercato finanziario. La formula consente di calcolare il valore teorico di una stock option. In Qlik Sense la funzione **BlackAndSchole** restituisce il valore in base alla formula Black and Scholes non modificata (opzioni in stile europeo).

```
BlackAndSchole(strike , time_left , underlying_price , vol , risk_free_rate ,
type)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
strike	Il prezzo futuro di acquisto dell'azione.
time_left	Il numero di intervalli di tempo rimanenti.
underlying_price	Il valore attuale dell'azione.
vol	La volatilità (del prezzo dell'azione) espressa come percentuale in forma decimale, per periodo di tempo.
risk_free_rate	Il tasso senza rischi espresso come percentuale in forma decimale, per periodo di tempo.
call_or_put	Il tipo di opzione: 'c', 'call' o qualsiasi valore numerico diverso da zero per le opzioni di chiamata 'p', 'put' o 0 per e le opzioni di inserimento.

Limiti:

Il valore di strike, time_left e underlying_price deve essere >0.

Il valore di vol e risk_free_rate deve essere: <0 o >0.

Esempi e risultati:

Esempi di script

Esempio	Risultato
`BlackAndSchole(130, 4, 68.5, 0.4, 0.04, 'call')` Consente di calcolare il prezzo teorico di un'opzione per acquistare un'azione che ha attualmente un valore pari a 68,5, a un valore di 130 in 4 anni. La formula utilizza una volatilità di 0,4 (40%) all'anno e un tasso di interesse senza rischi di 0,04 (4%).	Restituisce 11,245

FV

Questa funzione restituisce il valore futuro di un investimento basato su pagamenti periodici e costanti e un tasso di interesse annuale semplice.

Sintassi:

```
FV(rate, nper, pmt [ ,pv [ , type ] ])
```

Tipo di dati restituiti: numerico. Per impostazione predefinita, il risultato sarà formattato come valuta..

Argomenti:

Argomenti

Argomento	Descrizione
rate	Il tasso di interesse per periodo.
nper	Il numero totale dei periodi di pagamento in un'annualità.
pmt	Il pagamento effettuato per ogni periodo. Non può essere modificato per tutta la durata dell'annualità. Un pagamento viene indicato con un numero negativo, ad esempio, -20.
pv	Il valore attuale, o l'ammontare della somma forfettaria, che verrà saldato da una serie di pagamenti futuri. Se **pv** viene omesso, viene utilizzato 0 (zero).
type	Deve essere 0 se i pagamenti sono in scadenza alla fine del periodo oppure 1 se sono in scadenza all'inizio del periodo. Se **type** viene omesso, viene utilizzato 0.

Esempi e risultati:

Esempio di script

Esempio	Risultato
Un nuovo elettrodomestico viene pagato con 36 rate mensili di $20. Il tasso di interesse annuo è del 6%. La fattura deve essere pagata alla fine di ogni mese. Qual è il valore totale investito, al momento del pagamento dell'ultima fattura? `FV(0.005,36,-20)`	Restituisce $786.72

nPer

Questa funzione restituisce il numero dei periodi per un investimento basato su pagamenti periodici e costanti con un tasso di interesse costante.

Sintassi:

```
nPer(rate, pmt, pv [ ,fv [ , type ] ])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
rate	Il tasso di interesse per periodo.
nper	Il numero totale dei periodi di pagamento in un'annualità.

Argomento	Descrizione
pmt	Il pagamento effettuato per ogni periodo. Non può essere modificato per tutta la durata dell'annualità. Un pagamento viene indicato con un numero negativo, ad esempio, -20.
pv	Il valore attuale, o l'ammontare della somma forfettaria, che verrà saldato da una serie di pagamenti futuri. Se **pv** viene omesso, viene utilizzato 0 (zero).
fv	Il valore futuro, o un saldo di cassa, che si desidera raggiungere dopo avere effettuato l'ultimo pagamento. Se **fv** viene omesso, viene utilizzato 0.
type	Deve essere 0 se i pagamenti sono in scadenza alla fine del periodo oppure 1 se sono in scadenza all'inizio del periodo. Se **type** viene omesso, viene utilizzato 0.

Esempi e risultati:

Esempio di script

Esempio	Risultato
Si desidera vendere un elettrodomestico a rate mensili di $20. Il tasso di interesse annuo è del 6%. La fattura deve essere pagata alla fine di ogni mese. Quanti periodi sono richiesti se il valore del denaro che si è ricevuto dopo l'ultima fattura pagata deve corrispondere a $800? `nPer(0.005,-20,0,800)`	Restituisce 36,56

Pmt

Questa funzione restituisce il pagamento di un prestito basato su versamenti periodici e costanti e un tasso di interesse costante. Non può essere modificato per tutta la durata dell'annualità. Un pagamento viene indicato con un numero negativo, ad esempio, -20.

```
Pmt(rate, nper, pv [ ,fv [ , type ] ] )
```

Tipo di dati restituiti: numerico. Per impostazione predefinita, il risultato sarà formattato come valuta..

Per trovare la quantità totale pagata per la durata del prestito, moltiplicare il valore di **pmt** restituito da **nper**.

Argomenti:

Argomenti

Argomento	Descrizione
rate	Il tasso di interesse per periodo.
nper	Il numero totale dei periodi di pagamento in un'annualità.
pv	Il valore attuale, o l'ammontare della somma forfettaria, che verrà saldato da una serie di pagamenti futuri. Se **pv** viene omesso, viene utilizzato 0 (zero).

Argomento	Descrizione
fv	Il valore futuro, o un saldo di cassa, che si desidera raggiungere dopo avere effettuato l'ultimo pagamento. Se **fv** viene omesso, viene utilizzato 0.
type	Deve essere 0 se i pagamenti sono in scadenza alla fine del periodo oppure 1 se sono in scadenza all'inizio del periodo. Se **type** viene omesso, viene utilizzato 0.

Esempi e risultati:

Esempi di script

Esempio	Risultato
La seguente formula restituisce i pagamenti mensili su un prestito di $20.000 con tasso di interesse al 10%, da saldare in 8 mensilità: `Pmt(0.1/12,8,20000)`	Restituisce -$2,594.66
Per lo stesso prestito, se i pagamenti avvengono a inizio mensilità, il totale sarà: `Pmt(0.1/12,8,20000,0,1)`	Restituisce -$2,573.21

PV

Questa funzione restituisce il valore corrente di un investimento.

```
PV(rate, nper, pmt [ ,fv [ , type ] ])
```

Tipo di dati restituiti: numerico. Per impostazione predefinita, il risultato sarà formattato come valuta..

Il presente valore corrisponde all'importo totale attuale di una serie di pagamenti futuri. Ad esempio, quando si richiede un prestito, l'entità del prestito è il valore attuale di chi concede il prestito.

Argomenti:

Argomenti

Argomento	Descrizione
rate	Il tasso di interesse per periodo.
nper	Il numero totale dei periodi di pagamento in un'annualità.
pmt	Il pagamento effettuato per ogni periodo. Non può essere modificato per tutta la durata dell'annualità. Un pagamento viene indicato con un numero negativo, ad esempio, -20.
fv	Il valore futuro, o un saldo di cassa, che si desidera raggiungere dopo avere effettuato l'ultimo pagamento. Se **fv** viene omesso, viene utilizzato 0.
type	Deve essere 0 se i pagamenti sono in scadenza alla fine del periodo oppure 1 se sono in scadenza all'inizio del periodo. Se **type** viene omesso, viene utilizzato 0.

Esempi e risultati:

Esempio di script

Esempio	Risultato
Qual è il valore attuale di un debito quando occorre pagare $100 alla fine di ogni mese in un periodo di 5 anni, con un tasso di interesse del 7%? `PV(0.07/12,12*5,-100,0,0)`	Restituisce $5,050.20

Rate

Questa funzione restituisce il tasso di interesse per periodo di un anno. Il risultato presenta un formato numerico predefinito **Fix** con due decimali e %.

Sintassi:

```
Rate(nper, pmt , pv [ ,fv [ , type ] ])
```

Tipo di dati restituiti: numerico.

Il valore di **rate** viene calcolato mediante ripetizione e può avere zero o più soluzioni. Se i risultati di **rate** successivi non convergono, verrà restituito un valore NULL.

Argomenti:

Argomenti

Argomento	Descrizione
nper	Il numero totale dei periodi di pagamento in un'annualità.
pmt	Il pagamento effettuato per ogni periodo. Non può essere modificato per tutta la durata dell'annualità. Un pagamento viene indicato con un numero negativo, ad esempio, -20.
pv	Il valore attuale, o l'ammontare della somma forfettaria, che verrà saldato da una serie di pagamenti futuri. Se **pv** viene omesso, viene utilizzato 0 (zero).
fv	Il valore futuro, o un saldo di cassa, che si desidera raggiungere dopo avere effettuato l'ultimo pagamento. Se **fv** viene omesso, viene utilizzato 0.
type	Deve essere 0 se i pagamenti sono in scadenza alla fine del periodo oppure 1 se sono in scadenza all'inizio del periodo. Se **type** viene omesso, viene utilizzato 0.

Esempi e risultati:

Esempio di script

Esempio	Risultato
Qual è il tasso di interesse di un prestito in cinque anni di $10.000 annuali con pagamenti mensili di $300? `Rate(60,-300,10000)`	Restituisce 2.00%

5.12 Funzioni di formattazione

Le funzioni di formattazione impongono il formato di visualizzazione alle espressioni o ai campi numerici di input. A seconda del tipo di dati, è possibile specificare i caratteri per il separatore decimale, il separatore delle migliaia e così via.

Tutte le funzioni restituiscono un valore duale che riporta sia la stringa che il valore numerico, ma possono essere interpretate come una conversione da numero a stringa. **Dual()** è un caso speciale, tuttavia le altre funzioni di formattazione generano una stringa che rappresenta il numero a partire dal valore numerico dell'espressione di input.

Le funzioni di interpretazioni invece si comportano nel modo opposto: le espressioni delle stringhe vengono valutate come numeri specificando il formato del numero risultante.

Le funzioni possono essere utilizzate sia negli script di caricamento dei dati che nelle espressioni grafiche.

Tutte le rappresentazioni numeriche vengono fornite con un punto decimale come separatore decimale.

Panoramica sulle funzioni di formattazione

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

ApplyCodepage

ApplyCodepage() applica il set di caratteri di una pagina codici differente al campo o al testo dichiarato nell'espressione. L'argomento **codepage** deve essere in formato numerico.

```
ApplyCodepage (text, codepage)
```

Date

Date() consente di formattare un'espressione come una data utilizzando il formato impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo oppure una stringa di formattazione, se disponibile.

```
Date (number[, format])
```

Dual

Dual() combina un numero e una stringa in un unico record in modo che la rappresentazione numerica del record possa essere utilizzata per l'ordinamento e il calcolo, mentre il valore della stringa possa essere utilizzato per la visualizzazione.

```
Dual (text, number)
```

Interval

Interval() consente di formattare un numero come un intervallo di tempo utilizzando il formato impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo oppure una stringa di formattazione, se disponibile.

```
Interval (number[, format])
```

Money

Money() consente di formattare un'espressione numericamente come valore di valuta nel formato delle variabili di sistema impostato nello script di caricamento dei dati o nel sistema operativo, a meno che non vengano forniti una stringa di formattazione e separatori decimali e delle migliaia opzionali.

```
Money (number[, format[, dec_sep [, thou_sep]]])
```

Num

Num() formatta un numero, ovvero converte il valore numerico dell'input per visualizzare il testo usando il formato specificato nel secondo parametro. Se il secondo parametro viene omesso, utilizza i separatori decimali e delle migliaia impostati nello script di caricamento dei dati. I simboli dei separatori decimali e delle migliaia personalizzati sono parametri opzionali.

```
Num (number[, format[, dec_sep [, thou_sep]]])
```

Time

Time() consente di formattare un'espressione come valore ora nel formato dell'ora impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo, a meno che non venga fornita una stringa di formattazione.

```
Time (number[, format])
```

Timestamp

TimeStamp() consente di formattare un'espressione come valore data e ora nel formato dell'indicatore temporale impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo, a meno che non venga fornita una stringa di formattazione.

```
Timestamp (number[, format])
```

Vedere anche:

p *Funzioni di interpretazione (page 1255)*

ApplyCodepage

ApplyCodepage() applica il set di caratteri di una pagina codici differente al campo o al testo dichiarato nell'espressione. L'argomento **codepage** deve essere in formato numerico.

Sebbene ApplyCodepage possa essere utilizzato nelle espressioni grafiche, viene utilizzato più di frequente come funzione di script nell'editor caricamento dati. Ad esempio, dal momento che si caricano file che potrebbero essere stati salvati con set di caratteri differenti non controllati dall'utente, è possibile applicare la pagina codici che rappresenta il set di caratteri richiesto.

Sintassi:

```
ApplyCodepage(text, codepage)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
text	Campo o testo a cui si desidera applicare una pagina codici differente, data dall'argomento **codepage**.
codepage	Numero che rappresenta la pagina codici da applicare al campo o all'espressione data da **text**.

Esempi e risultati:

Esempi di script

Esempio	Risultato
`LOAD ApplyCodepage (ROWX,1253) as GreekProduct, ApplyCodepage (ROWY, 1255) as HebrewProduct, ApplyCodepage (ROWZ, 65001) as EnglishProduct; SQL SELECT ROWX, ROWY, ROWZ From Products;`	Durante il caricamento da SQL la sorgente potrebbe presentare una combinazione di set di caratteri differenti: cirillico, ebraico e così via, del formato UTF-8. Questi devono essere caricati riga per riga, applicano una pagina codici differente per ciascuna riga. Il valore **codepage** 1253 rappresenta il set di caratteri greco Windows, il valore 1255 rappresenta l'ebraico e il valore 65001 rappresenta i caratteri UTF-8 latini standard.

Vedere anche: *Set di caratteri (page 166)*

Date

Date() consente di formattare un'espressione come una data utilizzando il formato impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo oppure una stringa di formattazione, se disponibile.

Sintassi:

```
Date(number[, format])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
number	Il numero da formattare.
format	Stringa che descrive il formato della stringa risultante. Se non viene fornita la stringa di formattazione, verrà utilizzato il formato della data impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo.

Esempi e risultati:
Gli esempi riportati di seguito presuppongono l'utilizzo delle seguenti impostazioni predefinite:

- Impostazione della data 1: YY-MM-DD
- Impostazione della data 2: M/D/YY

Esempio:

```
Date( A )
```
dove A=35648

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	97-08-06	8/6/97
Numero:	35648	35648

Esempio:

```
Date( A, 'YY.MM.DD' )
```
dove A=35648

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	97.08.06	97.08.06
Numero:	35648	35648

Esempio:

```
Date( A, 'DD.MM.YYYY' )
```
dove A=35648.375

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	06.08.1997	06.08.1997
Numero:	35648.375	35648.375

Esempio:

```
Date( A, 'YY.MM.DD' )
```
dove A=8/6/97

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	NULL (nessun dato)	97.08.06
Numero:	NULL	35648

Dual

Dual() combina un numero e una stringa in un unico record in modo che la rappresentazione numerica del record possa essere utilizzata per l'ordinamento e il calcolo, mentre il valore della stringa possa essere utilizzato per la visualizzazione.

Sintassi:

```
Dual(text, number)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
text	Il valore della stringa da usare unitamente all'argomento del numero.
number	Il numero da usare unitamente alla stringa nell'argomento della stringa.

In Qlik Sense, tutti i valori di campo sono potenzialmente valori duali. Questo significa che i valori del campo possono avere sia un valore numerico che un valore testuale. Un esempio è una data che potrebbe avere un valore numerico di 40908 e la rappresentazione testuale '2011-12-31'.

Quando diversi dati letti da un campo presentano rappresentazioni di stringhe differenti per la stessa rappresentazione numerica valida, tutti condividono la prima rappresentazione a stringa rilevata.

*La funzione **dual** viene in genere utilizzata all'inizio dello script, prima che venga eseguita la lettura di altri dati nel campo interessato, affinché sia possibile creare la prima rappresentazione di stringa, che verrà poi visualizzata nelle caselle di filtro.*

Esempi e risultati:

Esempi di script

Esempio	Descrizione
Aggiungere gli esempi seguenti allo script ed eseguirlo. `Load dual ( NameDay,NumDay ) as DayOfWeek inline [ NameDay,NumDay Monday,0 Tuesday,1 Wednesday,2 Thursday,3 Friday,4 Saturday,5 Sunday,6 ];`	Il campo DayOfWeek può essere utilizzato in una visualizzazione, come una dimensione, ad esempio in una tabella in cui i giorni della settimana sono ordinati automaticamente nella sequenza numerica corretta, invece che in ordine alfabetico.
`Load Dual('Q' & Ceil (Month(Now())/3), Ceil(Month(Now ())/3)) as Quarter AutoGenerate 1;`	Questo esempio individua il trimestre attuale. Viene visualizzato come Q1 quando la funzione **Now()** viene eseguita nei primi tre mesi dell'anno, come Q2 per i secondi tre mesi e così via. Tuttavia, quando viene utilizzato nell'ordinamento, il campo Quarter si comporterà come il relativo valore numerico: da 1 a 4.
`Dual('Q' & Ceil (Month(Date)/3), Ceil(Month(Date)/3)) as Quarter`	Come nell'esempio precedente, il campo Quarter viene creato con i valori di testo da 'Q1' a 'Q4' a cui vengono assegnati i valori numerici da 1 a 4. Per poter utilizzarli nello script, i valori di Date devono venire caricati.
`Dual(WeekYear(Date) & '-W' & Week(Date), WeekStart(Date)) as YearWeek`	Questo esempio creerà un campo YearWeek con i valori testuali in forma di '2012-W22' e allo stesso tempo assegnerà un valore numerico corrispondente al numero della data del primo giorno della settimana, ad esempio: 41057. Per poter utilizzarli nello script, i valori di Date devono venire caricati.

Interval

Interval() consente di formattare un numero come un intervallo di tempo utilizzando il formato impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo oppure una stringa di formattazione, se disponibile.

Gli intervalli possono essere formattati come un'ora, come giorni oppure come combinazione di giorni, ore, minuti, secondi e frazioni di secondo.

Sintassi:

```
Interval(number[, format])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
number	Il numero da formattare.
format	Stringa che descrive come formattare la stringa dell'intervallo risultante. Se viene omessa, si utilizzerà il formato della data breve, il formato dell'ora e il separatore decimale impostati nel sistema operativo.

Esempi e risultati:
Gli esempi riportati di seguito presuppongono l'utilizzo delle seguenti impostazioni predefinite:

- Impostazione del formato della data 1: YY-MM-DD
- Impostazione del formato della data 2: hh:mm:ss
- Separatore decimale dei numeri: .

Tabella dei risultati

Esempio	Stringa	Numero
`Interval( A )` dove A=0,375	09:00:00	0.375
`Interval( A )` dove A=1,375	33:00:00	1.375
`Interval( A, 'D hh:mm' )` dove A=1,375	1 09:00	1.375
`Interval( A-B, 'D hh:mm' )` dove A=97-08-06 09:00:00 e B=96-08-06 00:00:00	365 09:00	365.375

Money

Money() consente di formattare un'espressione numericamente come valore di valuta nel formato delle variabili di sistema impostato nello script di caricamento dei dati o nel sistema operativo, a meno che non vengano forniti una stringa di formattazione e separatori decimali e delle migliaia opzionali.

Sintassi:

```
Money(number[, format[, dec_sep[, thou_sep]]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
number	Il numero da formattare.
format	Stringa che descrive come formattare la stringa della valuta risultante.
dec_sep	Stringa che specifica il separatore decimale dei numeri.
thou_sep	Stringa che specifica il separatore delle migliaia dei numeri.

Se gli argomenti 2-4 vengono omessi, viene usato il formato della valuta impostato nel sistema operativo.

Esempi e risultati:
Gli esempi riportati di seguito presuppongono l'utilizzo delle seguenti impostazioni predefinite:

- Impostazione MoneyFormat 1: kr ##0,00, MoneyThousandSep' '
- Impostazione MoneyFormat 2: $ #,##0.00, MoneyThousandSep','

Esempio:

```
Money( A )
```
dove A=35648

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	kr 35 648,00	$ 35,648.00
Numero:	35648.00	35648.00

Esempio:

```
Money( A, '#,##0 ¥', '.' , ',' )
```
dove A=3564800

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	3,564,800 ¥	3,564,800 ¥
Numero:	3564800	3564800

Num

Num() formatta un numero, ovvero converte il valore numerico dell'input per visualizzare il testo usando il formato specificato nel secondo parametro. Se il secondo parametro viene omesso, utilizza i separatori decimali e delle migliaia impostati nello script di caricamento dei dati. I simboli dei separatori decimali e delle migliaia personalizzati sono parametri opzionali.

Sintassi:

```
Num(number[, format[, dec_sep [, thou_sep]]])
```

Tipo di dati restituiti: duale

La funzione Num restituisce un valore duale contenente sia la stringa che il valore numerico. La funzione prende il valore numerico dell'espressione di input e genera una stringa che rappresenta il numero.

Argomenti:

Argomenti

Argomento	Descrizione
number	Il numero da formattare.
format	Stringa che specifica come formattare la stringa risultante. Se omessi, verranno utilizzati i separatori di decimali e migliaia impostati nello script di caricamento dei dati.
dec_sep	Stringa che specifica il separatore decimale dei numeri. Se omesso, viene utilizzato il valore della variabile DecimalSep impostato nello script di caricamento dei dati.
thou_sep	Stringa che specifica il separatore delle migliaia dei numeri. Se omesso, viene utilizzato il valore della variabile ThousandSep impostato nello script di caricamento dei dati.

Esempio: Espressione del grafico

Esempio:

La tabella seguente mostra i risultati quando il campo A equivale a 35648.312.

Risultati

Una	Risultato
Num(A)	35648.312 (dipende dalle variabili ambiente nello script)
Num(A, '0.0', '.')	35648.3
Num(A, '0,00', ',')	35648,31
Num(A, '#,##0.0', '.', ',')	35,648.3
Num(A, '# ##0', ',', ' ')	35 648

Esempio: Script di caricamento

Script di caricamento

Num può essere utilizzato in uno script di caricamento per formattare un numero, anche se i separatori delle migliaia e dei decimali sono già impostati nello script. Lo script di caricamento mostrato di seguito include specifici separatori delle migliaia e dei decimali, ma utilizza *Num* per formattare i dati in modi diversi.

Nell'**editor caricamento dati** creare una nuova sezione, aggiungere lo script di esempio ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

```
SET ThousandSep=','; SET DecimalSep='.'; Transactions: Load *, Num(transaction_amount) as [No formatting], Num(transaction_amount,'0') as [0], Num(transaction_amount,'#,##0') as [#,##0], Num(transaction_amount,'# ###,00') as [# ###,00], Num(transaction_amount,'# ###,00',',',' ') as [# ###,00 , ',' , ' '], Num(transaction_amount,'#,###.00','.',',') as [#,###.00 , '.' , ','], Num(transaction_amount,'$#,###.00') as [$#,###.00], ; Load * Inline [ transaction_id, transaction_date, transaction_amount, transaction_quantity, discount, customer_id, size, color_code 3750, 20180830, 12423.56, 23, 0,2038593, L, Red 3751, 20180907, 5356.31, 6, 0.1, 203521, m, orange 3752, 20180916, 15.75, 1, 0.22, 5646471, S, blue 3753, 20180922, 1251, 7, 0, 3036491, l, Black 3754, 20180922, 21484.21, 1356, 75, 049681, xs, Red 3756, 20180922, -59.18, 2, 0.333333333333333, 2038593, M, Blue 3757, 20180923, 3177.4, 21, .14, 203521, XL, Black ];
```

Tabella Qlik Sense che mostra i risultati di diversi utilizzi della funzione *Num* nello script di caricamento. La quarta colonna della tabella mostra un uso errato della formattazione, a scopo esemplificativo.

Nessuna formattazione	0	#,##0	# ###,00	# ###,00 , ',' , ''	#,###.00 , '.' , ','	$#,###.00
-59.18	-59	-59	-59###,00	-59,18	-59.18	$-59,18
15.75	16	16	16###,00	15,75	15.75	$15,75
1251	1251	1,251	1251###,00	1 251,00	1,251.00	$1,251.00
3177.4	3177	3,177	3177###,00	3 177,40	3,177.40	$3,177.40
5356.31	5356	5,356	5356###,00	5 356,31	5,356.31	$5,356.31
12423.56	12424	12,424	12424###,00	12 423,56	12,423.56	$12,423.56
21484.21	21484	21,484	21484###,00	21 484,21	21,484.21	$21,484.21

Esempio: Script di caricamento

Script di caricamento

Num può essere utilizzato in uno script di caricamento per formattare un numero come percentuale.

Nell'**editor caricamento dati** creare una nuova sezione, aggiungere lo script di esempio ed eseguirlo. Aggiungere quindi a un foglio nell'app almeno i campi elencati nella colonna dei risultati per visualizzare il risultato.

```
SET ThousandSep=','; SET DecimalSep='.'; Transactions: Load *, Num(discount,'#,##0%') as
[Discount #,##0%] ; Load * Inline [ transaction_id, transaction_date, transaction_amount,
transaction_quantity, discount, customer_id, size, color_code 3750, 20180830, 12423.56, 23,
0,2038593, L, Red 3751, 20180907, 5356.31, 6, 0.1, 203521, m, orange 3752, 20180916, 15.75, 1,
0.22, 5646471, S, blue 3753, 20180922, 1251, 7, 0, 3036491, l, Black 3754, 20180922, 21484.21,
1356, 75, 049681, xs, Red 3756, 20180922, -59.18, 2, 0.333333333333333, 2038593, M, Blue 3757,
20180923, 3177.4, 21, .14, 203521, XL, Black ];
```

Tabella Qlik Sense che mostra i risultati della funzione *Num* utilizzata nello script di caricamento per formattare le percentuali.

Sconto	Sconto #,##0%
0.333333333333333	33%
0.22	22%
0	0%
.14	14%
0.1	10%
0	0%
75	7,500%

Time

Time() consente di formattare un'espressione come valore ora nel formato dell'ora impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo, a meno che non venga fornita una stringa di formattazione.

Sintassi:

```
Time(number[, format])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
number	Il numero da formattare.
format	Stringa che descrive come formattare la stringa dell'ora risultante. Se viene omessa, si utilizzerà il formato della data breve, il formato dell'ora e il separatore decimale impostati nel sistema operativo.

Esempi e risultati:
Gli esempi riportati di seguito presuppongono l'utilizzo delle seguenti impostazioni predefinite:

- Impostazione del formato dell'ora 1: hh:mm:ss
- Impostazione del formato dell'ora 2: hh.mm.ss

Esempio:

```
Time( A )
```
dove A=0,375

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	09:00:00	09.00.00
Numero:	0.375	0.375

Esempio:

```
Time( A )
```
dove A=35648,375

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	09:00:00	09.00.00
Numero:	35648.375	35648.375

Esempio:

```
Time( A, 'hh-mm' )
```
dove A=0,99999

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	23-59	23-59
Numero:	0.99999	0.99999

Timestamp

TimeStamp() consente di formattare un'espressione come valore data e ora nel formato dell'indicatore temporale impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo, a meno che non venga fornita una stringa di formattazione.

Sintassi:

```
Timestamp(number[, format])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
number	Il numero da formattare.
format	Stringa che descrive come formattare la stringa dell'indicatore temporale risultante. Se viene omessa, si utilizzerà il formato della data breve, il formato dell'ora e il separatore decimale impostati nel sistema operativo.

Esempi e risultati:
Gli esempi riportati di seguito presuppongono l'utilizzo delle seguenti impostazioni predefinite:

- Impostazione TimeStampFormat 1: YY-MM-DD hh:mm:ss
- Impostazione TimeStampFormat 2: M/D/YY hh:mm:ss

Esempio:

```
Timestamp( A )
```
dove A=35648,375

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	97-08-06 09:00:00	8/6/97 09:00:00
Numero:	35648.375	35648.375

Esempio:

```
Timestamp( A,'YYYY-MM-DD hh.mm')
```
dove A=35648

Tabella dei risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	1997-08-06 00.00	1997-08-06 00.00
Numero:	35648	35648

5.13 Funzioni numeriche generiche

Nelle seguenti funzioni numeriche generiche gli argomenti sono espressioni, dove **x** può essere interpretato come un numero reale valutato. Tutte le funzioni possono essere utilizzate sia negli script di caricamento dei dati che nelle espressioni grafiche.

Panoramica delle funzioni numeriche generiche

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

bitcount

BitCount() restituisce quanti bit nell'equivalente binario di un numero decimale sono impostati su 1, ossia la funzione restituisce il numero di bit impostati in **integer_number**, dove **integer_number** viene interpretato come numero intero a 32 bit con segno.

```
BitCount(integer_number)
```

div

Div() restituisce la parte intera della divisione aritmetica del primo argomento per il secondo argomento. Entrambi i parametri vengono interpretati come numeri reali, ossia, non devono essere numeri interi.

```
Div (integer_number1, integer_number2)
```

fabs

Fabs() restituisce il valore assoluto di **x**. Il risultato è un numero positivo.

```
Fabs (x)
```

fact

Fact() restituisce il fattoriale di un numero intero positivo **x**.

```
Fact (x)
```

frac

Frac() restituisce la parte frazionaria di **x**.

```
Frac (x)
```

sign

Sign() restituisce 1, 0 o -1 a seconda che **x** sia un numero positivo, 0 o un numero negativo.

```
Sign (x)
```

Funzioni di combinazione e permutazione

combin

Combin() restituisce il numero di combinazioni di elementi **q** che può essere scelto da un gruppo di voci **p**. È rappresentata dalla formula: `Combin(p,q) = p! / q!(p-q)!` L'ordine con cui vengono selezionate le voci non è significativo.

```
Combin (p, q)
```

permut

Permut() restituisce il numero di permutazioni di elementi **q** che può essere selezionato da una serie di voci **p**. È rappresentata dalla formula: `Permut(p,q) = (p)! / (p - q)!` L'ordine con cui vengono selezionate le voci è significativo.

```
Permut (p, q)
```

Funzioni modulo

fmod

fmod() è una funzione di modulo generalizzato che restituisce la parte rimanente della divisione di un numero intero del primo argomento (il dividendo) per il secondo argomento (il divisore). Il risultato è un numero reale. Entrambi gli argomenti vengono interpretati come numeri reali, ossia, non devono essere numeri interi.

```
Fmod (a, b)
```

mod

Mod() è una funzione di modulo matematico che restituisce la parte restante non negativa di una divisione di numero intero. Il primo argomento è il dividendo il secondo argomento è il divisore ed entrambi gli argomenti devono essere valori interi.

```
Mod (integer_number1, integer_number2)
```

Funzioni di parità

even

Even() restituisce True (-1), se **integer_number** è un numero intero pari o zero. Restituisce False (0), se **integer_number** è un numero intero dispari e NULL se **integer_number** non è un numero intero.

```
Even (integer_number)
```

odd

Odd() restituisce True (-1), se **integer_number** è un numero intero dispari o zero. Restituisce False (0), se **integer_number** è un numero intero pari e NULL se **integer_number** non è un numero intero.

```
Odd (integer_number)
```

Funzioni di arrotondamento

ceil

Ceil() arrotonda per eccesso un numero al multiplo più vicino di **step** modificato in base al numero di **offset** .

```
Ceil (x[, step[, offset]])
```

floor

Floor() arrotonda per eccesso un numero al multiplo più vicino di **step** modificato in base al numero di **offset** .

```
Floor (x[, step[, offset]])
```

round

Round() restituisce il risultato dell'arrotondamento di un numero per eccesso o per difetto al multiplo più vicino di **step** modificato in base al numero di **offset** .

```
Round ( x [ , step [ , offset ]])
```

BitCount

BitCount() restituisce quanti bit nell'equivalente binario di un numero decimale sono impostati su 1, ossia la funzione restituisce il numero di bit impostati in **integer_number**, dove **integer_number** viene interpretato come numero intero a 32 bit con segno.

Sintassi:

```
BitCount(integer_number)
```

Tipo di dati restituiti: numero intero

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`BitCount ( 3 )`	3 è 11 binario, pertanto restituisce 2
`BitCount ( -1 )`	-1 è 64 uno in binario, quindi restituisce 64

Ceil

Ceil() arrotonda per eccesso un numero al multiplo più vicino di **step** modificato in base al numero di **offset** .

Confrontare con la funzione **floor**, la quale consente di arrotondare i numeri di input per difetto.

Sintassi:

```
Ceil(x[, step[, offset]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
x	Numero di input.
step	Incremento intervallo. Il valore predefinito è 1.
offset	Definisce la base dell'intervallo della fase. Il valore predefinito è 0.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Ceil(2.4 )`	Restituisce 3 In questo esempio, la dimensione della fase è uno e la base dell'intervallo della fase è 0. Gli intervalli sono ...0 < x <=1, 1 < x <= 2, **2< x <=3**, 3< x <=4...
`Ceil(4.2 )`	Restituisce 5
`Ceil(3.88 ,0.1)`	Restituisce 3,9 In questo esempio, la dimensione dell'intervallo è 0.1 e la base dell'intervallo è 0. Gli intervalli sono ... 3.7 < x <= 3.8, **3.8 < x <= 3.9**, 3.9 < x <= 4.0...
`Ceil(3.88 ,5)`	Restituisce 5
`Ceil(1.1 ,1)`	Restituisce 2
`Ceil(1.1 ,1,0.5)`	Restituisce 1,5 In questo esempio, la dimensione della fase è 1 e l'offset è 0,5. Significa che la base dell'intervallo della fase è 0,5 e non 0. Gli intervalli sono ...**0.5 < x <=1.5**, 1.5 < x <= 2.5, 2.5< x <=3.5, 3.5< x <=4.5...
`Ceil(1.1 ,1,-0.01)`	Restituisce 1,99 Gli intervalli sono ...-0.01< x <= 0.99, **0.99< x <= 1.99**, 1.99 < x <=2.99...

Combin

Combin() restituisce il numero di combinazioni di elementi **q** che può essere scelto da un gruppo di voci **p**. È rappresentata dalla formula: `Combin(p,q) = p! / q!(p-q)!` L'ordine con cui vengono selezionate le voci non è significativo.

Sintassi:

```
Combin(p, q)
```

Tipo di dati restituiti: numero intero

Limiti:

Le voci che non sono numeri interi vengono troncate.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
Quante combinazioni di 7 numeri possono essere scelte da un totale di 35 numeri a disposizione? `Combin( 35,7 )`	Restituisce 6.724.520

Div

Div() restituisce la parte intera della divisione aritmetica del primo argomento per il secondo argomento. Entrambi i parametri vengono interpretati come numeri reali, ossia, non devono essere numeri interi.

Sintassi:

```
Div(integer_number1, integer_number2)
```

Tipo di dati restituiti: numero intero

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Div( 7,2 )`	Restituisce 3
`Div( 7.1,2.3 )`	Restituisce 3
`Div( 9,3 )`	Restituisce 3
`Div( -4,3 )`	Restituisce -1
`Div( 4,-3 )`	Restituisce -1
`Div( -4,-3 )`	Restituisce 1

Even

Even() restituisce True (-1), se **integer_number** è un numero intero pari o zero. Restituisce False (0), se **integer_number** è un numero intero dispari e NULL se **integer_number** non è un numero intero.

Sintassi:

```
Even(integer_number)
```

Tipo di dati restituiti: Booleano

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Even( 3 )`	Restituisce 0, False
`Even( 2 * 10 )`	Restituisce -1, True
`Even( 3.14 )`	Restituisce NULL

Fabs

Fabs() restituisce il valore assoluto di **x**. Il risultato è un numero positivo.

Sintassi:

```
fabs(x)
```

Tipo di dati restituiti: numerico

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`fabs( 2.4 )`	Restituisce 2,4
`fabs( -3.8 )`	Restituisce 3,8

Fact

Fact() restituisce il fattoriale di un numero intero positivo **x**.

Sintassi:

```
Fact(x)
```

Tipo di dati restituiti: numero intero

Limiti:

Se il numero **x** non è un numero intero, verrà troncato. I numeri non positivi restituiranno NULL.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Fact( 1 )`	Restituisce 1
`Fact( 5 )`	Restituisce 120 (1 * 2 * 3 * 4 * 5 = 120)
`Fact( -5 )`	Restituisce NULL

Floor

Floor() arrotonda per eccesso un numero al multiplo più vicino di **step** modificato in base al numero di **offset** .

Confrontarla con la funzione **ceil**, che arrotonda per eccesso i numeri in input.

Sintassi:

```
Floor(x[, step[, offset]])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
x	Numero di input.
step	Incremento intervallo. Il valore predefinito è 1.
offset	Definisce la base dell'intervallo della fase. Il valore predefinito è 0.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Floor(2.4)`	Restituisce 2 In this example, the size of the step is 1 and the base of the step interval is 0. The intervals are ...0 <= x <1, 1 <= x < 2, **2<= x <3**, 3<= x <4....
`Floor(4.2)`	Restituisce 4

Esempi	Risultati
`Floor(3.88 ,0.1)`	Restituisce 3,8 In questo esempio la dimensione dell'intervallo è 0,1 e la base dell'intervallo è 0. Gli intervalli sono ... 3.7 <= x < 3.8, **3.8 <= x < 3.9**, 3.9 <= x < 4.0...
`Floor(3.88 ,5)`	Restituisce 0
`Floor(1.1 ,1)`	Restituisce 1
`Floor(1.1 ,1,0.5)`	Restituisce 0,5 In questo esempio, la dimensione della fase è 1 e l'offset è 0,5. Significa che la base dell'intervallo della fase è 0,5 e non 0. Gli intervalli sono ...**0.5 <= x <1.5**, 1.5 <= x < 2.5, 2.5<= x <3.5,...

Fmod

fmod() è una funzione di modulo generalizzato che restituisce la parte rimanente della divisione di un numero intero del primo argomento (il dividendo) per il secondo argomento (il divisore). Il risultato è un numero reale. Entrambi gli argomenti vengono interpretati come numeri reali, ossia, non devono essere numeri interi.

Sintassi:

```
fmod(a, b)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
a	Dividendo
b	Divisore

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`fmod( 7,2 )`	Restituisce 1
`fmod( 7.5,2 )`	Restituisce 1,5
`fmod( 9,3 )`	Restituisce 0

Esempi	Risultati
`fmod( -4,3 )`	Restituisce -1
`fmod( 4,-3 )`	Restituisce 1
`fmod( -4,-3 )`	Restituisce -1

Frac

Frac() restituisce la parte frazionaria di **x**.

La frazione viene definita in modo tale che `Frac(x ) + Floor(x ) = x`. In termini semplici questo significa che la parte frazionale di un numero positivo è la differenza tra il numero (x) e il numero intero che precede la parte frazionale.

Ad esempio: la parte frazionale di 11,43 = 11,43 - 11 = 0,43

Per un numero negativo, ad esempio -1,4, Floor(-1.4) = -2, il quale produce il seguente risultato:

La parte frazionale di -1,4 = 1,4 - (-2) = -1,4 + 2 = 0,6

Sintassi:

```
Frac(x)
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
x	Numero per cui restituire la frazione.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Frac( 11.43 )`	Restituisce 0,43
`Frac( -1.4 )`	Restituisce 0,6
Estrarre il componente temporale dalla rappresentazione numerica di un timestamp, omettendo così la data. `Time(Frac(44518.663888889))`	Restituisce 3:56:00 PM

Mod

Mod() è una funzione di modulo matematico che restituisce la parte restante non negativa di una divisione di numero intero. Il primo argomento è il dividendo il secondo argomento è il divisore ed entrambi gli argomenti devono essere valori interi.

Sintassi:

```
Mod(integer_number1, integer_number2)
```

Tipo di dati restituiti: numero intero

Limiti:

integer_number2 deve essere maggiore di 0.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Mod( 7,2 )`	Restituisce 1
`Mod( 7.5,2 )`	Restituisce NULL
`Mod( 9,3 )`	Restituisce 0
`Mod( -4,3 )`	Restituisce 2
`Mod( 4,-3 )`	Restituisce NULL
`Mod( -4,-3 )`	Restituisce NULL

Odd

Odd() restituisce True (-1), se **integer_number** è un numero intero dispari o zero. Restituisce False (0), se **integer_number** è un numero intero pari e NULL se **integer_number** non è un numero intero.

Sintassi:

```
Odd(integer_number)
```

Tipo di dati restituiti: Booleano

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Odd( 3 )`	Restituisce -1, True
`Odd( 2 * 10 )`	Restituisce 0, False
`Odd( 3.14 )`	Restituisce NULL

Permut

Permut() restituisce il numero di permutazioni di elementi **q** che può essere selezionato da una serie di voci **p**. È rappresentata dalla formula: `Permut(p,q) = (p)! / (p - q)!` L'ordine con cui vengono selezionate le voci è significativo.

Sintassi:

```
Permut(p, q)
```

Tipo di dati restituiti: numero intero

Limiti:

Gli argomenti che non sono numeri interi vengono troncati.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
In quanti modi possono essere distribuite le medaglie d'oro, di argento e di bronzo dopo una finale dei 100 metri con 8 partecipanti? `Permut( 8,3 )`	Restituisce 336

Round

Round() restituisce il risultato dell'arrotondamento di un numero per eccesso o per difetto al multiplo più vicino di **step** modificato in base al numero di **offset** .

Se il numero da arrotondare è esattamente alla metà di un intervallo, viene arrotondato per eccesso.

Sintassi:

```
Round(x[, step[, offset]])
```

Tipo di dati restituiti: numerico

> *Se si sta eseguendo l'arrotondamento di un numero a virgola mobile, si potrebbero ottenere risultati non corretti. Questi errori di arrotondamento sono dovuti al fatto che i numeri a virgola mobile vengono rappresentati da un numero finito di cifre binarie. Pertanto, i risultati vengono calcolati utilizzando un numero già arrotondato. Se questi errori di arrotondamento influiscono sul proprio lavoro, moltiplicare i numeri per convertirli in numeri interi prima dell'arrotondamento.*

Argomenti:

Argomenti

Argomento	Descrizione
x	Numero di input.
step	Incremento intervallo. Il valore predefinito è 1.
offset	Definisce la base dell'intervallo della fase. Il valore predefinito è 0.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Round(3.8 )`	Restituisce 4 In questo esempio, la dimensione della fase è uno e la base dell'intervallo della fase è 0. Gli intervalli sono ...0 <= x <1, 1 <= x < 2, 2<= x <3, **3<= x <4**...
`Round(3.8,4 )`	Restituisce 4
`Round(2.5 )`	Restituisce 3. In questo esempio, la dimensione della fase è uno e la base dell'intervallo della fase è 0. Gli intervalli sono ...0 <= x <1, 1 <= x <2, **2<= x <3**...
`Round(2,4 )`	Restituisce 4. Viene arrotondato in eccesso perché 2 è esattamente la metà dell'intervallo della fase di 4. In questo esempio, la dimensione della fase è 4 e la base dell'intervallo della fase è 0. Gli intervalli sono ...**0 <= x <4**, 4 <= x <8, 8<= x <12...
`Round(2,6 )`	Restituisce 0. Viene arrotondato per difetto perché 2 è meno della metà dell'intervallo della fase di 6. In questo esempio, la dimensione della fase è 6 e la base dell'intervallo della fase è 0. Gli intervalli sono ...**0 <= x <6**, 6 <= x <12, 12<= x <18...

Esempi	Risultati
`Round(3.88 ,0.1)`	Restituisce 3,9 In questo esempio, la dimensione della fase è 0,1 e la base dell'intervallo della fase è 0. Gli intervalli sono ... 3.7 <= x <3.8, **3.8 <= x <3.9**, 3.9 <= x < 4.0...
`Round (3.88875,1/1000)`	Restituisce 3.889 In questo esempio, la dimensione della fase è 0,001, che arrotonda il numero per eccesso e lo limita a tre cifre decimali.
`Round(3.88 ,5)`	Restituisce 5
`Round(1.1 ,1,0.5)`	Restituisce 1,5 In questo esempio, la dimensione del passo è 1 e la base dell'intervallo del passo è 0,5. Gli intervalli sono ...**0.5 <= x <1.5**, 1.5 <= x <2.5, 2.5<= x <3.5...

Sign

Sign() restituisce 1, 0 o -1 a seconda che **x** sia un numero positivo, 0 o un numero negativo.

Sintassi:

```
Sign(x)
```

Tipo di dati restituiti: numerico

Limiti:

Se non viene trovato nessun valore numerico, viene restituito NULL.

Esempi e risultati:

Esempi e risultati

Esempi	Risultati
`Sign( 66 )`	Restituisce 1
`Sign( 0 )`	Restituisce 0
`Sign( - 234 )`	Restituisce -1

5.14 Funzioni geospaziali

Queste funzioni vengono utilizzate per gestire i dati geospaziali nelle visualizzazioni delle mappe. Qlik Sense segue le specifiche GeoJSON per i dati geospaziali e supporta i tipi di geometria seguenti:

- Point
- Linestring
- Polygon
- Multipolygon

Per ulteriori informazioni sulle specifiche GeoJSON, vedere:
≤ GeoJSON.org

Panoramica delle funzioni geospaziali

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Sono disponibili due categorie di funzioni geospaziali: funzioni di aggregazione e funzioni di non aggregazione.

Le funzioni di aggregazione prendono un set geometrico (punti o aree) come input e restituiscono una singola geometria. Ad esempio, più aree possono essere unite e sulla mappa può essere disegnato un singolo limite per l'aggregazione.

Una funzione di non aggregazione prende una singola geometria e restituisce una geometria. Ad esempio, per la funzione GeoGetPolygonCenter(), se la geometria del limite di un'area viene impostata come input, per il centro di tale area viene restituita la geometria del punto (longitudine e latitudine).

Le funzioni di aggregazione sono le seguenti:

GeoAggrGeometry

La funzione **GeoAggrGeometry()** viene utilizzata per aggregare un numero di aree in un'area più estesa, aggregando ad esempio un numero di sottoregioni in una singola regione.

```
GeoAggrGeometry (field_name)
```

GeoBoundingBox

La funzione **GeoBoundingBox()** viene utilizzata per aggregare una geometria in un'area e calcolare la casella di delimitazione più piccola contenente tutte le coordinate.

```
GeoBoundingBox (field_name)
```

GeoCountVertex

La funzione **GeoCountVertex()** viene utilizzata per trovare il numero di vertici contenuti nella geometria di un poligono.

```
GeoCountVertex(field_name)
```

GeoInvProjectGeometry

La funzione **GeoInvProjectGeometry()** viene utilizzata per aggregare una geometria in un'area e applicare l'inverso di una proiezione.

```
GeoInvProjectGeometry(type, field_name)
```

GeoProjectGeometry

La funzione **GeoProjectGeometry()** viene utilizzata per aggregare una geometria in un'area e applicare una proiezione.

```
GeoProjectGeometry(type, field_name)
```

GeoReduceGeometry

La funzione **GeoReduceGeometry()** viene utilizzata per ridurre il numero di vertici di una geometria e per aggregare un certo numero di aree in una singola area, continuando a visualizzare le linee di confine delle singole aree.

```
GeoReduceGeometry (geometry)
```

Le funzioni di non aggregazione sono le seguenti:

GeoGetBoundingBox

La funzione **GeoGetBoundingBox()** viene utilizzata negli script e nelle espressioni grafiche per calcolare la casella di delimitazione geospaziale più piccola contenente tutte le coordinate di una geometria.

```
GeoGetBoundingBox (geometry)
```

GeoGetPolygonCenter

La funzione **GeoGetPolygonCenter()** viene utilizzata negli script e nelle espressioni grafiche per calcolare e restituire il punto centrale di una geometria.

```
GeoGetPolygonCenter (geometry)
```

GeoMakePoint

La funzione **GeoMakePoint()** viene utilizzata negli script e nelle espressioni grafiche per creare e assegnare un tag a un punto con latitudine e longitudine.

```
GeoMakePoint (lat_field_name, lon_field_name)
```

GeoProject

La funzione **GeoProject()** viene utilizzata negli script e nelle espressioni grafiche per applicare una proiezione a una geometria.

```
GeoProject (type, field_name)
```

GeoAggrGeometry

La funzione **GeoAggrGeometry()** viene utilizzata per aggregare un numero di aree in un'area più estesa, aggregando ad esempio un numero di sottoregioni in una singola regione.

Sintassi:

```
GeoAggrGeometry(field_name)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.

In genere, la funzione **GeoAggrGeometry()** può essere utilizzata per combinare i dati relativi ai limiti geospaziali. Ad esempio, si potrebbe disporre di aree di codice di avviamento postale per le parti periferiche di una città e dei ricavi delle vendite per ciascuna area. Se l'area del venditore copre più aree di codice di avviamento postale, potrebbe risultare utile presentare le vendite totali in base all'area di vendita, anziché singole aree, e visualizzare i risultati su una mappa codificata con i colori.

GeoAggrGeometry() può calcolare l'aggregazione delle singole geometrie della periferia e generare la geometria dei territori uniti in un modello dati. Quindi, se i limiti dell'area di vendita vengono modificati, quando i dati vengono ricaricati i nuovi limiti uniti e i ricavi verranno riprodotti nella mappa.

Poiché **GeoAggrGeometry()** è una funzione di aggregazione, se viene utilizzata nello script è necessario utilizzare un'istruzione **LOAD** con una clausola **Group by**.

*Le linee dei limiti delle mappe create mediante **GeoAggrGeometry()** sono quelle delle aree unite. Se si desidera visualizzare le linee dei singoli limiti delle aree preaggregate, utilizzare **GeoReduceGeometry()**.*

Esempi:
In questo esempio viene caricato un file KML con dati di area e quindi una tabella con i dati di area aggregati.

```
[MapSource]: LOAD [world.Name], [world.Point], [world.Area] FROM [lib://Downloads/world.kml]
(kml, Table is [World.shp/Features]); Map: LOAD world.Name, GeoAggrGeometry(world.Area) as
[AggrArea] resident MapSource Group By world.Name;

 Drop Table MapSource;
```

GeoBoundingBox

La funzione **GeoBoundingBox()** viene utilizzata per aggregare una geometria in un'area e calcolare la casella di delimitazione più piccola contenente tutte le coordinate.

Una funzione GeoBoundingBox è rappresentata come elenco di quattro valori: sinistro, destro, superiore, inferiore.

Sintassi:

```
GeoBoundingBox(field_name)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.

GeoBoundingBox() aggrega una serie di geometrie e restituisce quattro coordinate per il rettangolo più piccolo che contiene tutte le coordinate della geometria aggregata.

Per visualizzare il risultato su una mappa, è necessario trasferire la stringa risultante di quattro coordinate in un formato di poligono, contrassegnare il campo trasferito con un formato di geopoligono e trascinare tale campo nell'oggetto mappa. Le caselle rettangolari verranno quindi visualizzate nella visualizzazione della mappa.

GeoCountVertex

La funzione **GeoCountVertex()** viene utilizzata per trovare il numero di vertici contenuti nella geometria di un poligono.

Sintassi:

```
GeoCountVertex(field_name)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.

GeoGetBoundingBox

La funzione **GeoGetBoundingBox()** viene utilizzata negli script e nelle espressioni grafiche per calcolare la casella di delimitazione geospaziale più piccola contenente tutte le coordinate di una geometria.

Una casella di delimitazione geospaziale, creata dalla funzione GeoBoundingBox() è rappresentata come elenco di quattro valori: sinistro, destro, superiore, inferiore.

Sintassi:

```
GeoGetBoundingBox(field_name)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.

*Non utilizzare la clausola **Group by** nell'editor caricamento dati con questa funzione e altre funzioni geospaziali non aggreganti, in quanto verrebbe generato un errore di caricamento.*

GeoGetPolygonCenter

La funzione **GeoGetPolygonCenter()** viene utilizzata negli script e nelle espressioni grafiche per calcolare e restituire il punto centrale di una geometria.

In alcuni casi il requisito è disegnare un punto anziché un riempimento di colore su una mappa. Se il dato geospaziale esistente è unicamente disponibile sotto forma di geometria dell'area (ad esempio un limite), utilizzare **GeoGetPolygonCenter()** per recuperare una coppia di longitudine e latitudine per il centro dell'area.

Sintassi:

```
GeoGetPolygonCenter(field_name)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.

*Non utilizzare la clausola **Group by** nell'editor caricamento dati con questa funzione e altre funzioni geospaziali non aggreganti, in quanto verrebbe generato un errore di caricamento.*

GeoInvProjectGeometry

La funzione **GeoInvProjectGeometry()** viene utilizzata per aggregare una geometria in un'area e applicare l'inverso di una proiezione.

Sintassi:

```
GeoInvProjectGeometry(type, field_name)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
type	Il tipo di proiezione utilizzato nella trasformazione della geometria della mappa. Questo può assumere uno di due valori: "unit", (valore predefinito), che dà come risultato una proiezione 1:1, o "mercator", che utilizza la proiezione standard Mercator.
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.

Esempio:

Esempio di script

Esempio	Risultato
In un'istruzione `Load`: `GeoInvProjectGeometry ('mercator',AreaPolygon) as InvProjectGeometry`	La geometria caricata come **AreaPolygon** viene utilizzata mediante la trasformazione inversa della proiezione Mercator, quindi viene memorizzata come **InvProjectGeometry** per essere utilizzata nelle visualizzazioni.

GeoMakePoint

La funzione **GeoMakePoint()** viene utilizzata negli script e nelle espressioni grafiche per creare e assegnare un tag a un punto con latitudine e longitudine. GeoMakePoint restituisce punti nell'ordine longitudine e latitudine.

Sintassi:

```
GeoMakePoint(lat_field_name, lon_field_name)
```

Tipo di dati restituiti: stringa, formattata [longitudine, latitudine]

Argomenti:

Argomenti

Argomento	Descrizione
lat_field_name	Un campo o un'espressione che fa riferimento a un campo che rappresenta la latitudine del punto.
lon_field_name	Un campo o un'espressione che fa riferimento a un campo che rappresenta la longitudine del punto.

*Non utilizzare la clausola **Group by** nell'editor caricamento dati con questa funzione e altre funzioni geospaziali non aggreganti, in quanto verrebbe generato un errore di caricamento.*

GeoProject

La funzione **GeoProject()** viene utilizzata negli script e nelle espressioni grafiche per applicare una proiezione a una geometria.

Sintassi:

```
GeoProject(type, field_name)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
type	Il tipo di proiezione utilizzato nella trasformazione della geometria della mappa. Questo può assumere uno di due valori: "unit", (valore predefinito), che dà come risultato una proiezione 1:1, o "mercator", che utilizza la proiezione standard Mercator Web.
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.

*Non utilizzare la clausola **Group by** nell'editor caricamento dati con questa funzione e altre funzioni geospaziali non aggreganti, in quanto verrebbe generato un errore di caricamento.*

Esempio:

Esempi di script

Esempio	Risultato
In un'istruzione `Load`: `GeoProject ('mercator',Area) as GetProject`	La proiezione Mercator viene applicata alla geometria caricata come **Area** e il risultato viene memorizzato come **GetProject**.

GeoProjectGeometry

La funzione **GeoProjectGeometry()** viene utilizzata per aggregare una geometria in un'area e applicare una proiezione.

Sintassi:

```
GeoProjectGeometry(type, field_name)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
type	Il tipo di proiezione utilizzato nella trasformazione della geometria della mappa. Questo può assumere uno di due valori: "unit", (valore predefinito), che dà come risultato una proiezione 1:1, o "mercator", che utilizza la proiezione standard Mercator Web.
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.

Esempio:

Esempio	Risultato
In un'istruzione `Load`: `GeoProjectGeometry ('mercator',AreaPolygon) as ProjectGeometry`	La geometria caricata come **AreaPolygon** viene trasformata mediante la proiezione Mercator, quindi viene memorizzata come **ProjectGeometry** per essere utilizzata nelle visualizzazioni.

GeoReduceGeometry

La funzione **GeoReduceGeometry()** viene utilizzata per ridurre il numero di vertici di una geometria e per aggregare un certo numero di aree in una singola area, continuando a visualizzare le linee di confine delle singole aree.

Sintassi:

```
GeoReduceGeometry(field_name[, value])
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Un campo o un'espressione che fa riferimento a un campo contenente la geometria da rappresentare. Potrebbe trattarsi di un punto (o di una serie di punti) che fornisce la longitudine e la latitudine o un'area.
value	La quantità di riduzione da applicare alla geometria. La scala va da 0 a 1, in cui 0 rappresenta nessuna riduzione e 1 rappresenta la riduzione massima dei vertici.  *Utilizzando un value di 0,9 o superiore con una serie di dati complessa è possibile ridurre il numero di vertici a un livello in cui la rappresentazione visiva non è accurata.*

GeoReduceGeometry() esegue inoltre una funzione simile a **GeoAggrGeometry()** in quanto aggrega un certo numero di aree in un'area. La differenza è che le singole linee dei limiti dei dati di preaggregazione vengono visualizzate sulla mappa se si utilizza **GeoReduceGeometry()**.

Poiché **GeoReduceGeometry()** è una funzione di aggregazione, se viene utilizzata nello script è necessario utilizzare un'istruzione **LOAD** con una clausola **Group by**.

Esempi:
In questo esempio viene caricato un file KML con dati di area e quindi una tabella con i dati di area ridotti e aggregati.

```
[MapSource]: LOAD [world.Name], [world.Point], [world.Area] FROM [lib://Downloads/world.kml]
(kml, Table is [World.shp/Features]); Map: LOAD world.Name, GeoReduceGeometry(world.Area,0.5)
as [ReducedArea] resident MapSource Group By world.Name;

 Drop Table MapSource;
```

5.15 Funzioni di interpretazione

Le funzioni di interpretazione valutano i contenuti dei campi o delle espressioni di testo di input e impongono il formato dati specificato al valore numerico risultante. Queste funzioni consentono di specificare il formato del numero, in conformità con il relativo tipo di dati, includendo attributi come i separatori decimali, i separatori delle migliaia e il formato dati.

Tutte le funzioni di interpretazione restituiscono un valore duale che riporta sia la stringa che il valore numerico, ma possono essere interpretate come una conversione da stringa a numero. Queste funzioni generano un numero che rappresenta la stringa a partire dal valore di testo dell'espressione di input.

Le funzioni di formattazione invece si comportano nel modo opposto: le espressioni numeriche vengono valutate come stringhe specificando il formato di visualizzazione del testo risultante.

Se non vengono utilizzate le funzioni di interpretazione, Qlik Sense interpreta i dati come un insieme di numeri, date, ore, indicatori temporali e stringhe, utilizzando le impostazioni predefinite per il formato numerico, il formato della data e il formato dell'ora definite dalle variabili dello script e dal sistema operativo.

Tutte le funzioni di interpretazione possono essere utilizzate sia negli script di caricamento dei dati che nelle espressioni grafiche.

Tutte le rappresentazioni numeriche vengono fornite con un punto decimale come separatore decimale.

Prospetto delle funzioni di interpretazione

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Date#

Date# valuta un'espressione come data nel formato specificato nel secondo argomento, se fornito. Se si omette il codice del formato, verrà utilizzato il formato di data predefinito impostato nel sistema operativo.

```
Date# (page 1257)(text[, format])
```

Interval#

Interval#() valuta un'espressione di testo come intervallo di tempo nel formato impostato nel sistema operativo per impostazione predefinita, oppure nel formato specificato nel secondo argomento, se disponibile.

```
Interval# (page 1258)(text[, format])
```

Money#

Money#() converte una stringa di testo in un valore di valuta nel formato impostato nello script di caricamento o nel sistema operativo, a meno che non venga fornita una stringa di formattazione. I simboli dei separatori decimali e delle migliaia personalizzati sono parametri opzionali.

```
Money# (page 1259)(text[, format[, dec_sep[, thou_sep ] ] ])
```

Num#

Num#() interpreta una stringa di testo come un valore numerico, ovvero converte la stringa di input in un numero usando il formato specificato nel secondo parametro. Se il secondo parametro viene omesso, utilizza i separatori decimali e delle migliaia impostati nello script di caricamento dei dati. I simboli dei separatori decimali e delle migliaia personalizzati sono parametri opzionali.

```
Num# (page 1260)(text[ , format[, dec_sep[ , thou_sep]]])
```

Text
Text() obbliga a considerare l'espressione come testo, anche nel caso in cui sia possibile un'interpretazione numerica.

```
Text(expr)
```

Time#
Time#() valuta un'espressione come valore ora nel formato dell'ora impostato nello script di caricamento dei dati o nel sistema operativo, a meno che non venga fornita una stringa di formattazione..

```
Time# (page 1262)(text[, format])
```

Timestamp#
Timestamp#() valuta un'espressione come valore data e ora nel formato dell'indicatore temporale impostato nello script di caricamento dei dati o nel sistema operativo, a meno che non venga fornita una stringa di formattazione.

```
Timestamp# (page 1263)(text[, format])
```

Vedere anche:
p *Funzioni di formattazione (page 1221)*

Date#

Date# valuta un'espressione come data nel formato specificato nel secondo argomento, se fornito.

Sintassi:

```
Date#(text[, format])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa di testo da valutare.
format	Stringa che descrive il formato della stringa di testo da valutare. Se omessa, verrà utilizzato il formato della data impostato nelle variabili di sistema nello script di caricamento dei dati o nel sistema operativo.

Esempi e risultati:
Nel seguente esempio viene utilizzato il formato della data **M/D/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati.

Aggiungere questo script di esempio all'app ed eseguirlo.

```
Load *,
Num(Date#(StringDate)) as Date;
LOAD * INLINE [
StringDate
8/7/97
8/6/1997
]
```

Se si crea una tabella utilizzando **StringDate** e **Date** come dimensioni, si otterranno i seguenti risultati:

Risultati

StringDate	Date
8/7/97	35649
8/6/1997	35648

Interval#

Interval#() valuta un'espressione di testo come intervallo di tempo nel formato impostato nel sistema operativo per impostazione predefinita, oppure nel formato specificato nel secondo argomento, se disponibile.

Sintassi:

```
Interval#(text[, format])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa di testo da valutare.
format	Stringa che descrive il formato di input previsto da utilizzare per la conversione della stringa in un intervallo numerico. Se viene omessa, si utilizzerà il formato della data breve, il formato dell'ora e il separatore decimale impostati nel sistema operativo.

La funzione **interval#** converte un intervallo di tempo in formato testuale in un equivalente numerico.

Esempi e risultati:
Gli esempi riportati di seguito presuppongono l'utilizzo delle seguenti impostazioni del sistema operativo:

- Formato data breve: YY-MM-DD
- Formato ora: M/D/YY
- Separatore decimale dei numeri: .

Risultati

Esempio	Risultato
`Interval#( A, 'D hh:mm' )` dove A='1 09:00'	1.375

Money#

Money#() converte una stringa di testo in un valore di valuta nel formato impostato nello script di caricamento o nel sistema operativo, a meno che non venga fornita una stringa di formattazione. I simboli dei separatori decimali e delle migliaia personalizzati sono parametri opzionali.

Sintassi:

```
Money#(text[, format[, dec_sep [, thou_sep ] ] ])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa di testo da valutare.
format	Stringa che descrive il formato di input previsto da utilizzare per la conversione della stringa in un intervallo numerico. Se viene omessa, viene utilizzato il formato della valuta impostato nel sistema operativo.
dec_sep	Stringa che specifica il separatore decimale dei numeri. Se omessa, verrà utilizzato il valore MoneyDecimalSep impostato nello script di caricamento dei dati.
thou_sep	Stringa che specifica il separatore delle migliaia dei numeri. Se omessa, verrà utilizzato il valore MoneyThousandSep impostato nello script di caricamento dei dati.

In genere, la funzione **money#** si comporta in modo analogo alla funzione **num#**, ma acquisisce i propri valori predefiniti per il separatore decimale e delle migliaia dalle variabili dello script per il formato valuta oppure dalle impostazioni di valuta del sistema.

Esempi e risultati:
Gli esempi riportati di seguito presuppongono l'utilizzo delle due seguenti impostazioni del sistema operativo:

- Impostazione predefinita del formato della valuta 1: kr # ##0,00
- Impostazione predefinita del formato della valuta 2: $ #,##0.00

```
Money#(A , '# ##0,00 kr' )
```
dove A=35 648,37 kr

Risultati

Risultati	Impostazione 1	Impostazione 2
Stringa	35 648.37 kr	35 648.37 kr
Numero	35648.37	3564837

```
Money#( A, ' $#', '.', ',' )
```
dove A= $35.648,37

Risultati

Risultati	Impostazione 1	Impostazione 2
Stringa	$35,648.37	$35,648.37
Numero	35648.37	35648.37

Num#

Num#() interpreta una stringa di testo come un valore numerico, ovvero converte la stringa di input in un numero usando il formato specificato nel secondo parametro. Se il secondo parametro viene omesso, utilizza i separatori decimali e delle migliaia impostati nello script di caricamento dei dati. I simboli dei separatori decimali e delle migliaia personalizzati sono parametri opzionali.

Sintassi:

```
Num#(text[, format[, dec_sep [, thou_sep ] ] ])
```

Tipo di dati restituiti: duale

La funzione **Num#()** restituisce un valore duale contenente sia la stringa che il valore numerico. La funzione prende la rappresentazione testuale dell'espressione di input e genera una stringa che rappresenta il numero. Non modifica il formato del numero: l'output viene formattato nello stesso modo dell'input.

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa di testo da valutare.
format	Stringa che specifica il formato numerico utilizzato nel primo parametro. Se omessi, verranno utilizzati i separatori di decimali e migliaia impostati nello script di caricamento dei dati.
dec_sep	Stringa che specifica il separatore decimale dei numeri. Se omesso, viene utilizzato il valore della variabile DecimalSep impostato nello script di caricamento dei dati.

Argomento	Descrizione
thou_sep	Stringa che specifica il separatore delle migliaia dei numeri. Se omesso, viene utilizzato il valore della variabile ThousandSep impostato nello script di caricamento dei dati.

Esempi e risultati:
La tabella seguente mostra il risultato di *Num#(A, '#', '.' , ',')* per valori diversi di A.

		Risultati
Una	**Rappresentazione stringa**	**Valore numerico (qui visualizzato con punto decimale)**
35,648.31	35,648.31	35648.31
35 648.312	35 648.312	35648.312
35.648,3123	35.648,3123	-
35 648,31234	35 648,31234	-

Text

Text() obbliga a considerare l'espressione come testo, anche nel caso in cui sia possibile un'interpretazione numerica.

Sintassi:

```
Text (expr)
```

Tipo di dati restituiti: duale

Esempio:

```
Text( A )
```
dove A=1234

	Risultati
Stringa	**Numero**
1234	-

Esempio:

```
Text( pi( ) )
```

	Risultati
Stringa	**Numero**
3.1415926535898	-

Time#

Time#() valuta un'espressione come valore ora nel formato dell'ora impostato nello script di caricamento dei dati o nel sistema operativo, a meno che non venga fornita una stringa di formattazione..

Sintassi:

```
time#(text[, format])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa di testo da valutare.
format	Stringa che descrive il formato della stringa di testo da valutare. Se viene omessa, si utilizzerà il formato della data breve, il formato dell'ora e il separatore decimale impostati nel sistema operativo.

Esempio:

- Impostazione predefinita del formato dell'ora 1: hh:mm:ss
- Impostazione predefinita del formato dell'ora 2: hh.mm.ss

```
time#( A )
```
dove A=09:00:00

Risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	09:00:00	09:00:00
Numero:	0.375	-

Esempio:

- Impostazione predefinita del formato dell'ora 1: hh:mm:ss
- Impostazione predefinita del formato dell'ora 2: hh.mm.ss

```
time#( A, 'hh.mm' )
```
dove A=09.00

Risultati

Risultati	Impostazione 1	Impostazione 2
Stringa:	09.00	09.00
Numero:	0.375	0.375

Timestamp#

Timestamp#() valuta un'espressione come valore data e ora nel formato dell'indicatore temporale impostato nello script di caricamento dei dati o nel sistema operativo, a meno che non venga fornita una stringa di formattazione.

Sintassi:

```
timestamp#(text[, format])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa di testo da valutare.
format	Stringa che descrive il formato della stringa di testo da valutare. Se viene omessa, si utilizzerà il formato della data breve, il formato dell'ora e il separatore decimale impostati nel sistema operativo. ISO 8601 è supportato per gli indicatori temporali.

Esempio:

Nel seguente esempio viene utilizzato il formato della data **M/D/YYYY**. Il formato della data viene specificato nell'istruzione **SET DateFormat** nella parte superiore dello script di caricamento dei dati.

Aggiungere questo script di esempio all'app ed eseguirlo.

```
Load *,
Timestamp(Timestamp#(String)) as TS;
LOAD * INLINE [
Stringa
2015-09-15T12:13:14
1952-10-16T13:14:00+0200
1109-03-01T14:15
];
```

Se si crea una tabella utilizzando **String** e **TS** come dimensioni, si otterranno i seguenti risultati:

Risultati

Stringa	TS
2015-09-15T12:13:14	9/15/2015 12:13:14 PM
1952-10-16T13:14:00+0200	10/16/1952 11:14:00 AM
1109-03-01T14:15	3/1/1109 2:15:00 PM

5.16 Funzioni intra-record

Le funzioni intra-record sono utilizzate:

- Nello script di caricamento dei dati quando è necessario il valore di un record di dati precedentemente caricato per valutare il record attuale.
- Nelle espressioni grafiche quando è necessario un altro valore della serie di dati di una visualizzazione.

L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza una funzione di grafico intra-record in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza una funzione di grafico intra-record in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito nella funzione intra-record. Questa limitazione non si applica alla funzione di script equivalente, se esiste.

Le definizioni di espressioni autoreferenziali possono essere create in modo affidabile solo in tabelle con meno di 100 righe, anche se ciò può variare in base all'hardware su cui Qlik Engine è in esecuzione.

Funzioni di riga

Queste funzioni possono essere utilizzate solo nelle espressioni grafiche.

Above

Above() valuta un'espressione in una riga sopra la riga attuale all'interno di un segmento di colonna in una tabella. La riga per la quale viene calcolata dipende dal valore dell'**offset**, se presente, e l'impostazione predefinita è la riga direttamente sopra. Per i grafici diversi dalle tabelle, **Above()** restituisce un valore per la riga sopra la riga attuale nell'equivalente di tabella lineare del grafico.

```
Above - funzione per grafici([TOTAL [<fld{,fld}>]] expr [ , offset [,count]])
```

Below

Below() valuta un'espressione in una riga sotto la riga attuale all'interno di un segmento di colonna in una tabella. La riga per la quale viene calcolata dipende dall'**offset**, se presente, e l'impostazione predefinita è la riga direttamente sotto. Per i grafici diversi dalle tabelle, **Below()** restituisce un valore per la riga sotto la

colonna attuale nell'equivalente di tabella lineare del grafico.

```
Below - funzione per grafici([TOTAL[<fld{,fld}>]] expression [ , offset
[,count ]])
```

Bottom

Bottom() valuta un'espressione nell'ultima riga (inferiore) di un segmento di colonna in una tabella. La riga per la quale viene calcolata dipende dal valore dell'**offset**, se presente, e l'impostazione predefinita è la riga direttamente nella parte inferiore. Per i grafici diversi dalle tabelle, la valutazione viene effettuata sull'ultima riga della colonna attuale nell'equivalente di tabella lineare del grafico.

```
Bottom - funzione per grafici([TOTAL[<fld{,fld}>]] expr [ , offset [,count
 ]])
```

Top

Top() valuta un'espressione nella prima riga (superiore) di un segmento colonna in una tabella. La riga per la quale viene calcolata dipende dal valore dell'**offset**, se presente, e l'impostazione predefinita è la riga superiore. Per i grafici diversi dalle tabelle, la valutazione di **Top()** viene effettuata sulla prima riga della colonna attuale nell'equivalente di tabella lineare del grafico.

```
Top - funzione per grafici([TOTAL [<fld{,fld}>]] expr [ , offset [,count ]])
```

NoOfRows

NoOfRows() restituisce il numero di righe nel segmento di colonna attuale in una tabella. Per i grafici bitmap, **NoOfRows()** restituisce il numero di righe nell'equivalente della tabella lineare del grafico.

```
NoOfRows - funzione per grafici([TOTAL])
```

Funzioni di colonna

Queste funzioni possono essere utilizzate solo nelle espressioni grafiche.

Column

Column() restituisce il valore trovato nella colonna corrispondente a **ColumnNo** in una tabella lineare, ignorando le dimensioni. Ad esempio **Column(2)** restituisce il valore della seconda colonna della misura.

```
Column - funzione per grafici(ColumnNo)
```

Dimensionality

Dimensionality() restituisce il numero di dimensioni per la riga attuale. Nel caso delle tabelle pivot la funzione restituisce il numero totale di colonne di dimensione senza contenuti di aggregazione, ossia senza somme parziali o aggregati compressi.

```
Dimensionality - funzione per grafici ( )
```

Secondarydimensionality

SecondaryDimensionality() restituisce il numero di righe di dimensione di una tabella pivot senza contenuti di aggregazione; ad esempio, senza somme parziali o aggregati compressi. Questa funzione è equivalente alla funzione **dimensionality()** per le dimensioni orizzontali delle tabelle pivot.

```
SecondaryDimensionality - funzione per grafici ( )
```

Funzioni di campo

FieldIndex

FieldIndex() restituisce la posizione del valore di campo **value** presente nel campo **field_name** (in ordine di caricamento).

```
FieldIndex (field_name , value)
```

FieldValue

FieldValue() restituisce il valore trovato nella posizione **elem_no** del campo **field_name** (in ordine di caricamento).

```
FieldValue (field_name , elem_no)
```

FieldValueCount

FieldValueCount() è una funzione **integer** che restituisce il numero di valori distinti di un campo.

```
FieldValueCount (field_name)
```

Funzioni tabella pivot

Queste funzioni possono essere utilizzate solo nelle espressioni grafiche.

After

After() restituisce il valore di un'espressione valutata con i valori di dimensione di una tabella pivot così come appaiono nella colonna successiva a quella attuale all'interno di un segmento di riga nella tabella pivot.

```
After - funzione per grafici([TOTAL] expression [ , offset [,n]])
```

Before

Before() restituisce il valore di un'espressione valutata con i valori di dimensione di una tabella pivot così come appaiono nella colonna precedente a quella attuale all'interno di un segmento di riga della tabella pivot.

```
Before - funzione per grafici([TOTAL] expression [ , offset [,n]])
```

First

First() restituisce il valore di un'espressione valutata con i valori di dimensione di una tabella pivot così come appaiono nella prima colonna del segmento di riga attuale della tabella pivot. Questa funzione restituisce NULL in tutti i tipi di grafico, ad eccezione delle tabelle pivot.

```
First - funzione per grafici([TOTAL] expression [ , offset [,n]])
```

Last

Last() restituisce il valore di un'espressione valutata con i valori di dimensione di una tabella pivot così come appaiono nell'ultima colonna del segmento di riga attuale della tabella pivot. Questa funzione restituisce NULL in tutti i tipi di grafico, ad eccezione delle tabelle pivot.

```
Last - funzione per grafici([TOTAL] expression [ , offset [,n]])
```

ColumnNo

ColumnNo() restituisce il numero della colonna attuale all'interno del segmento di riga attuale in una tabella pivot. La prima colonna è la numero 1.

```
ColumnNo - funzione per grafici([TOTAL])
```

NoOfColumns

NoOfColumns() restituisce il numero di colonne nel segmento di riga attuale in una tabella pivot.

```
NoOfColumns - funzione per grafici([TOTAL])
```

Funzioni intra-record nello script di caricamento dei dati

Exists

Exists() determina se un valore di campo specifico è già stato caricato nel campo nello script di caricamento dei dati. La funzione restituisce TRUE o FALSE, quindi può essere utilizzata nella clausola **where** di un'istruzione **LOAD** o un'istruzione **IF**.

```
Exists (field_name [, expr])
```

LookUp

Lookup() osserva una tabella già caricata e restituisce il valore di **field_name** corrispondente alla prima occorrenza del valore **match_field_value** nel campo **match_field_name**. La tabella può essere la tabella attuale o un'altra tabella caricata in precedenza.

```
LookUp (field_name, match_field_name, match_field_value [, table_name])
```

Peek

Peek() restituisce il valore di un campo in una tabella per una riga che è già stata caricata. È possibile specificare il numero di riga così come la tabella. Se non viene specificato alcun numero di riga, verrà utilizzato l'ultimo record precedentemente caricato.

```
Peek (field_name[, row_no[, table_name ] ])
```

Previous

Previous() restituisce il valore dell'espressione **expr** utilizzando i dati del record di input precedente che non è stato eliminato a causa di una clausola **where**. Nel primo record di una tabella interna, la funzione restituirà NULL.

```
Previous (page 1300)(expr)
```

Vedere anche:

p *Funzioni di scala (page 1321)*

Above - funzione per grafici

Above() valuta un'espressione in una riga sopra la riga attuale all'interno di un segmento di colonna in una tabella. La riga per la quale viene calcolata dipende dal valore dell'**offset**, se presente, e l'impostazione predefinita è la riga direttamente sopra. Per i grafici diversi dalle tabelle, **Above()** restituisce un valore per la riga sopra la riga attuale nell'equivalente di tabella lineare del grafico.

Sintassi:

```
Above([TOTAL] expr [ , offset [,count]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
offset	Specificando un offsetn maggiore di 0, è possibile spostare la valutazione dell'espressione n righe più in alto della riga attuale. Specificando un offset uguale a 0 verrà valutata l'espressione nella riga attuale. Specificando un numero di offset negativo, la funzione Above diventa equivalente alla funzione Below con il numero di offset positivo corrispondente.
count	Specificando un terzo argomento **count** maggiore di 1, la funzione restituirà una scala di valori **count**, uno per ciascuna delle righe della tabella **count** contando verso l'alto a partire dalla cella originaria. In questo modulo la funzione può essere utilizzata come argomento per una qualsiasi delle funzioni di scala speciali. *Funzioni di scala (page 1321)*
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

In corrispondenza della prima riga di un segmento di colonna verrà restituito un valore NULL perché non vi sono righe che la precedono.

Un segmento di colonna viene definito come un sottogruppo consecutivo di celle con gli stessi valori per le dimensioni nell'ordine attuale. Le funzioni grafiche intra-record vengono calcolate nel segmento colonna escludendo la dimensione più a destra nel grafico della tabella lineare equivalente. Se nel grafico è presente una sola dimensione oppure è specificato il qualificatore TOTAL, l'espressione viene valutata nell'intera tabella.

Se la tabella o l'equivalente di tabella include più dimensioni verticali, il segmento colonna attuale includerà solo righe contenenti gli stessi valori della riga attuale in tutte le colonne di dimensione, eccetto la colonna che mostra l'ultima dimensione nell'ordinamento tra campi.

Limiti:

- Le chiamate ricorrenti restituiranno NULL.
- L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Esempi e risultati:

Example 1:

Visualizzazione della tabella per l'esempio 1

Customer	Sum([Sales])	Above(Sum(Sales))	Sum(Sales)+Above(Sum(Sales))	Above offset 3	Higher?
	2566	-	-	-	-
Astrida	587	-	-	-	-
Betacab	539	587	1126	-	-
Canutility	683	539	1222	-	Higher
Divadip	757	683	1440	1344	Higher

Nella schermata della tabella mostrata in questo esempio, la visualizzazione della tabella viene creata dalla dimensione **Customer** e dalle misure: `Sum(Sales)` e `Above(Sum(Sales))`.

La colonna Above(Sum(Sales)) restituisce NULL per la riga **Customer** contenente **Astrida** perché non ci sono righe che la precedono. Il risultato per la riga **Betacab** mostra il valore di Sum(Sales) per **Astrida**, il risultato per **Canutility** mostra il valore per **Sum(Sales)** per **Betacab**, e così via.

Per la colonna con etichetta Sum(Sales)+Above(Sum(Sales)), la riga per **Betacab** mostra il risultato dell'aggiunta dei valori **Sum(Sales)** per le righe **Betacab** + **Astrida** (539+587). Il risultato per la riga **Canutility** mostra il risultato dell'aggiunta dei valori **Sum(Sales)** per **Canutility** + **Betacab** (683+539).

La misura con etichetta Above offset 3 creata utilizzando l'espressione `Sum(Sales)+Above(Sum(Sales), 3)` ha l'argomento **offset**, impostato su 3, e ha l'effetto di portare il valore della riga tre righe sopra il valore attuale. Aggiunge il valore **Sum(Sales)** per il valore **Customer** attuale al valore per le tre righe **Customer**precedenti. I valori restituiti per le prime tre righe **Customer** sono null.

Nella tabella sono mostrate anche misure più complesse: una creata da `Sum(Sales)+Above(Sum(Sales))` e una con etichetta **Higher?**, che viene creata da `IF(Sum(Sales)>Above(Sum(Sales)), 'Higher')`.

Questa funzione può essere utilizzata nei grafici diversi dalle tabelle, ad esempio nei grafici a barre.

Per gli altri tipi di grafici, convertire il grafico nell'equivalente di tabella lineare per individuare facilmente la riga a cui si riferisce la funzione.

Example 2:

Nelle schermate delle tabelle mostrate in questo esempio sono state aggiunte altre dimensioni alle visualizzazioni: **Month** e **Product**. Per i grafici con più dimensioni, i risultati delle espressioni contenenti le funzioni **Above**, **Below**, **Top** e **Bottom** dipendono dalla modalità con cui le dimensioni di colonna vengono ordinate da Qlik Sense. Qlik Sense valuta le funzioni in base ai segmenti di colonna che derivano dalla dimensione ordinata per ultima. L'ordinamento delle colonne viene controllato dal pannello delle proprietà in **Ordinamento** e non è necessariamente l'ordine in cui le colonne vengono visualizzate in una tabella.

Nella seguente schermata della visualizzazione della tabella relativa all'esempio 2, l'ultima dimensione ordinata è **Month**, in modo che la funzione **Above** venga valutata in base ai mesi. Esiste una serie di risultati per ciascun valore **Product** per ciascun mese (da **Jan** a **Aug**): un segmento colonna. Viene seguita da una serie per il segmento di colonna successivo: per ciascun mese **Month** per il valore **Product** successivo. Esisterà un segmento di colonna per ciascun valore **Customer** per ciascun valore **Product**.

Visualizzazione della tabella per l'esempio 2

Customer	Product	Month	Sum([Sales])	Above(Sum(Sales))
			2566	-
Astrida	AA	Jan	46	-
Astrida	AA	Feb	60	46
Astrida	AA	Mar	70	60
Astrida	AA	Apr	13	70
Astrida	AA	May	78	13
Astrida	AA	Jun	20	78
Astrida	AA	Jul	45	20
Astrida	AA	Aug	65	45

Example 3:

Nella schermata della visualizzazione della tabella per l'esempio 3, l'ultima dimensione ordinata è **Product**. Questo risultato viene ottenuto spostando la dimensione Product nella posizione 3 nella scheda Ordinamento nel pannello delle proprietà. La funzione **Above** viene valutata per ciascun valore **Product** e, poiché esistono solo due prodotti, **AA** e **BB**, esiste solo un risultato non null in ciascuna serie. Nella riga **BB** per il mese **Jan**, il valore per **Above(Sum(Sales))** è 46. Per la riga **AA**, il valore è null. Il valore in ciascuna riga **AA** per ciascun mese sarà sempre null perché non esiste alcun valore **Product** sopra AA. La seconda serie viene valutata in **AA** e **BB** per il mese **Feb**, per il valore **Customer**, **Astrida**. Quando tutti i mesi sono stati valutati per **Astrida**, la sequenza viene ripetuta per il secondo **Customer**Betacab e così via.

Visualizzazione della tabella per l'esempio 3

Customer	Product	Month	Sum([Sales])	Above(Sum(Sales))
			2566	-
Astrida	AA	Jan	46	-
Astrida	BB	Jan	46	46
Astrida	AA	Feb	60	-
Astrida	BB	Feb	60	60
Astrida	AA	Mar	70	-
Astrida	BB	Mar	70	70
Astrida	AA	Apr	13	-
Astrida	BB	Apr	13	13

Esempio 4

Example 4:	**Risultato**
La funzione Above può essere utilizzata come input per le funzioni di scala. Ad esempio: `RangeAvg (Above(Sum (Sales),1,3))`.	Negli argomenti per la funzione Above(), offset viene impostato su 1 e count viene impostato su 3. La funzione trova i risultati dell'espressione Sum(Sales) nelle tre righe immediatamente sopra la riga attuale nel segmento di colonna (dove si trova una riga). Questi tre valori vengono utilizzati come input per la funzione RangeAvg(), che individua la media dei valori nella scala di numeri fornita. Una tabella in cui è stato impostato Customer come dimensione restituisce i seguenti risultati per l'espressione RangeAvg(). Astrida - Betacab 587 Canutility 563 Divadip: 603

Dati utilizzati negli esempi:

```
Monthnames:
LOAD *, Dual(MonthText,MonthNumber) as Month INLINE [
MonthText, MonthNumber
Jan, 1
Feb, 2
Mar, 3
Apr, 4
May, 5
Jun, 6
Jul, 7
Aug, 8
```

```
Sep, 9
Oct, 10
Nov, 11
Dec, 12
];

Sales2013:
Crosstable (MonthText, Sales) LOAD * inline [
Customer|Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec
Astrida|46|60|70|13|78|20|45|65|78|12|78|22
Betacab|65|56|22|79|12|56|45|24|32|78|55|15
Canutility|77|68|34|91|24|68|57|36|44|90|67|27
Divadip|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');
```

Vedere anche:

p *Below - funzione per grafici (page 1272)*
p *Bottom - funzione per grafici (page 1276)*
p *Top - funzione per grafici (page 1302)*
p *RangeAvg (page 1324)*

Below - funzione per grafici

Below() valuta un'espressione in una riga sotto la riga attuale all'interno di un segmento di colonna in una tabella. La riga per la quale viene calcolata dipende dall'**offset**, se presente, e l'impostazione predefinita è la riga direttamente sotto. Per i grafici diversi dalle tabelle, **Below()** restituisce un valore per la riga sotto la colonna attuale nell'equivalente di tabella lineare del grafico.

Sintassi:

```
Below([TOTAL] expr [ , offset [,count ]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
offset	Specificando un **offset**n maggiore di 1, la valutazione dell'espressione viene spostata di n righe più in basso rispetto alla riga attuale. Specificando un offset uguale a 0 verrà valutata l'espressione nella riga attuale. Specificando un numero di offset negativo, la funzione **Below** diventa equivalente alla funzione **Above** con il numero di offset positivo corrispondente.

Argomento	Descrizione
count	Specificando un terzo parametro **count** maggiore di 1, la funzione restituirà una scala di valori **count**, uno per ciascuna delle righe della tabella **count** contando verso il basso a partire dalla cella originaria. In questo modulo la funzione può essere utilizzata come argomento per una qualsiasi delle funzioni di scala speciali. *Funzioni di scala (page 1321)*
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Sull'ultima riga di un segmento di colonna viene restituito un valore NULL, perché non vi sono righe che la seguono.

Un segmento di colonna viene definito come un sottogruppo consecutivo di celle con gli stessi valori per le dimensioni nell'ordine attuale. Le funzioni grafiche intra-record vengono calcolate nel segmento colonna escludendo la dimensione più a destra nel grafico della tabella lineare equivalente. Se nel grafico è presente una sola dimensione oppure è specificato il qualificatore TOTAL, l'espressione viene valutata nell'intera tabella.

Se la tabella o l'equivalente di tabella include più dimensioni verticali, il segmento colonna attuale includerà solo righe contenenti gli stessi valori della riga attuale in tutte le colonne di dimensione, eccetto la colonna che mostra l'ultima dimensione nell'ordinamento tra campi.

Limiti:

- Le chiamate ricorrenti restituiranno NULL.
- L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Esempi e risultati:

Example 1:

Visualizzazione della tabella per l'esempio 1

Customer	Sum([Sales])	Below(Sum(Sales))	Sum(Sales)+Below(Sum(Sales))	Below + Offset 3	Higher
	2566	-	-	-	-
Astrida	587	539	1126	1344	Higher
Betacab	539	683	1222	-	-
Canutility	683	757	1440	-	-
Divadip	757	-	-	-	-

Nella tabella mostrata in questa schermata per l'esempio 1 la visualizzazione della tabella viene creata dalla dimensione **Customer** e dalle misure: `Sum(Sales)` e `Below(Sum(Sales))`.

La colonna **Below(Sum(Sales))** restituisce NULL per la riga **Customer** contenente **Divadip** perché non vi sono righe che la seguono. Il risultato per la riga **Canutility** mostra il valore di Sum(Sales) per **Divadip**, il risultato per **Betacab** mostra il valore per **Sum(Sales)** per **Canutility**, e così via.

La tabella mostra inoltre misure complesse che è possibile vedere nelle colonne con etichetta: `Sum (Sales)+Below(Sum(Sales))`, **Below +Offset 3** e **Higher?**. Queste espressioni funzionano come descritto nei seguenti paragrafi.

Per la colonna con etichetta **Sum(Sales)+Below(Sum(Sales))**, la riga per **Astrida** mostra il risultato dell'aggiunta dei valori **Sum(Sales)** per le righe **Betacab** + **Astrida** (539+587). Il risultato per la riga **Betacab** mostra il risultato dell'aggiunta dei valori **Sum(Sales)** per **Canutility** + **Betacab** (539+683).

La misura con etichetta **Below +Offset 3** creata utilizzando l'espressione `Sum(Sales)+Below(Sum(Sales), 3)` ha l'argomento **offset**, impostato su 3, e ha l'effetto di portare il valore della riga tre righe sotto il valore attuale. Aggiunge il valore **Sum(Sales)** per il valore **Customer** attuale dal valore **Customer** tre righe successive. I valori per le prime tre righe **Customer** più in basso sono null.

La misura con etichetta **Higher?** viene creata dall'espressione:`IF(Sum(Sales)>Below(Sum(Sales)), 'Higher')`. Questa esegue il confronto dei valori della riga attuale nella misura **Sum(Sales)** con le righe sotto di essa. Se la riga attuale presenta un valore più alto, viene restituito "Higher".

Questa funzione può essere utilizzata nei grafici diversi dalle tabelle, ad esempio nei grafici a barre.

Per gli altri tipi di grafici, convertire il grafico nell'equivalente di tabella lineare per individuare facilmente la riga a cui si riferisce la funzione.

Per i grafici con più dimensioni, i risultati delle espressioni contenenti le funzioni **Above**, **Below**, **Top** e **Bottom** dipendono dalla modalità con cui le dimensioni di colonna vengono ordinate da Qlik Sense. Qlik Sense valuta le funzioni in base ai segmenti di colonna che derivano dalla dimensione ordinata per ultima. L'ordinamento delle colonne viene controllato dal pannello delle proprietà in **Ordinamento** e non è necessariamente l'ordine in cui le colonne vengono visualizzate in una tabella.Per ulteriori informazioni, fare riferimento all'esempio 2 nella sezione relativa alla funzione **Above**.

Esempio 2

Example 2:	Risultato
La funzione **Below** può essere utilizzata come input per le funzioni di scala. Ad esempio: `RangeAvg (Below(Sum(Sales),1,3)).`	Negli argomenti per la funzione **Below()**, offset viene impostato su 1 e count viene impostato su 3. La funzione trova i risultati dell'espressione **Sum (Sales)** nelle tre righe immediatamente sotto la riga attuale nel segmento di colonna (dove si trova una riga). Questi tre valori vengono utilizzati come input per la funzione RangeAvg(), che individua la media dei valori nella scala di numeri fornita. Una tabella in cui è stato impostato **Customer** come dimensione restituisce i seguenti risultati per l'espressione RangeAvg(). Astrida 659.67 Betacab 720 Canutility 757 Divadip: -

Dati utilizzati negli esempi:

```
Monthnames:
LOAD *, Dual(MonthText,MonthNumber) as Month INLINE [
MonthText, MonthNumber
Jan, 1
Feb, 2
Mar, 3
Apr, 4
May, 5
Jun, 6
Jul, 7
Aug, 8
Sep, 9
Oct, 10
Nov, 11
Dec, 12
];

Sales2013:
Crosstable (MonthText, Sales) LOAD * inline [
Customer|Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec
Astrida|46|60|70|13|78|20|45|65|78|12|78|22
Betacab|65|56|22|79|12|56|45|24|32|78|55|15
Canutility|77|68|34|91|24|68|57|36|44|90|67|27
Divadip|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');
```

Vedere anche:

p *Above - funzione per grafici (page 1267)*
p *Bottom - funzione per grafici (page 1276)*
p *Top - funzione per grafici (page 1302)*
p *RangeAvg (page 1324)*

Bottom - funzione per grafici

Bottom() valuta un'espressione nell'ultima riga (inferiore) di un segmento di colonna in una tabella. La riga per la quale viene calcolata dipende dal valore dell'**offset**, se presente, e l'impostazione predefinita è la riga direttamente nella parte inferiore. Per i grafici diversi dalle tabelle, la valutazione viene effettuata sull'ultima riga della colonna attuale nell'equivalente di tabella lineare del grafico.

Sintassi:

```
Bottom([TOTAL] expr [ , offset [,count ]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
offset	Specificando un **offset**n maggiore di 1, la valutazione dell'espressione viene spostata di n righe sopra rispetto alla riga inferiore. Specificando un numero di offset negativo, la funzione **Bottom** diventa equivalente alla funzione **Top** con il numero di offset positivo corrispondente.
count	Specificando un terzo parametro **count** maggiore di 1, la funzione restituirà non un solo valore, ma una scala di valori **count**, uno per ciascuna delle ultime righe **count** del segmento di colonna attuale. In questo modulo la funzione può essere utilizzata come argomento per una qualsiasi delle funzioni di scala speciali. *Funzioni di scala (page 1321)*
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Un segmento di colonna viene definito come un sottogruppo consecutivo di celle con gli stessi valori per le dimensioni nell'ordine attuale. Le funzioni grafiche intra-record vengono calcolate nel segmento colonna escludendo la dimensione più a destra nel grafico della tabella lineare equivalente. Se nel grafico è presente una sola dimensione oppure è specificato il qualificatore TOTAL, l'espressione viene valutata nell'intera tabella.

Se la tabella o l'equivalente di tabella include più dimensioni verticali, il segmento colonna attuale includerà solo righe contenenti gli stessi valori della riga attuale in tutte le colonne di dimensione, eccetto la colonna che mostra l'ultima dimensione nell'ordinamento tra campi.

Limiti:

- Le chiamate ricorrenti restituiranno NULL.
- L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Esempi e risultati:

Visualizzazione della tabella per l'esempio 1

Customer	Sum([Sales])	Bottom(Sum(Sales))	Sum(Sales)+Bottom(Sum(Sales))	Bottom offset 3
	2566	**757**	**3323**	**3105**
Astrida	587	757	1344	1126
Betacab	539	757	1296	1078
Canutility	683	757	1440	1222
Divadip	757	757	1514	1296

Nella schermata della tabella mostrata in questo esempio, la visualizzazione della tabella viene creata dalla dimensione **Customer** e dalle misure: `Sum(Sales)` e `Bottom(Sum(Sales))`.

La colonna **Bottom(Sum(Sales))** restituisce 757 per tutte le righe perché questo è il valore dell'ultima riga: **Divadip**.

Nella tabella sono mostrate anche misure più complesse: una creata da `Sum(Sales)+Bottom(Sum(Sales))` e una con etichetta **Bottom offset 3** che viene creata utilizzando l'espressione `Sum(Sales)+Bottom(Sum(Sales), 3)` e in cui l'argomento **offset** è impostato su 3. Aggiunge il valore **Sum(Sales)** per la riga attuale al valore della riga tre righe sopra l'ultima riga, ossia, la riga attuale più il valore per **Betacab**.

Esempio: 2

Nelle schermate delle tabelle mostrate in questo esempio sono state aggiunte altre dimensioni alle visualizzazioni: **Month** e **Product**. Per i grafici con più dimensioni, i risultati delle espressioni contenenti le funzioni **Above**, **Below**, **Top** e **Bottom** dipendono dalla modalità con cui le dimensioni di colonna vengono ordinate da Qlik Sense. Qlik Sense valuta le funzioni in base ai segmenti di colonna che derivano dalla dimensione ordinata per ultima. L'ordinamento delle colonne viene controllato dal pannello delle proprietà in **Ordinamento** e non è necessariamente l'ordine in cui le colonne vengono visualizzate in una tabella.

Nella prima tabella l'espressione viene valutata in base a **Month** e nella seconda tabella viene valutata in base a **Product**. La misura **End value** contiene l'espressione `Bottom(Sum(Sales))`. La riga inferiore per **Month** è Dec e il valore per Dec, per entrambi i valori di **Product** mostrati nella schermata, è 22. (alcune righe sono state modificate fuori dalla schermata per risparmiare spazio).

Prima tabella per l'esempio 2. Il valore di Bottom per la misura End value basata su Month (Dec).

Customer	Product	Month	Sum(Sales)	End value
			2566	-
Astrida	AA	Jan	46	22
Astrida	AA	Feb	60	22
Astrida	AA	Mar	70	22
Astrida	AA	Sep	78	22
Astrida	AA	Oct	12	22
Astrida	AA	Nov	78	22
Astrida	AA	Dec	22	22
Astrida	BB	Jan	46	22

Seconda tabella per l'esempio 2. Il valore di Bottom per la misura End value basata su Product (BB per Astrida).

Customer	Product	Month	Sum(Sales)	End value
			2566	-
Astrida	AA	Jan	46	46
Astrida	BB	Jan	46	46
Astrida	AA	Feb	60	60
Astrida	BB	Feb	60	60
Astrida	AA	Mar	70	70
Astrida	BB	Mar	70	70
Astrida	AA	Apr	13	13
Astrida	BB	Apr	13	13

Per ulteriori informazioni, fare riferimento all'esempio 2 nella sezione relativa alla funzione **Above**.

Esempio 3

Esempio: 3	Risultato
La funzione **Bottom** può essere utilizzata come input per le funzioni di scala. Ad esempio: `RangeAvg (Bottom(Sum(Sales),1,3)).`	Negli argomenti per la funzione **Bottom()**, offset viene impostato su 1 e count viene impostato su 3. La funzione trova i risultati dell'espressione **Sum (Sales)** nelle tre righe che iniziano con la riga immediatamente sopra la riga inferiore nel segmento colonna (perché offset=1) e le due righe immediatamente sopra a essa (dove si trova una riga). Questi tre valori vengono utilizzati come input per la funzione RangeAvg(), che individua la media dei valori nella scala di numeri fornita. Una tabella in cui è stato impostato **Customer** come dimensione restituisce i seguenti risultati per l'espressione RangeAvg().
	Astrida 659.67 Betacab 659.67 Canutility 659.67 Divadip: 659.67

```
Monthnames:
LOAD *, Dual(MonthText,MonthNumber) as Month INLINE [
MonthText, MonthNumber
Jan, 1
Feb, 2
Mar, 3
Apr, 4
May, 5
Jun, 6
Jul, 7
Aug, 8
Sep, 9
Oct, 10
Nov, 11
Dec, 12
];

Sales2013:
Crosstable (MonthText, Sales) LOAD * inline [
Customer|Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec
Astrida|46|60|70|13|78|20|45|65|78|12|78|22
Betacab|65|56|22|79|12|56|45|24|32|78|55|15
Canutility|77|68|34|91|24|68|57|36|44|90|67|27
Divadip|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');
```

Vedere anche:

p *Top - funzione per grafici (page 1302)*

Column - funzione per grafici

Column() restituisce il valore trovato nella colonna corrispondente a **ColumnNo** in una tabella lineare, ignorando le dimensioni. Ad esempio **Column(2)** restituisce il valore della seconda colonna della misura.

Sintassi:

```
Column(ColumnNo)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
ColumnNo	Numero di una colonna nella tabella contenente una misura. *La funzione Column() ignora le colonne delle dimensioni.*

Limiti:

- Le chiamate ricorrenti restituiranno NULL.
- Se **ColumnNo** fa riferimento a una colonna per la quale non esiste una misura, viene restituito un valore NULL.
- L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Esempi e risultati:

Esempio: Percentuale delle vendite totali

Customer	Product	UnitPrice	UnitSales	Order Value	Total Sales Value	% Sales
A	AA	15	10	150	505	29.70
A	AA	16	4	64	505	12.67
A	BB	9	9	81	505	16.04

Customer	Product	UnitPrice	UnitSales	Order Value	Total Sales Value	% Sales
B	BB	10	5	50	505	9.90
B	CC	20	2	40	505	7.92
B	DD	25	-	0	505	0.00
C	AA	15	8	120	505	23.76
C	CC	19	-	0	505	0.00

Esempio: Percentuale delle vendite per il cliente selezionato

Customer	Product	UnitPrice	UnitSales	Order Value	Total Sales Value	% Sales
A	AA	15	10	150	295	50.85
A	AA	16	4	64	295	21.69
A	BB	9	9	81	295	27.46

Esempi e risultati

Esempi	Risultati
Order Value viene aggiunto alla tabella come misura con l'espressione: `Sum (UnitPrice*UnitSales)`. Total Sales Value viene aggiunto come misura con l'espressione: `Sum(TOTAL UnitPrice*UnitSales)` % Sales viene aggiunto come misura con l'espressione `100*Column(1)/Column(2)`	Il risultato di Column(1) deriva dalla colonna Order Value perché questa è la prima colonna della misura. Il risultato di Column(2) deriva da Total Sales Value perché questa è la seconda colonna della misura. Vedere i risultati nella colonna % Sales nell'esempio *Percentuale delle vendite totali (page 1280)*.
Selezionare Customer A.	La selezione modifica Total Sales Value e quindi %Sales. Vedere l'esempio *Percentuale delle vendite per il cliente selezionato (page 1281)*.

Dati utilizzati negli esempi:

```
ProductData:
LOAD * inline [
Customer|Product|UnitSales|UnitPrice
Astrida|AA|4|16
Astrida|AA|10|15
Astrida|BB|9|9
Betacab|BB|5|10
Betacab|CC|2|20
Betacab|DD||25
Canutility|AA|8|15
Canutility|CC||19
```

```
] (delimiter is '|');
```

Dimensionality - funzione per grafici

Dimensionality() restituisce il numero di dimensioni per la riga attuale. Nel caso delle tabelle pivot la funzione restituisce il numero totale di colonne di dimensione senza contenuti di aggregazione, ossia senza somme parziali o aggregati compressi.

Sintassi:

```
Dimensionality ( )
```

Tipo di dati restituiti: numero intero

Limiti:

Questa funzione è disponibile solo nei grafici. Per tutti gli altri tipi di grafici, eccetto le tabelle pivot, restituirà il numero di dimensioni in tutte le righe eccetto il totale, che sarà 0.

L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Esempio: Espressione del grafico usando la Dimensionalità

Esempio: Espressione del grafico
La funzione **Dimensionalità()** può essere usata con una tabella pivot come espressione del grafico dove si vuole applicare una formattazione diversa delle celle a seconda del numero di dimensioni in una riga che ha dati non aggregati. Questo esempio usa la funzione Dimensionalità() per applicare un colore di sfondo alle celle della tabella che corrispondono a una data condizione.

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare l'esempio di espressione del grafico in basso.

```
ProductSales:

Load * inline [
Country,Product,Sales,Budget
Sweden,AA,100000,50000
Germany,AA,125000,175000
Canada,AA,105000,98000
Norway,AA,74850,68500
Ireland,AA,49000,48000
Sweden,BB,98000,99000
Germany,BB,115000,175000
Norway,BB,71850,68500
Ireland,BB,31000,48000
] (delimiter is ',');
```

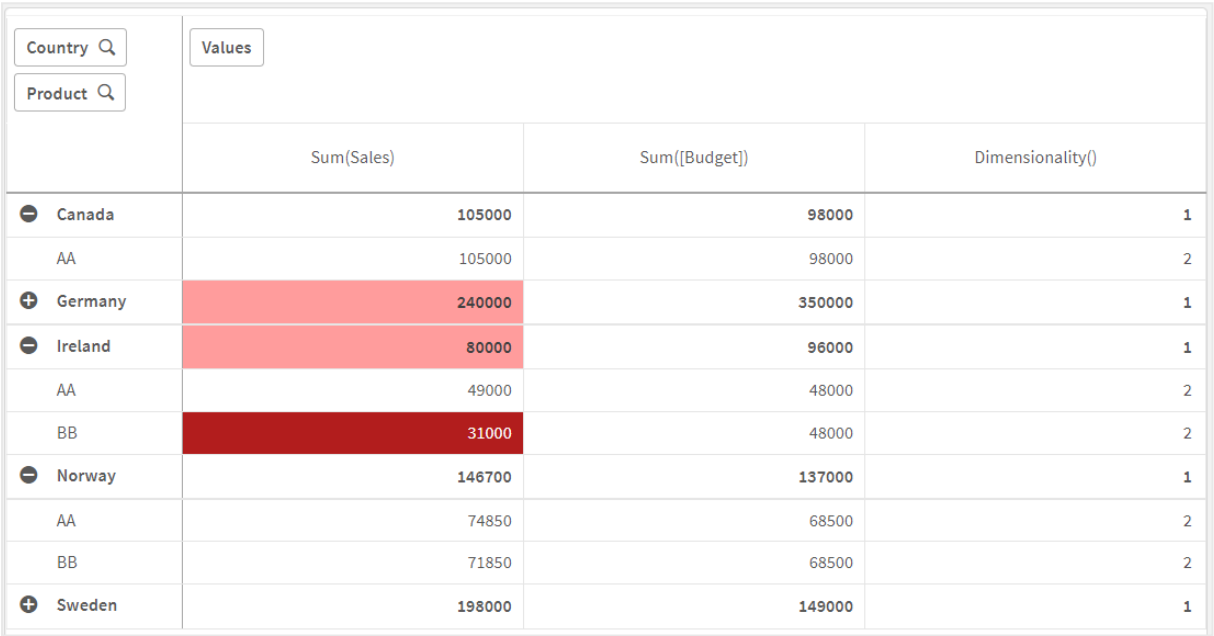
Espressione del grafico

Creare una visualizzazione tabella pivot in un foglio Qlik Sense con **Paese** e **Prodotto** come dimensioni. Aggiungere **Sum(Sales)**, **Sum(Budget)** e **Dimensionality()** come misure.

Nel pannello **Proprietà**, inserire l'espressione seguente come **Espressione colore sfondo** per la misura **Sum(Sales)**:

```
If(Dimensionality()=1 and Sum(Sales)<Sum(Budget),RGB(255,156,156),
If(Dimensionality()=2 and Sum(Sales)<Sum(Budget),RGB(178,29,29)
))
```

Risultato:



Country / Product	Sum(Sales)	Sum([Budget])	Dimensionality()
Canada	105000	98000	1
AA	105000	98000	2
Germany	240000	350000	1
Ireland	80000	96000	1
AA	49000	48000	2
BB	31000	48000	2
Norway	146700	137000	1
AA	74850	68500	2
BB	71850	68500	2
Sweden	198000	149000	1

Spiegazione

L'espressione `If(Dimensionality()=1 and Sum(Sales)<Sum(Budget),RGB(255,156,156), If (Dimensionality()=2 and Sum(Sales)<Sum(Budget),RGB(178,29,29)))` contiene istruzioni condizionali che controllano il valore di dimensionalità e Sum(Sales) e Sum(Budget) per ciascun prodotto. Se le condizioni vengono rispettate, viene applicato un colore di sfondo al valore Sum(Sales).

Exists

Exists() determina se un valore di campo specifico è già stato caricato nel campo nello script di caricamento dei dati. La funzione restituisce TRUE o FALSE, quindi può essere utilizzata nella clausola **where** di un'istruzione **LOAD** o un'istruzione **IF**.

*È possibile utilizzare **Not Exists()** per determinare se un valore di campo non è stato caricato, ma occorre prestare attenzione se si usa **Not Exists()** in una clausola Where. La funzione **Exists()** verifica sia le tabelle caricate in precedenza che i valori caricati in precedenza nella tabella corrente. Verrà quindi caricata solo la prima occorrenza. Quando viene rilevata la seconda occorrenza, il valore sarà già caricato. Vedere gli esempi per ulteriori informazioni.*

Sintassi:

```
Exists(field_name [, expr])
```

Tipo di dati restituiti: Booleano

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Nome del campo in cui si desidera cercare un valore. È possibile utilizzare un nome di campo esplicito senza virgolette. Il campo deve già essere caricato dallo script. Ciò significa che non è possibile fare riferimento a un campo caricato in una clausola che si trova più in basso nello script.
expr	Valore del quale si vuole verificare l'esistenza. È possibile utilizzare un valore esplicito o un'espressione che fa riferimento a uno o più campi nell'istruzione LOAD corrente. *Non è possibile fare riferimento a campi non inclusi nell'istruzione LOAD corrente.* Questo argomento è facoltativo. Se viene omesso, la funzione verificherà se il valore di **field_name** nel record corrente è già esistente.

Esempi e risultati:

Esempio 1

```
Exists (Employee)
```

Restituisce -1 (True) se il valore di campo **Employee** nel record attuale esiste già in uno qualsiasi dei record letti in precedenza contenenti tale campo.

Le istruzioni `Exists (Employee, Employee)` e `Exists (Employee)` sono equivalenti.

Esempio 2

```
Exists(Employee, 'Bill')
```

Restituisce -1 (True) se il valore di campo **'Bill'** viene trovato nel contenuto attuale del campo **Employee**.

Esempio 3

```
Employees: LOAD * inline [ Employee|ID|Salary Bill|001|20000 John|002|30000 Steve|003|35000 ]
(delimiter is '|');  Citizens: Load * inline [ Employee|Address Bill|New York Mary|London
Steve|Chicago Lucy|Madrid Lucy|Paris John|Miami ] (delimiter is '|') where Exists (Employee);
Drop Tables Employees;
```

Viene generata una tabella che può essere usata in una visualizzazione di tabella con le dimensioni Employee e Address.

La clausola where, `where Exists (Employee)`, significa che nella nuova tabella vengono caricati solo i nomi della tabella Citizens che sono presenti anche in Employees. L'istruzione Drop rimuove la tabella Employees per evitare confusione.

Risultati

Employee	Address
Bill	New York
John	Miami
Steve	Chicago

Esempio 4

```
Employees: Load * inline [  Employee|ID|Salary Bill|001|20000 John|002|30000 Steve|003|35000 ]
(delimiter is '|');   Citizens: Load * inline [ Employee|Address Bill|New York Mary|London
Steve|Chicago Lucy|Madrid Lucy|Paris John|Miami ] (delimiter is '|') where not Exists
(Employee);   Drop Tables Employees;
```

La clausola where include not: `where not Exists (Employee)`.

Ciò significa che nella nuova tabella vengono caricati solo i nomi della tabella Citizens che non sono presenti in Employees.

Si noti che sono presenti due valori per Lucy nella tabella Citizens, ma solo uno è incluso nella tabella risultante. Quando si carica la prima riga con il valore Lucy, esso viene incluso nel campo Employee. Quindi, quando viene controllata la seconda riga, il valore esiste già.

Risultati

Employee	**Indirizzo**
Mary	London
Lucy	Madrid

Esempio 5

Questo esempio mostra come caricare tutti i valori.

```
Employees: Load Employee As Name; LOAD * inline [ Employee|ID|Salary Bill|001|20000
John|002|30000 Steve|003|35000 ] (delimiter is '|');   Citizens: Load * inline [
Employee|Address Bill|New York Mary|London Steve|Chicago Lucy|Madrid Lucy|Paris John|Miami ]
(delimiter is '|') where not Exists (Name, Employee);   Drop Tables Employees;
```

Per ottenere tutti i valori per Lucy è stato necessario effettuare due modifiche:

- È stato inserito un precedente caricamento per la tabella Employees in cui Employee è stato rinominato Name.
  `Load Employee As Name;`
- La condizione Where in Citizens è stata modificata in:
  `not Exists (Name, Employee).`

Ciò crea campi per Name e Employee. Quando viene controllata la seconda riga con Lucy, il valore è ancora inesistente in Name.

Risultati

Employee	Indirizzo
Mary	London
Lucy	Madrid
Lucy	Paris

FieldIndex

FieldIndex() restituisce la posizione del valore di campo **value** presente nel campo **field_name** (in ordine di caricamento).

Sintassi:

```
FieldIndex(field_name , value)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Nome del campo per cui è richiesto l'indice. Ad esempio, la colonna in una tabella. Deve essere specificato come valore di stringa. Questo significa che il nome di campo deve essere racchiuso tra virgolette singole.
value	Il valore del campo **field_name**.

Limiti:

- Se non è possibile trovare **value** tra i valori del campo **field_name**, verrà restituito 0.
- L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si

utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione. Questa limitazione non si applica alla funzione di script equivalente.

Esempi e risultati:

Gli esempi seguenti utilizzano il campo: **First name** dalla tabella **Names**.

Esempi e risultati

Esempi	Risultati
Aggiungere i dati campione all'app ed eseguirli.	La tabella **Names** viene caricata come nei seguenti dati campione.
Funzione grafica: in una tabella contenente la dimensione First name, aggiungere come misura:	
`FieldIndex ('First name','John')`	1, in quanto 'John' è visualizzato per primo nell'ordine di caricamento del campo **First name**. Tenere presente che in una casella di filtro **John** sarebbe stato visualizzato come numero 2 a partire dall'alto, in quanto è ordinato in modo alfabetico e non secondo l'ordine di caricamento.
`FieldIndex ('First name','Peter')`	4, in quanto **FieldIndex()** restituisce solo un valore, che corrisponde alla prima occorrenza nell'ordine di caricamento.
Funzione script: la tabella **Names** viene caricata come nei seguenti dati campione:	

`John1:`
`Load FieldIndex('First name','John') as MyJohnPos`
`Resident Names;`
	`MyJohnPos=`1, in quanto 'John' è visualizzato per primo nell'ordine di caricamento del campo **First name**. Tenere presente che in una casella di filtro **John** sarebbe stato visualizzato come numero 2 a partire dall'alto, in quanto è ordinato in modo alfabetico e non secondo l'ordine di caricamento.

`Peter1:`
`Load FieldIndex('First name','Peter') as MyPeterPos`
`Resident Names;`
	`MyPeterPos=`4, in quanto **FieldIndex()** restituisce un solo valore, che corrisponde alla prima occorrenza nell'ordine di caricamento.

Dati utilizzati nell'esempio:

```
Names:
LOAD * inline [
First name|Last name|Initials|Has cellphone
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC|No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');
```

```
John1:
Load FieldIndex('First name','John') as MyJohnPos
Resident Names;

Peter1:
Load FieldIndex('First name','Peter') as MyPeterPos
Resident Names;
```

FieldValue

FieldValue() restituisce il valore trovato nella posizione **elem_no** del campo **field_name** (in ordine di caricamento).

Sintassi:

```
FieldValue(field_name , elem_no)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Nome del campo per cui è richiesto il valore. Ad esempio, la colonna in una tabella. Deve essere specificato come valore di stringa. Questo significa che il nome di campo deve essere racchiuso tra virgolette singole.
elem_no	Il numero della posizione (elemento) del campo, che segue l'ordine di caricamento, per cui viene restituito il valore. Ciò potrebbe corrispondere alla riga in una tabella, anche se dipende dall'ordine di caricamento degli elementi (righe).

Limiti:

- Se **elem_no** è maggiore del numero di valori del campo, viene restituito NULL .
- L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione. Questa limitazione non si applica alla funzione di script equivalente.

Esempio

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare l'esempio in basso.

```
Names:

LOAD * inline [
```

```
First name|Last name|Initials|Has cellphone
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC |No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');

John1:

Load FieldValue('First name',1) as MyPos1
Resident Names;

Peter1:

Load FieldValue('First name',5) as MyPos2
Resident Names;
```

Creazione di una visualizzazione

Creare una visualizzazione di tabelle in un foglio Qlik Sense. Aggiungere i campi **First name**, **MyPos1** e **MyPos2** alla tabella.

Risultato

First name	MyPos1	MyPos2
Jane	John	Jane
John	John	Jane
Mark	John	Jane
Peter	John	Jane
Sue	John	Jane

Spiegazione

FieldValue('First name','1') restituisce John come valore per **MyPos1** per tutti i nomi perché John appare prima nell'ordine di caricamento del campo **Nome**. Tenere presente che in una casella di filtro John sarebbe stato visualizzato come numero 2 a partire dall'alto, dopo Jane, in quanto è ordinato in modo alfabetico e non secondo l'ordine di caricamento.

FieldValue('First name','5') restituisce Jane come valore per **MyPos2** per tutti i nomi, dato che Jane appare quinta nell'ordine di caricamento del campo **First name**.

FieldValueCount

FieldValueCount() è una funzione **integer** che restituisce il numero di valori distinti di un campo.

Un ricaricamento parziale può rimuovere valori dai dati, che non si rifletteranno nel numero restituito. Il numero restituito corrisponderà a tutti i valori distinti che sono stati caricati nel ricaricamento iniziale o in qualsiasi ricaricamento parziale successivo.

L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione. Questa limitazione non si applica alla funzione di script equivalente.

Sintassi:

```
FieldValueCount(field_name)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Nome del campo per cui è richiesto il valore. Ad esempio, la colonna in una tabella. Deve essere specificato come valore di stringa. Questo significa che il nome di campo deve essere racchiuso tra virgolette singole.

Esempi e risultati:

Gli esempi seguenti utilizzano il campo **First name** dalla tabella **Names**.

Esempi e risultati

Esempi	Risultati
Aggiungere i dati campione all'app ed eseguirli.	La tabella **Names** viene caricata come nei seguenti dati campione.
Funzione grafica: in una tabella contenente la dimensione First name, aggiungere come misura:	
`FieldValueCount('First name')`	5 in quanto **Peter** è visualizzato due volte.
`FieldValueCount('Initials')`	6 in quanto **Initials** presenta solo due valori distinti.
Funzione script: la tabella **Names** viene caricata come nei seguenti dati campione:	

`FieldCount1:`
`Load FieldValueCount('First name') as MyFieldCount1`
`Resident Names;`
	`MyFieldCount1`=5, in quanto 'Peter' è visualizzato due volte.

`FieldCount2:`
`Load FieldValueCount('Initials') as MyInitialsCount1`
`Resident Names;`
	`MyFieldCount1`=6, in quanto 'Initials' presenta solo due valori distinti.

Dati utilizzati negli esempi:

```
Names:
LOAD * inline [
First name|Last name|Initials|Has cellphone
John|Anderson|JA|Yes
Sue|Brown|SB|Yes
Mark|Carr|MC|No
Peter|Devonshire|PD|No
Jane|Elliot|JE|Yes
Peter|Franc|PF|Yes ] (delimiter is '|');

FieldCount1:
Load FieldValueCount('First name') as MyFieldCount1
Resident Names;

FieldCount2:
Load FieldValueCount('Initials') as MyInitialsCount1
Resident Names;
```

LookUp

Lookup() osserva una tabella già caricata e restituisce il valore di **field_name** corrispondente alla prima occorrenza del valore **match_field_value** nel campo **match_field_name**. La tabella può essere la tabella attuale o un'altra tabella caricata in precedenza.

Sintassi:

```
lookup(field_name, match_field_name, match_field_value [, table_name])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Nome del campo per cui è richiesto il valore restituito. Il valore di input deve essere specificato come stringa (ad esempio, un valore letterale tra virgolette).
match_field_name	Nome del campo in cui ricercare **match_field_value**. Il valore di input deve essere specificato come stringa (ad esempio, un valore letterale tra virgolette).
match_field_value	Valore da ricercare nel campo **match_field_name**.
table_name	Nome della tabella in cui ricercare il valore. Il valore di input deve essere specificato come stringa (ad esempio un valore letterale tra virgolette). Se il valore **table_name** viene omesso, verrà utilizzata la tabella attuale.

Gli argomenti non racchiusi tra virgolette fanno riferimento alla tabella attuale. Per fare riferimento ad altre tabelle, racchiudere l'argomento tra virgolette singole.

Limiti:

L'ordine in cui viene eseguita la ricerca è l'ordine di caricamento, a meno che la tabella non sia il risultato di operazioni complesse come unioni, nel qual caso l'ordine non sarà definito in modo preciso. Sia **field_name** che **match_field_name** devono essere campi della stessa tabella, specificata da **table_name**.

Se non viene trovata una corrispondenza, viene restituito il valore NULL.

Esempio

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare l'esempio in basso.

```
ProductList: Load * Inline [ ProductID|Product|Category|Price 1|AA|1|1 2|BB|1|3 3|CC|2|8
4|DD|3|2 ] (delimiter is '|');   OrderData: Load *, Lookup('Category', 'ProductID', ProductID,
'ProductList') as CategoryID Inline [ InvoiceID|CustomerID|ProductID|Units 1|Astrida|1|8
1|Astrida|2|6 2|Betacab|3|10 3|Divadip|3|5 4|Divadip|4|10 ] (delimiter is '|');   Drop Table
ProductList;
```

Creazione di una visualizzazione

Creare una visualizzazione di tabelle in un foglio Qlik Sense. Aggiungere i campi **ProductID**, **InvoiceID**, **CustomerID**, **Units** e **CategoryID** alla tabella.

Risultato

Tabella risultante

ProductID	InvoiceID	CustomerID	Unità	CategoryID
1	1	Astrida	8	1
2	1	Astrida	6	1
3	2	Betacab	10	2
3	3	Divadip	5	2
4	4	Divadip	10	3

Spiegazione

I dati campione utilizzano la funzione **Lookup()** con il seguente formato:

```
Lookup('Category', 'ProductID', ProductID, 'ProductList')
```

La tabella **ProductList** viene caricata per prima.

La funzione **Lookup()** viene utilizzata per creare la tabella **OrderData**. Specifica il terzo argomento come **ProductID**. Si tratta del campo per cui il valore deve essere ricercato nel secondo argomento **'ProductID'** in **ProductList**, come indicato dalle virgolette singole che lo racchiudono.

La funzione restituisce il valore per '**Category**' (nella tabella **ProductList**), caricata come **CategoryID**.

L'istruzione **drop** elimina la tabella **ProductList** dal modello dati poiché non è richiesta, il che lascia la tabella **OrderData** risultante.

> *La funzione Lookup() è flessibile e può accedere a qualsiasi tabella caricata in precedenza. Tuttavia, risulta più lenta rispetto alla funzione Applymap().*

Vedere anche:

p *ApplyMap (page 1313)*

NoOfRows - funzione per grafici

NoOfRows() restituisce il numero di righe nel segmento di colonna attuale in una tabella. Per i grafici bitmap, **NoOfRows()** restituisce il numero di righe nell'equivalente della tabella lineare del grafico.

Se la tabella o l'equivalente di tabella include più dimensioni verticali, il segmento colonna attuale includerà solo righe contenenti gli stessi valori della riga attuale in tutte le colonne di dimensione, eccetto la colonna che mostra l'ultima dimensione nell'ordinamento tra campi.

> *L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.*

Sintassi:

```
NoOfRows([TOTAL])
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Esempio: Espressione del grafico usando NoOfRows

Esempio - Espressione del grafico

Script di caricamento

Caricare i dati seguenti come un caricamento inline nell'editor caricamento dati per creare gli esempi di espressione del grafico in basso.

```
        Temp:
LOAD * inline [
Region|SubRegion|RowNo()|NoOfRows()
Africa|Eastern
Africa|Western
Americas|Central
Americas|Northern
Asia|Eastern
Europe|Eastern
Europe|Northern
Europe|Western
Oceania|Australia
] (delimiter is '|');
```

Espressione del grafico

Crea una visualizzazione tabella in un foglio Qlik Sense con **Region** e **SubRegion** come dimensioni. Aggiungere `RowNo( ), NoOfRows()` e `NoOfRows(Total)` come misure.

Risultato

Region	SubRegion	RowNo()	NoOfRows()	NoOfRows (Total)
Africa	Eastern	1	2	9
Africa	Western	2	2	9
Americas	Central	1	2	9
Americas	Northern	2	2	9
Asia	Eastern	1	1	9
Europe	Eastern	1	3	9
Europe	Northern	2	3	9
Eurrope	Western	3	3	9
Oceania	Australia	1	1	9

Spiegazione

In questo esempio, il criterio di ordinamento è per la prima dimensione, Regione. Di conseguenza, ciascun segmento colonna è formato da un gruppo di regioni che presenta lo stesso valore, ad esempio, Africa.

La colonna **RowNo()** mostra i numeri di riga per ciascun segmento colonna, ad esempio, sono presenti due righe per la regione Africa. La numerazione delle righe riparte da 1 per il segmento di colonna successivo, vale a dire Americas.

La colonna **NoOfRows()** conteggia il numero di righe in ciascun segmento colonna, ad esempio, Europa ha tre righe nel segmento colonna.

La colonna **NoOfRows(Total)** ignora le dimensioni a causa dell'argomento `TOTAL` per `NoOfRows()` e conteggia le righe nella tabella.

Se la tabella è stata ordinata in base alla seconda dimensione, SubRegion, i segmenti colonna si baserebbero su tale dimensione, pertanto la numerazione delle righe cambierebbe per ciascun SubRegion.

Vedere anche:

p *RowNo - funzione per grafici (page 581)*

Peek

Peek() restituisce il valore di un campo in una tabella per una riga che è già stata caricata. È possibile specificare il numero di riga così come la tabella. Se non viene specificato alcun numero di riga, verrà utilizzato l'ultimo record precedentemente caricato.

La funzione peek() viene principalmente utilizzata per trovare i limiti pertinenti in una tabella precedentemente caricata, ovvero, il primo valore o l'ultimo valore di un campo specifico. In gran parte dei casi, questo valore viene archiviato in una variabile per l'utilizzo futuro, ad esempio, come una condizione in un loop do-while.

Sintassi:

```
Peek(
field_name
[, row_no[, table_name ] ])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Nome del campo per cui è richiesto il valore restituito.Il valore di input deve essere specificato come stringa (ad esempio, un valore letterale tra virgolette).
row_no	La riga nella tabella che specifica il campo richiesto. Può essere un'espressione, tuttavia deve restituire un numero intero. 0 indica il primo record, 1 indica il secondo e così via. I numeri negativi indicano l'ordine dalla fine della tabella. -1 indica l'ultimo record letto. Se non viene dichiarato alcun valore **row_no** verrà utilizzato il valore -1.
table_name	Un'etichetta di tabella senza due punti finali. Se non è dichiarato **table_name**, viene utilizzata la tabella attuale. Se utilizzato al di fuori dell'istruzione **LOAD** o se fa riferimento a un'altra tabella, è necessario includere **table_name**.

Limiti:

La funzione può solo restituire valori da record già caricati. Questo significa che nel primo record di una tabella, una chiamata che usa -1 come row_no restituirà NULL.

Esempi e risultati:

Esempio 1

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
EmployeeDates: Load * Inline [ EmployeeCode|StartDate|EndDate 101|02/11/2010|23/06/2012
102|01/11/2011|30/11/2013 103|02/01/2012| 104|02/01/2012|31/03/2012 105|01/04/2012|31/01/2013
106|02/11/2013| ] (delimiter is '|');       First_last_Employee: Load  EmployeeCode,  Peek
('EmployeeCode',0,'EmployeeDates') As FirstCode,  Peek('EmployeeCode',-1,'EmployeeDates') As
LastCode Resident EmployeeDates;
```

Tabella risultante

Codice dipendente	StartDate	EndDate	FirstCode	LastCode
101	02/11/2010	23/06/2012	101	106
102	01/11/2011	30/11/2013	101	106
103	02/01/2012		101	106
104	02/01/2012	31/03/2012	101	106
105	01/04/2012	31/01/2013	101	106
106	02/11/2013		101	106

FirstCode = 101 poiché `Peek('EmployeeCode',0, 'EmployeeDates')` restituisce il primo valore di EmployeeCode nella tabella EmployeeDates.

LastCode = 106 perché `Peek('EmployeeCode',-1, 'EmployeeDates')` restituisce l'ultimo valore di EmployeeCode nella tabella EmployeeDates.

La sostituzione del valore dell'argomento **row_no** restituisce i valori delle altre righe nella tabella, nel modo seguente:

`Peek('EmployeeCode',2, 'EmployeeDates')` restituisce il terzo valore, 103, nella tabella come FirstCode.

Tuttavia, tenere presente che se non si specifica la tabella come il terzo argomento **table_name** in questi esempi, la funzione farà riferimento alla tabella attuale (in questo caso, interna).

Esempio 2

Se si desidera accedere ai dati più in profondità in una tabella, è possibile farlo in due passaggi: prima, caricare l'intera tabella in una tabella temporanea, quindi riordinarla quando si usa **Peek()**.

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
T1: LOAD * inline [ ID|Value 1|3 1|4 1|6 3|7 3|8 2|1 2|11 5|2 5|78 5|13 ] (delimiter is '|');
T2:  LOAD *, IF(ID=Peek('ID'), Peek('List')&','&Value,Value) AS List RESIDENT T1 ORDER BY ID
ASC; DROP TABLE T1;
```

Create a table in a sheet in your app with **ID**, **List**, and **Value** as the dimensions.

Tabella risultante

ID	Elenco	Valore
1	3,4	4
1	3,4,6	6
1	3	3
2	1,11	11
2	1	1
3	7,8	8
3	7	7
5	2,78	78
5	2,78,13	13
5	2	2

L'istruzione **IF()** viene creata a partire dalla tabella temporanea T1.
`Peek('ID')` fa riferimento al campo ID nella riga precedente nella tabella corrente T2.
`Peek('List')` fa riferimento al campo List nella riga precedente nella tabella T2, attualmente in fase di creazione mentre l'espressione viene valutata.

L'istruzione viene valutata nel seguente modo:
Se il valore attuale di ID è identico al valore precedente di ID, scrivere il valore di Peek('List') concatenandolo con il valore corrente di Value. In alternativa, scrivere solo il valore attuale di Value.

Se Peek('List') contiene già un risultato concatenato, il nuovo risultato di Peek('List') verrà concatenato a tale risultato.

*Prendere nota della clausola **Order by**. Questa clausola specifica il metodo di ordinamento della tabella (in base all'ID in ordine crescente). Senza questa clausola, la funzione Peek() utilizzerebbe qualsiasi ordinamento arbitrario presente nella tabella interna, portando a risultati potenzialmente imprevedibili.*

Esempio 3

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
Amounts: Load  Date#(Month,'YYYY-MM') as Month, Amount, Peek(Amount) as AmountMonthBefore
Inline [Month,Amount 2022-01,2 2022-02,3 2022-03,7 2022-04,9 2022-05,4 2022-06,1];
```

Tabella risultante

Importo	AmountMonthBefore	Mese
1	4	2022-06
2	-	2022-01
3	2	2022-02
4	9	2022-05
7	3	2022-03
9	7	2022-04

Il campo AmountMonthBefore conterrà l'importo dal mese precedente.

Qui i parametri row_no e table_name vengono omessi, pertanto vengono utilizzati i valori predefiniti. In questo esempio, le seguenti tre chiamate di funzione sono equivalenti:

- Peek(Amount)
- Peek(Amount,-1)
- Peek(Amount,-1,'Amounts')

L'utilizzo di -1 come row_no indica che verrà utilizzato il valore dalla riga precedente. Sostituendo tale valore, sarà possibile recuperare i valori delle altre righe nella tabella:

Peek(Amount,2) restituisce il terzo valore nella tabella: 7.

Esempio 4

I dati devono essere ordinati correttamente per ottenere i risultati corretti ma, sfortunatamente, questo non è sempre il caso. Inoltre, la funzione Peek() non può essere utilizzata per fare riferimento a dati che non sono ancora stati caricati. Utilizzando tabelle temporanee ed eseguendo più passaggi attraverso i dati, questi problemi possono essere evitati.

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
tmp1Amounts: Load * Inline [Month,Product,Amount 2022-01,B,3 2022-01,A,8 2022-02,B,4 2022-02,A,6 2022-03,B,1 2022-03,A,6 2022-04,A,5 2022-04,B,5 2022-05,B,6 2022-05,A,7 2022-06,A,4 2022-06,B,8];  tmp2Amounts: Load *, If(Product=Peek(Product),Peek(Amount)) as AmountMonthBefore Resident tmp1Amounts Order By Product, Month Asc;  Drop Table tmp1Amounts; Amounts: Load *, If(Product=Peek(Product),Peek(Amount)) as AmountMonthAfter Resident tmp2Amounts Order By Product, Month Desc;  Drop Table tmp2Amounts;
```

Spiegazione

La tabella iniziale è ordinata in base al mese, il che significa che la funzione peek() restituirebbe in molti casi l'importo del prodotto sbagliato. Quindi, questa tabella deve essere riordinata. Ciò avviene eseguendo un secondo passaggio attraverso i dati creando una nuova tabella tmp2Amounts. Prendere nota della clausola Order by. Ordina i record prima per prodotto, poi per mese in ordine crescente.

La funzione If() è necessaria poiché AmountMonthBefore dovrebbe essere calcolato solo se la riga precedente contiene i dati per lo stesso prodotto ma per il mese precedente. Confrontando il prodotto sulla riga corrente con il prodotto sulla riga precedente, questa condizione può essere convalidata.

Quando viene creata la seconda tabella, la prima tabella tmp1Amounts viene eliminata utilizzando un'istruzione Drop Table.

Infine, viene fatto un terzo passaggio attraverso i dati, ma ora con i mesi ordinati in ordine inverso. In questo modo, può essere calcolato anche AmountMonthAfter.

Le clausole Ordina per specificano come è ordinata la tabella; senza di loro, la funzione Peek() userà qualsiasi ordine arbitrario della tabella interna, il che può portare a risultati imprevedibili.

Risultato

Tabella risultante

Mese	Product	Importo	AmountMonthBefore	AmountMonthAfter
2022-01	Una	8	-	6
2022-02	B	3	-	4
2022-03	Una	6	8	6
2022-04	B	4	3	1
2022-05	Una	6	6	5
2022-06	B	1	4	5
2022-01	Una	5	6	7
2022-02	B	5	1	6
2022-03	Una	7	5	4
2022-04	B	6	5	8
2022-05	Una	4	7	-
2022-06	B	8	6	-

Esempio 5

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
T1: Load * inline [ Quarter, Value 2003q1, 10000 2003q1, 25000 2003q1, 30000 2003q2, 1250
2003q2, 55000 2003q2, 76200 2003q3, 9240 2003q3, 33150 2003q3, 89450 2003q4, 1000 2003q4, 3000
2003q4, 5000 2004q1, 1000 2004q1, 1250 2004q1, 3000 2004q2, 5000 2004q2, 9240 2004q2, 10000
2004q3, 25000 2004q3, 30000 2004q3, 33150 2004q4, 55000 2004q4, 76200 2004q4, 89450 ];  T2:
Load *, rangesum(SumVal,peek('AccSumVal')) as AccSumVal; Load Quarter, sum(Value) as SumVal
resident T1 group by Quarter;
```

Risultato

Tabella risultante

Trimestre	SumVal	AccSumVal
2003q1	65000	65000
2003q2	132450	197450
2003q3	131840	329290
2003q4	9000	338290
2004q1	5250	343540
2004q2	24240	367780
2004q3	88150	455930
2004q4	220650	676580

Spiegazione

L'istruzione LOAD **Load *, rangesum(SumVal,peek('AccSumVal')) as AccSumVal** include una chiamata ricorsiva dove i valori precedenti vengono aggiunti al valore corrente. Questa operazione è utilizzata per calcolare un accumulo di valori nello script.

Vedere anche:

Previous

Previous() restituisce il valore dell'espressione **expr** utilizzando i dati del record di input precedente che non è stato eliminato a causa di una clausola **where**. Nel primo record di una tabella interna, la funzione restituirà NULL.

Sintassi:

```
Previous(expr)
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare. L'espressione può contenere funzioni **previous()** nidificate per accedere ai record precedenti. I dati vengono caricati direttamente dalla sorgente di input, rendendo possibile fare riferimento anche a campi che non sono stati caricati in Qlik Sense, vale a dire persino se non sono stati memorizzati nel relativo database associativo.

Limiti:

Nel primo record di una tabella interna, la funzione restituisce NULL.

Esempio:

Inserire quanto riportato di seguito nel proprio script di caricamento

```
Sales2013:
Load *, (Sales - Previous(Sales) )as Increase Inline [
Month|Sales
1|12
2|13
3|15
4|17
5|21
6|21
7|22
8|23
9|32
10|35
11|40
12|41
] (delimiter is '|');
```

Utilizzando la funzione **Previous()** nell'istruzione **Load**, è possibile confrontare il valore attuale di Sales con il valore precedente e utilizzarlo in un terzo campo, Increase.

Tabella risultante

Mese	Sales	Incremento
1	12	-
2	13	1
3	15	2
4	17	2
5	21	4
6	21	0
7	22	1
8	23	1
9	32	9
10	35	3
11	40	5
12	41	1

Top - funzione per grafici

Top() valuta un'espressione nella prima riga (superiore) di un segmento colonna in una tabella. La riga per la quale viene calcolata dipende dal valore dell'**offset**, se presente, e l'impostazione predefinita è la riga superiore. Per i grafici diversi dalle tabelle, la valutazione di **Top()** viene effettuata sulla prima riga della colonna attuale nell'equivalente di tabella lineare del grafico.

Sintassi:

```
Top([TOTAL] expr [ , offset [,count ]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
offset	Specificando un **offset** di n maggiore di 1, la valutazione dell'espressione viene spostata in basso di n righe rispetto alla riga superiore. Specificando un numero di offset negativo, la funzione **Top** diventa equivalente alla funzione **Bottom** con il numero di offset positivo corrispondente.
count	Specificando un terzo parametro **count** maggiore di 1, la funzione restituirà una scala di valori **count**, uno per ciascuna delle ultime righe **count** del segmento di colonna attuale. In questo modulo la funzione può essere utilizzata come argomento per una qualsiasi delle funzioni di scala speciali. *Funzioni di scala (page 1321)*
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Un segmento di colonna viene definito come un sottogruppo consecutivo di celle con gli stessi valori per le dimensioni nell'ordine attuale. Le funzioni grafiche intra-record vengono calcolate nel segmento colonna escludendo la dimensione più a destra nel grafico della tabella lineare equivalente. Se nel grafico è presente una sola dimensione oppure è specificato il qualificatore TOTAL, l'espressione viene valutata nell'intera tabella.

Se la tabella o l'equivalente di tabella include più dimensioni verticali, il segmento colonna attuale includerà solo righe contenenti gli stessi valori della riga attuale in tutte le colonne di dimensione, eccetto la colonna che mostra l'ultima dimensione nell'ordinamento tra campi.

Limiti:

- Le chiamate ricorrenti restituiranno NULL.
- L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Esempi e risultati:

Esempio: 1

Nella schermata della tabella mostrata in questo esempio, la visualizzazione della tabella viene creata dalla dimensione **Customer** e dalle misure: `Sum(Sales)` e `Top(Sum(Sales))`.

La colonna **Top(Sum(Sales))** restituisce 587 per tutte le righe perché questo è il valore della riga più in alto: **Astrida**.

Nella tabella sono mostrate anche misure più complesse: una creata da `Sum(Sales)+Top(Sum(Sales))` e una con etichetta **Top offset 3** che viene creata utilizzando l'espressione `Sum(Sales)+Top(Sum(Sales), 3)` e in cui l'argomento **offset** è impostato su 3. Aggiunge il valore **Sum(Sales)** per la riga attuale al valore della terza riga a partire dalla prima riga, ossia, la riga attuale più il valore per **Canutility**.

Esempio 1

Top and Bottom

Customer	Sum(Sales)	Top(Sum(Sales))	Sum(Sales)+Top(Sum(Sales))	Top offset 3
Totals	2566	587	3153	3249
Astrida	587	587	1174	1270
Betacab	539	587	1126	1222
Canutility	683	587	1270	1366
Divadip	757	587	1344	1440

Esempio: 2

Nelle schermate delle tabelle mostrate in questo esempio sono state aggiunte altre dimensioni alle visualizzazioni: **Month** e **Product**. Per i grafici con più dimensioni, i risultati delle espressioni contenenti le funzioni **Above**, **Below**, **Top** e **Bottom** dipendono dalla modalità con cui le dimensioni di colonna vengono ordinate da Qlik Sense. Qlik Sense valuta le funzioni in base ai segmenti di colonna che derivano dalla dimensione ordinata per ultima. L'ordinamento delle colonne viene controllato dal pannello delle proprietà in **Ordinamento** e non è necessariamente l'ordine in cui le colonne vengono visualizzate in una tabella.

Prima tabella per l'esempio 2. Il valore di Top per la misura First value basata su Month (Jan).

Customer	Product	Month	Sum(Sales)	First value
			2566	-
Astrida	AA	Jan	46	46
Astrida	AA	Feb	60	46
Astrida	AA	Mar	70	46
Astrida	AA	Apr	13	46
Astrida	AA	May	78	46
Astrida	AA	Jun	20	46
Astrida	AA	Jul	45	46
Astrida	AA	Aug	65	46
Astrida	AA	Sep	78	46
Astrida	AA	Oct	12	46
Astrida	AA	Nov	78	46
Astrida	AA	Dec	22	46

Seconda tabella per l'esempio 2. Il valore di Top per la misura First value basata su Product (AA per Astrida).

Customer	Product	Month	Sum(Sales)	First value
			2566	-
Astrida	AA	Jan	46	46
Astrida	BB	Jan	46	46
Astrida	AA	Feb	60	60
Astrida	BB	Feb	60	60
Astrida	AA	Mar	70	70
Astrida	BB	Mar	70	70
Astrida	AA	Apr	13	13
Astrida	BB	Apr	13	13

Per ulteriori informazioni, fare riferimento all'esempio 2 nella sezione relativa alla funzione **Above**.

Esempio 3

Esempio: 3	Risultato
La funzione **Top** può essere utilizzata come input per le funzioni di scala. Ad esempio: `RangeAvg (Top(Sum(Sales),1,3))`.	Negli argomenti per la funzione **Top()**, offset viene impostato su 1 e count viene impostato su 3. La funzione trova i risultati dell'espressione **Sum (Sales)** nelle tre righe che iniziano con la riga immediatamente sotto la riga inferiore nel segmento colonna (perché offset=1) e le due righe immediatamente sotto a essa (dove si trova una riga). Questi tre valori vengono utilizzati come input per la funzione RangeAvg(), che individua la media dei valori nella scala di numeri fornita. Una tabella in cui è stato impostato **Customer** come dimensione restituisce i seguenti risultati per l'espressione RangeAvg(). Astrida 603 Betacab 603 Canutility 603 Divadip: 603

```
Monthnames:
LOAD *, Dual(MonthText,MonthNumber) as Month INLINE [
MonthText, MonthNumber
Jan, 1
Feb, 2
Mar, 3
Apr, 4
May, 5
Jun, 6
Jul, 7
Aug, 8
Sep, 9
Oct, 10
Nov, 11
Dec, 12
];

Sales2013:
Crosstable (MonthText, Sales) LOAD * inline [
Customer|Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec
Astrida|46|60|70|13|78|20|45|65|78|12|78|22
Betacab|65|56|22|79|12|56|45|24|32|78|55|15
Canutility|77|68|34|91|24|68|57|36|44|90|67|27
Divadip|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');
```

Vedere anche:

p *Bottom - funzione per grafici (page 1276)*
p *Above - funzione per grafici (page 1267)*
p *Sum - funzione per grafici (page 351)*
p *RangeAvg (page 1324)*
p *Funzioni di scala (page 1321)*

SecondaryDimensionality - funzione per grafici

SecondaryDimensionality() restituisce il numero di righe di dimensione di una tabella pivot senza contenuti di aggregazione; ad esempio, senza somme parziali o aggregati compressi. Questa funzione è equivalente alla funzione **dimensionality()** per le dimensioni orizzontali delle tabelle pivot.

Sintassi:

```
SecondaryDimensionality( )
```

Tipo di dati restituiti: numero intero

Limiti:

- A meno che non venga utilizzata nelle tabelle pivot, la funzione **SecondaryDimensionality** restituisce sempre 0.
- L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

After - funzione per grafici

After() restituisce il valore di un'espressione valutata con i valori di dimensione di una tabella pivot così come appaiono nella colonna successiva a quella attuale all'interno di un segmento di riga nella tabella pivot.

Sintassi:

```
after([TOTAL] expr [, offset [, count ]])
```

L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Questa funzione restituisce NULL in tutti i tipi di grafico, ad eccezione delle tabelle pivot.

Argomenti:

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
offset	Specificando un **offset** n maggiore di 1, la valutazione dell'espressione viene spostata di n righe verso destra rispetto alla riga attuale. Specificando un offset uguale a 0 verrà valutata l'espressione nella riga attuale. Specificando un numero di offset negativo, la funzione **After** diventa equivalente alla funzione **Before** con il numero di offset positivo corrispondente.
count	Specificando un terzo parametro **count** maggiore di 1, la funzione restituirà una scala di valori, uno per ciascuna delle righe della tabella fino al valore di **count**, contando verso destra a partire dalla cella originaria.
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

In corrispondenza dell'ultima colonna di un segmento di riga verrà restituito un valore NULL, perché non vi sono colonne che la seguono.

Se una tabella pivot include più dimensioni orizzontali, il segmento della riga attuale includerà solo le colonne con gli stessi valori della colonna attuale in tutte le righe della dimensione, tranne per la riga che visualizza l'ultima dimensione orizzontale dell'ordinamento tra campi. La sequenza di ordinamento tra campi per le dimensioni orizzontali nelle tabelle pivot è definita semplicemente dall'ordine delle dimensioni dall'alto verso il basso.

Esempio:

```
after( sum( Sales ))
after( sum( Sales ), 2 )
after( total sum( Sales ))
```

`rangeavg (after(sum(x),1,3))` restituisce la media dei tre risultati della funzione **sum(x)** calcolata in base alle tre colonne immediatamente a destra di quella attuale.

Before - funzione per grafici

Before() restituisce il valore di un'espressione valutata con i valori di dimensione di una tabella pivot così come appaiono nella colonna precedente a quella attuale all'interno di un segmento di riga della tabella pivot.

Sintassi:

```
before([TOTAL] expr [, offset [, count]])
```

Questa funzione restituisce NULL in tutti i tipi di grafico, ad eccezione delle tabelle pivot.

L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
offset	Specificando un **offset** n maggiore di 1, la valutazione dell'espressione viene spostata di n righe verso sinistra rispetto alla riga attuale. Specificando un offset uguale a 0 verrà valutata l'espressione nella riga attuale. Specificando un numero di offset negativo, la funzione **Before** diventa equivalente alla funzione **After** con il numero di offset positivo corrispondente.
count	Specificando un terzo parametro **count** maggiore di 1, la funzione restituirà una scala di valori, uno per ciascuna delle righe della tabella fino al valore di **count**, contando verso sinistra a partire dalla cella originaria.
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

In corrispondenza della prima colonna di un segmento di riga verrà restituito un valore NULL, perché non vi sono colonne che la precedono.

Se una tabella pivot include più dimensioni orizzontali, il segmento della riga attuale includerà solo le colonne con gli stessi valori della colonna attuale in tutte le righe della dimensione, tranne per la riga che visualizza l'ultima dimensione orizzontale dell'ordinamento tra campi.La sequenza di ordinamento tra campi per le dimensioni orizzontali nelle tabelle pivot è definita semplicemente dall'ordine delle dimensioni dall'alto verso il basso.

Esempi:

```
before( sum( Sales ))
before( sum( Sales ), 2 )
before( total sum( Sales ))
```

`rangeavg (before(sum(x),1,3))` restituisce la media dei tre risultati della funzione **sum(x)** calcolata in base alle tre colonne immediatamente a sinistra di quella attuale.

First - funzione per grafici

First() restituisce il valore di un'espressione valutata con i valori di dimensione di una tabella pivot così come appaiono nella prima colonna del segmento di riga attuale della tabella pivot. Questa funzione restituisce NULL in tutti i tipi di grafico, ad eccezione delle tabelle pivot.

> *L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.*

Sintassi:

```
first([TOTAL] expr [, offset [, count]])
```

Argomenti:

Argomenti

Argomento	Descrizione
expression	L'espressione o il campo contenente i dati da misurare.
offset	Specificando un **offset** n maggiore di 1, la valutazione dell'espressione viene spostata di n righe verso destra rispetto alla riga attuale. Specificando un offset uguale a 0 verrà valutata l'espressione nella riga attuale. Specificando un numero di offset negativo, la funzione **First** diventa equivalente alla funzione **Last** con il numero di offset positivo corrispondente.
count	Specificando un terzo parametro **count** maggiore di 1, la funzione restituirà una scala di valori, uno per ciascuna delle righe della tabella fino al valore di **count**, contando verso destra a partire dalla cella originaria.
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Se una tabella pivot include più dimensioni orizzontali, il segmento della riga attuale includerà solo le colonne con gli stessi valori della colonna attuale in tutte le righe della dimensione, tranne per la riga che visualizza l'ultima dimensione orizzontale dell'ordinamento tra campi.La sequenza di ordinamento tra campi per le dimensioni orizzontali nelle tabelle pivot è definita semplicemente dall'ordine delle dimensioni dall'alto verso il basso.

Esempi:

```
first( sum( Sales ))
first( sum( Sales ), 2 )
first( total sum( Sales )
```

`rangeavg (first(sum(x),1,5))` restituisce una media dei risultati della funzione **sum(x)** calcolata in base alle cinque colonne più a sinistra del segmento di riga attuale.

Last - funzione per grafici

Last() restituisce il valore di un'espressione valutata con i valori di dimensione di una tabella pivot così come appaiono nell'ultima colonna del segmento di riga attuale della tabella pivot. Questa funzione restituisce NULL in tutti i tipi di grafico, ad eccezione delle tabelle pivot.

> *L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.*

Sintassi:

```
last([TOTAL] expr [, offset [, count]])
```

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
offset	Specificando un **offset** n maggiore di 1, la valutazione dell'espressione viene spostata di n righe verso sinistra rispetto alla riga attuale. Specificando un offset uguale a 0 verrà valutata l'espressione nella riga attuale. Specificando un numero di offset negativo, la funzione **First** diventa equivalente alla funzione **Last** con il numero di offset positivo corrispondente.
count	Specificando un terzo parametro **count** maggiore di 1, la funzione restituirà una scala di valori, uno per ciascuna delle righe della tabella fino al valore di **count**, contando verso sinistra a partire dalla cella originaria.
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Se una tabella pivot include più dimensioni orizzontali, il segmento della riga attuale includerà solo le colonne con gli stessi valori della colonna attuale in tutte le righe della dimensione, tranne per la riga che visualizza l'ultima dimensione orizzontale dell'ordinamento tra campi.La sequenza di ordinamento tra campi per le dimensioni orizzontali nelle tabelle pivot è definita semplicemente dall'ordine delle dimensioni dall'alto verso il basso.

Esempio:

```
last( sum( Sales ))
last( sum( Sales ), 2 )
```

`last( total sum( Sales )`
`rangeavg (last(sum(x),1,5))` restituisce una media dei risultati della funzione **sum(x)** calcolata in base alle cinque colonne più a destra del segmento di riga attuale.

ColumnNo - funzione per grafici

ColumnNo() restituisce il numero della colonna attuale all'interno del segmento di riga attuale in una tabella pivot. La prima colonna è la numero 1.

Sintassi:

```
ColumnNo([total])
```

Argomenti:

Argomenti

Argomento	Descrizione
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Se una tabella pivot include più dimensioni orizzontali, il segmento della riga attuale includerà solo le colonne con gli stessi valori della colonna attuale in tutte le righe della dimensione, tranne per la riga che visualizza l'ultima dimensione orizzontale dell'ordinamento tra campi.La sequenza di ordinamento tra campi per le dimensioni orizzontali nelle tabelle pivot è definita semplicemente dall'ordine delle dimensioni dall'alto verso il basso.

L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Esempio:

```
if( ColumnNo( )=1, 0, sum( Sales ) / before( sum( Sales )))
```

NoOfColumns - funzione per grafici

NoOfColumns() restituisce il numero di colonne nel segmento di riga attuale in una tabella pivot.

L'ordinamento sui valori y nei grafici o l'ordinamento per colonne di espressione nelle tabelle non è consentito quando si utilizza questa funzione di grafico in una qualsiasi delle espressioni del grafico. Queste alternative di ordinamento verranno perciò disattivate automaticamente. Quando si utilizza questa funzione di grafico in una visualizzazione o in una tabella, l'ordinamento della visualizzazione torna all'ordinamento inserito in questa funzione.

Sintassi:

```
NoOfColumns([total])
```

Argomenti:

Argomenti

Argomento	Descrizione
TOTAL	Se la tabella è unidimensionale o se è utilizzato il qualificatore **TOTAL** come argomento, il segmento colonna attuale sarà sempre uguale all'intera colonna.

Se la tabella pivot ha più dimensioni orizzontali, il segmento di riga attuale includerà solo le colonne con lo stesso valore della colonna attuale in tutte le righe di dimensione, eccetto la riga che mostra l'ultima dimensione nell'ordinamento tra campi.La sequenza di ordinamento tra campi per le dimensioni orizzontali nelle tabelle pivot è definita semplicemente dall'ordine delle dimensioni dall'alto verso il basso.

Esempio:

```
if( ColumnNo( )=NoOfColumns( ), 0, after( sum( Sales )))
```

5.17 Funzioni logiche

In questa sezione vengono descritte le funzioni di gestione delle operazioni logiche. Tutte le funzioni possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche.

IsNum

Restituisce -1 (True) se l'espressione può essere interpretata come numero, altrimenti restituisce 0 (False).

```
IsNum( expr )
```

IsText

Restituisce -1 (True) se l'espressione presenta una rappresentazione di testo, altrimenti restituisce 0 (False).

```
IsText( expr )
```

*Entrambe **IsNum** e **IsText** restituiscono 0 se l'espressione è NULL.*

Esempio:

Nel seguente esempio viene caricata una tabella inline contenente testo e valori numerici misti e vengono aggiunti due campi per verificare se il valore è, rispettivamente, un valore numerico o un valore di testo.

```
Load *, IsNum(Value), IsText(Value)
Inline [
```

```
Value
23
Green
Blue
12
33Red];
```

La tabella risultante avrà l'aspetto seguente:

Resulting table

Value	IsNum(Value)	IsText(Value)
23	-1	0
Green	0	-1
Blue	0	-1
12	-1	0
33Red	0	-1

5.18 Funzioni di mapping

In questa sezione vengono descritte le funzioni di gestione delle tabelle di mapping. È possibile utilizzare una tabella di mapping per sostituire valori o nomi di campo durante l'esecuzione dello script.

È possibile utilizzare le funzioni di mapping solo nello script di caricamento dei dati.

Panoramica sulle funzioni di mapping

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

ApplyMap

La funzione di script **ApplyMap** viene utilizzata per eseguire il mapping dell'output di un'espressione a una tabella di mapping caricata in precedenza.

```
ApplyMap ('mapname', expr [ , defaultexpr ] )
```

MapSubstring

La funzione di script **MapSubstring** consente di eseguire il mapping di parti di un'espressione a una tabella di mapping caricata in precedenza. Il mapping rispetta la distinzione tra maiuscole e minuscole e non è iterativo, mentre il mapping delle sottostringhe viene eseguito da sinistra a destra.

```
MapSubstring ('mapname', expr)
```

ApplyMap

La funzione di script **ApplyMap** viene utilizzata per eseguire il mapping dell'output di un'espressione a una tabella di mapping caricata in precedenza.

Sintassi:

```
ApplyMap('map_name', expression [ , default_mapping ] )
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
map_name	Il nome di una tabella di mapping che è stata creata in precedenza mediante l'istruzione **mapping load** o **mapping select**. Il nome deve essere incluso in virgolette singole diritte.  *Se si utilizza questa funzione in una variabile con espansione macro e si fa riferimento a una tabella di mapping non esistente, la chiamata alla funzione ha esito negativo e non viene creato un campo.*
expression	L'espressione, il cui risultato verrà mappato.
default_ mapping	Se definito, questo valore verrà utilizzato come un valore predefinito nel caso in cui la tabella di mapping non contenga un valore corrispondente per expression. Se non definito, il valore di expression verrà restituito così com'è.

 *Il campo di output di ApplyMap non deve avere lo stesso nome di uno dei suoi campi di input. In caso contrario, possono essere generati risultati imprevisti. Esempio da non usare:* *`ApplyMap('Map', A) as A.`*

Esempio:

In questo esempio viene caricato un elenco del personale addetto alle vendite con un codice paese che ne identifica il paese di residenza. Viene utilizzata una tabella per il mapping di un codice paese a un paese al fine di per sostituire il codice paese con il relativo nome. Nella tabella di mapping vengono definiti solo tre paesi, mentre gli altri codici paese vengono mappati a 'Rest of the world'.

```
// Load mapping table of country codes:
map1:
mapping LOAD *
Inline [
CCode, Country
Sw, Sweden
Dk, Denmark
No, Norway
] ;

// Load list of salesmen, mapping country code to country
// If the country code is not in the mapping table, put Rest of the world
```

```
Salespersons:
LOAD *,
ApplyMap('map1', CCode,'Rest of the world') As Country
Inline [
CCode, Salesperson
Sw, John
Sw, Mary
Sw, Per
Dk, Preben
Dk, Olle
No, Ole
Sf, Risttu
] ;

// We don't need the CCode anymore
Drop Field 'CCode';
```

La tabella risultante (Salespersons) avrà l'aspetto seguente:

Resulting table

Salesperson	Country
John	Sweden
Mary	Sweden
Per	Sweden
Preben	Denmark
Olle	Denmark
Ole	Norway
Risttu	Rest of the world

MapSubstring

La funzione di script **MapSubstring** consente di eseguire il mapping di parti di un'espressione a una tabella di mapping caricata in precedenza. Il mapping rispetta la distinzione tra maiuscole e minuscole e non è iterativo, mentre il mapping delle sottostringhe viene eseguito da sinistra a destra.

Sintassi:

```
MapSubstring('map_name', expression)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
map_name	Il nome di un tabella di mapping letta in precedenza da un'istruzione **mapping load** o **mapping select**. Il nome deve essere racchiuso tra virgolette singole diritte.  *Se si utilizza questa funzione in una variabile con espansione macro e si fa riferimento a una tabella di mapping non esistente, la chiamata alla funzione ha esito negativo e non viene creato un campo.*
expression	L'espressione il cui risultato deve essere mappato dalle sottostringhe.

Esempio:

In questo esempio verrà caricato un elenco di modelli del prodotto. Ogni modello presenta una serie di attributi che sono descritti da un codice composto. Utilizzando la tabella di mapping con MapSubstring, è possibile espandere i codici degli attributi per ottenere una descrizione.

```
map2:
mapping LOAD *
Inline [
AttCode, Attribute
R, Red
Y, Yellow
B, Blue
C, Cotton
P, Polyester
S, Small
M, Medium
L, Large
] ;

Productmodels:
LOAD *,
MapSubString('map2', AttCode) as Description
Inline [
Model, AttCode
Twixie, R C S
Boomer, B P L
Raven, Y P M
Seedling, R C L
SeedlingPlus, R C L with hood
Younger, B C with patch
MultiStripe, R Y B C S/M/L
] ;
```

```
// We don't need the AttCode anymore
Drop Field 'AttCode';
```

La tabella risultante avrà l'aspetto seguente:

Resulting table

Model	Description
Twixie	Red Cotton Small
Boomer	Blue Polyester Large
Raven	Yellow Polyester Medium
Seedling	Red Cotton Large
SeedlingPlus	Red Cotton Large with hood
Younger	Blue Cotton with patch
MultiStripe	Red Yellow Blue Cotton Small/Medium/Large

5.19 Funzioni matematiche

In questa sezione vengono descritte le funzioni per le costanti matematiche e i valori booleani. Queste funzioni non presentano parametri, tuttavia le parentesi sono obbligatorie.

Tutte le funzioni possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche.

e

La funzione restituisce la base degli algoritmi naturali, **e** (2.71828...).

```
e( )
```

false

La funzione restituisce un valore duale con il valore di testo 'False' e il valore numerico 0, che può essere utilizzato come valore logico false nelle espressioni.

```
false( )
```

pi

La funzione restituisce il valore di π (3.14159...).

```
pi( )
```

rand

La funzione restituisce un numero casuale tra 0 e 1. Può essere utilizzata per creare dati campione.

```
rand( )
```

Esempio:

Questo script di esempio crea una tabella di 1000 record con caratteri maiuscoli selezionati casualmente, vale a dire caratteri compresi nell'intervallo tra 65 e 91 (65+26).

```
Load
     Chr( Floor(rand() * 26) + 65) as UCaseChar,
     RecNo() as ID
     Autogenerate 1000;
```

true
La funzione restituisce un valore duale con il valore di testo 'True' e il valore numerico -1, che può essere utilizzato come valore logico true nelle espressioni.

```
true( )
```

5.20 Funzioni NULL

In questa sezione vengono descritte le funzioni per la restituzione o il rilevamento dei valori NULL.

Tutte le funzioni possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche.

Panoramica sulle funzioni di NULL

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

EmptyIsNull
La funzione **EmptyIsNull** converte stringhe vuote in NULL. Pertanto, restituisce NULL se il parametro è una stringa vuota, altrimenti restituisce il parametro.

```
EmptyIsNull (expr )
```

IsNull
La funzione **IsNull** verifica se il valore di un'espressione sia NULL restituendo -1 (True) in caso affermativo e 0 (False) in caso contrario.

```
IsNull (expr )
```

Null
La funzione **Null** restituisce un valore NULL.

```
NULL( )
```

EmptyIsNull

La funzione **EmptyIsNull** converte stringhe vuote in NULL. Pertanto, restituisce NULL se il parametro è una stringa vuota, altrimenti restituisce il parametro.

Sintassi:

```
EmptyIsNull(exp )
```

Esempi e risultati:

Esempi di script

Esempio	Risultato
`EmptyIsNull(AdditionalComments)`	Questa espressione restituirà come null qualsiasi valore di stringa vuota del campo *AdditionalComments*, al posto di stringhe vuote. Vengono restituite le stringhe e i numeri non vuoti.
`EmptyIsNull(PurgeChar (PhoneNumber, ' -()'))`	Questa espressione rimuoverà qualsiasi trattino, spazio e parentesi dal campo *PhoneNumber*. Se non sono rimasti caratteri, la funzione EmptyIsNull restituisce la stringa vuota come null; un numero di telefono vuoto corrisponde a un numero di telefono assente.

IsNull

La funzione **IsNull** verifica se il valore di un'espressione sia NULL restituendo -1 (True) in caso affermativo e 0 (False) in caso contrario.

Sintassi:

```
IsNull(expr )
```

*Una stringa con lunghezza zero non è considerata NULL e farà sì che **IsNull** restituisca False.*

Esempio: Script di caricamento dei dati

In questo esempio viene caricata una tabella inline con quattro righe, in cui le prime tre righe non contengono alcun valore oppure contengono - o 'NULL' nella colonna Value. Questi valori verranno convertiti in rappresentazioni di valori NULL true con l'istruzione centrale preceding **LOAD** mediante la funzione **Null**.

La prima istruzione preceding **LOAD** aggiunge un campo che consente di verificare se il valore è NULL mediante la funzione **IsNull**.

```
NullsDetectedAndConverted:

LOAD *,
If(IsNull(ValueNullConv), 'T', 'F') as IsItNull;

LOAD *,
If(len(trim(Value))= 0 or Value='NULL' or Value='-', Null(), Value ) as ValueNullConv;

LOAD * Inline
[ID, Value
```

```
0,
1,NULL
2,-
3,Value];
```

Questa è la tabella risultante. Nella colonna ValueNullConv i valori NULL vengono rappresentati da -.

Resulting table

ID	Value	ValueNullConv	IsItNull
0		-	T
1	NULL	-	T
2	-	-	T
3	Value	Value	F

NULL

La funzione **Null** restituisce un valore NULL.

Sintassi:

```
Null( )
```

Esempio: Script di caricamento dei dati

In questo esempio viene caricata una tabella inline con quattro righe, in cui le prime tre righe non contengono alcun valore oppure contengono - o 'NULL' nella colonna Value. Si desidera convertire questi valori in rappresentazioni di valore NULL true.

L'istruzione centrale preceding **LOAD** esegue la conversione utilizzando la funzione **Null**.

La prima istruzione preceding **LOAD** aggiunge un campo che verifica se il valore corrisponde a NULL, solo a scopo illustrativo in questo esempio.

```
NullsDetectedAndConverted:

LOAD *,
If(IsNull(ValueNullConv), 'T', 'F') as IsItNull;

LOAD *,
If(len(trim(Value))= 0 or Value='NULL' or Value='-', Null(), Value ) as ValueNullConv;

LOAD * Inline
[ID, Value
0,
1,NULL
2,-
3,Value];
```

Questa è la tabella risultante. Nella colonna ValueNullConv i valori NULL vengono rappresentati da -.

Resulting table

ID	Value	ValueNullConv	IsItNull
0		-	T
1	NULL	-	T
2	-	-	T
3	Value	Value	F

5.21 Funzioni di scala

Le funzioni scala sono funzioni che elaborano una matrice di valori e restituiscono un valore unico. Tutte le funzioni scala possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche.

Ad esempio, in una visualizzazione, una funzione scala è in grado di calcolare un valore singolo da una matrice intra-record. Nello script di caricamento dei dati, una funzione scala è in grado di calcolare un valore singolo dalla matrice di valori in una tabella interna.

*Le funzioni scala sostituiscono le seguenti funzioni numeriche generiche: **numsum**,**numavg**, **numcount**,**nummin** e **nummax**, che devono essere ritenute obsolete.*

Funzioni di scala di base

RangeMax
RangeMax() restituisce i valori numerici più alti trovati nel campo o nell'espressione.

```
RangeMax (first_expr[, Expression])
```

RangeMaxString
RangeMaxString() restituisce l'ultimo valore nell'ordine del testo che trova nel campo o nell'espressione.

```
RangeMaxString (first_expr[, Expression])
```

RangeMin
RangeMin() restituisce i valori numerici più bassi trovati nel campo o nell'espressione.

```
RangeMin (first_expr[, Expression])
```

RangeMinString
RangeMinString() restituisce il primo valore nell'ordine del testo che trova nel campo o nell'espressione.

```
RangeMinString (first_expr[, Expression])
```

RangeMode
RangeMode() trova il valore più comune (valore della modalità) nel campo o nell'espressione.

```
RangeMode (first_expr[, Expression])
```

RangeOnly

RangeOnly() è una funzione duale che restituisce un valore se l'espressione viene valutata in base a un valore univoco. In caso contrario, viene restituito **NULL**.

```
RangeOnly (first_expr[, Expression])
```

RangeSum

RangeSum() restituisce la somma di una scala di valori. Tutti i valori non numerici vengono trattati come 0.

```
RangeSum (first_expr[, Expression])
```

Funzioni di scala di conteggio

RangeCount

RangeCount() restituisce il numero di valori, sia testuali che numerici, nell'espressione o nel campo.

```
RangeCount (first_expr[, Expression])
```

RangeMissingCount

RangeMissingCount() restituisce il numero di valori non numerici (incluso NULL) nell'espressione o nel campo.

```
RangeMissingCount (first_expr[, Expression])
```

RangeNullCount

RangeNullCount() trova il numero di valori NULL nel campo o nell'espressione.

```
RangeNullCount (first_expr[, Expression])
```

RangeNumericCount

RangeNumericCount() trova il numero di valori numerici nel campo o nell'espressione.

```
RangeNumericCount (first_expr[, Expression])
```

RangeTextCount

RangeTextCount() restituisce il numero di valori di testo nel campo o nell'espressione.

```
RangeTextCount (first_expr[, Expression])
```

Funzioni di scala statistiche

RangeAvg

RangeAvg() restituisce la media di una scala. L'input per la funzione può essere una scala di valori o un'espressione.

```
RangeAvg (first_expr[, Expression])
```

RangeCorrel

RangeCorrel() restituisce il coefficiente di correlazione per due serie di dati. Il coefficiente di correlazione è una misura della relazione tra le serie di dati.

```
RangeCorrel (x_values , y_values[, Expression])
```

RangeFractile

RangeFractile() restituisce il valore che corrisponde al **fractile** n-esimo (quantile) di un intervallo di numeri.

```
RangeFractile (fractile, first_expr[ ,Expression])
```

RangeKurtosis

RangeKurtosis() restituisce il valore che corrisponde al kurtosis di una scala di numeri.

```
RangeKurtosis (first_expr[, Expression])
```

RangeSkew

RangeSkew() restituisce il valore che corrisponde all'asimmetria di una scala di numeri.

```
RangeSkew (first_expr[, Expression])
```

RangeStdev

RangeStdev() trova la deviazione standard di una scala di numeri.

```
RangeStdev (expr1[, Expression])
```

Funzioni di scala finanziarie

RangeIRR

RangeIRR() restituisce il tasso di rendimento interno per una serie di flussi di cassa rappresentati dai valori di input.

```
RangeIRR (value[, value][, Expression])
```

RangeNPV

RangeNPV() restituisce il valore attuale netto di un investimento basato su un tasso di sconto e una serie di futuri pagamenti periodici (valori negativi) ed entrate (valori positivi). Il risultato presenta un formato numerico predefinito di **money**.

```
RangeNPV (discount_rate, value[, value][, Expression])
```

RangeXIRR

RangeXIRR() restituisce il tasso interno di ritorno per una programmazione di flussi di denaro che non è necessariamente su base periodica. Per calcolare il tasso di rendimento interno di una serie di flussi di cassa periodici, utilizzare la funzione **RangeIRR**.

```
RangeXIRR (values, dates[, Expression])
```

RangeXNPV

RangeXNPV() restituisce il valore attuale netto per una programmazione di flussi di cassa non necessariamente periodica. Il risultato è in un formato numerico predefinito della valuta. Per calcolare il valore attuale netto per una serie di flussi di cassa periodici, utilizzare la funzione **RangeNPV**.

```
RangeXNPV (discount_rate, values, dates[, Expression])
```

Vedere anche:

p *Funzioni intra-record (page 1264)*

RangeAvg

RangeAvg() restituisce la media di una scala. L'input per la funzione può essere una scala di valori o un'espressione.

Sintassi:

```
RangeAvg(first_expr[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

Se non viene trovato nessun valore numerico, viene restituito NULL.

Esempi e risultati:

Esempi di script

Esempi	Risultati
`RangeAvg (1,2,4)`	Restituisce 2,33333333
`RangeAvg (1,'xyz')`	Restituisce 1
`RangeAvg (null( ), 'abc')`	Restituisce NULL

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
RangeTab3:
LOAD recno() as RangeID, RangeAvg(Field1,Field2,Field3) as MyRangeAvg INLINE [
Field1, Field2, Field3
10,5,6
2,3,7
```

```
8,2,8
18,11,9
5,5,9
9,4,2
];
```

La tabella risultante mostra i valori restituiti in MyRangeAvg per ciascun record della tabella.

Tabella risultante

RangeID	MyRangeAvg
1	7
2	4
3	6
4	12.666
5	6.333
6	5

Esempio con espressione:

```
RangeAvg (Above(MyField),0,3))
```

Restituisce la media mobile del risultato della scala di tre valori di **MyField** calcolati in base alla riga attuale e alle due righe che la precedono. Specificando 3 per il terzo argomento, la funzione **Above()** restituisce tre valori, se il numero di righe che precedono la riga attuale è sufficiente, che vengono considerati come input per la funzione **RangeAvg()**.

Dati utilizzati negli esempi:

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati campione

MyField	RangeAvg (Above (MyField,0,3))	Comments
10	10	Dato che questa è la prima riga, la scala presenta un unico valore.
2	6	Questa riga è preceduta da un'unica riga, quindi la scala è: 10,2.
8	6.6666666667	L'equivalente di RangeAvg(10,2,8)
18	9.333333333	-
5	10. 333333333	-
9	10.6666666667	-

```
RangeTab:
LOAD * INLINE [
```

```
MyField
10
2
8
18
5
9
] ;
```

Vedere anche:

p *Avg - funzione per grafici (page 400)*
p *Count - funzione per grafici (page 356)*

RangeCorrel

RangeCorrel() restituisce il coefficiente di correlazione per due serie di dati. Il coefficiente di correlazione è una misura della relazione tra le serie di dati.

Sintassi:

```
RangeCorrel(x_value , y_value[, Expression])
```

Tipo di dati restituiti: numerico

Le serie di dati devono essere immesse come coppie (x,y). Ad esempio, per valutare due serie di dati, l'insieme 1 e l'insieme 2, in cui l'insieme 1 = 2,6,9 e l'insieme 2 = 3,8,4, occorrerà scrivere `RangeCorrel (2,3,6,8,9,4)`, che restituisce 0,269.

Argomenti:

Argomenti

Argomento	Descrizione
x-value, y-value	Ogni valore rappresenta un valore singolo o una scala di valori restituiti da una funzione intra-record con un terzo parametro opzionale. Ogni valore o scala di valori deve corrispondere a un **x-value** oppure a una scala di **y-values**.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

Per essere calcolata, la funzione necessita almeno di due coppie di coordinate.

I valori di testo, i valori NULL e i valori mancanti restituiscono NULL.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeCorrel (2,3,6,8,9,4,8,5)`	Restituisce 0,2492. Questa funzione può essere caricata nello script o aggiunta a una visualizzazione nell'editor delle espressioni.

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
RangeList:
Load * Inline [
ID1|x1|y1|x2|y2|x3|y3|x4|y4|x5|y5|x6|y6
01|46|60|70|13|78|20|45|65|78|12|78|22
02|65|56|22|79|12|56|45|24|32|78|55|15
03|77|68|34|91|24|68|57|36|44|90|67|27
04|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');

XY:
LOAD recno() as RangeID, * Inline [
X|Y
2|3
6|8
9|4
8|5
](delimiter is '|');
```

In una tabella ID1 come dimensione e la misura: RangeCorrel(x1,y1,x2,y2,x3,y3,x4,y4,x5,y5,x6,y6)), la funzione **RangeCorrel()** trova il valore di **Correl** nella scala di sei coppie di x,y, per ognuno dei valori ID1.

Tabella risultante

ID1	MyRangeCorrel
01	-0.9517
02	-0.5209
03	-0.5209
04	-0.1599

Esempio:

```
XY:
LOAD recno() as RangeID, * Inline [
X|Y
2|3
```

```
6|8
9|4
8|5
](delimiter is '|');
```

In una tabella RangeID come dimensione e la misura: RangeCorrel(Below(X,0,4,BelowY,0,4)), la funzione **RangeCorrel()** utilizza i risultati delle funzioni **Below()**, che, in virtù del terzo argomento (count) impostato su 4, producono un intervallo di quattro valori x-y dalla tabella XY caricata.

Tabella risultante

RangeID	MyRangeCorrel2
01	0.2492
02	-0.9959
03	-1.0000
04	-

Il valore di RangeID 01 è uguale a inserire manualmente RangeCorrel(2,3,6,8,9,4,8,5). Per gli altri valori di RangeID, le serie prodotte dalla funzione Below() sono: (6,8,9,4,8,5), (9,4,8,5) e (8,5), l'ultima delle quali produce un risultato null.

Vedere anche:

p *Correl - funzione per grafici (page 403)*

RangeCount

RangeCount() restituisce il numero di valori, sia testuali che numerici, nell'espressione o nel campo.

Sintassi:

```
RangeCount(first_expr[, Expression])
```

Tipo di dati restituiti: numero intero

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da conteggiare.
Expression	Le espressioni o i campi opzionali contenenti la scala di dati da conteggiare.

Limiti:

I valori NULL non vengono considerati.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeCount (1,2,4)`	Restituisce 3
`RangeCount (2,'xyz')`	Restituisce 2
`RangeCount (null( ))`	Restituisce 0
`RangeCount (2,'xyz', null())`	Restituisce 2

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
RangeTab3:
LOAD recno() as RangeID, RangeCount(Field1,Field2,Field3) as MyRangeCount INLINE [
Field1, Field2, Field3
10,5,6
2,3,7
8,2,8
18,11,9
5,5,9
9,4,2
];
```

La tabella risultante mostra i valori restituiti in MyRangeCount per ciascun record della tabella.

Tabella dei risultati

RangeID	MyRangeCount
1	3
2	3
3	3
4	3
5	3
6	3

Esempio con espressione:

```
RangeCount (Above(MyField,1,3))
```

Restituisce il numero di valori contenuto nei tre risultati di **MyField**. Specificando il primo argomento della funzione **Above()** come 1 e il secondo argomento come 3, restituisce i valori dei primi tre campi sopra la riga corrente, in presenza di un numero di righe sufficiente prese come input della funzione **RangeCount()**.

Dati utilizzati negli esempi:

Dati campione

MyField	RangeCount(Above(MyField,1,3))
10	0
2	1
8	2
18	3
5	3
9	3

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
2
8
18
5
9
] ;
```

Vedere anche:

p *Count - funzione per grafici (page 356)*

RangeFractile

RangeFractile() restituisce il valore che corrisponde al **fractile** n-esimo (quantile) di un intervallo di numeri.

Durante il calcolo del frattale, RangeFractile() utilizza l'interpolazione lineare tra le classificazioni più prossime.

Sintassi:

```
RangeFractile(fractile, first_expr[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
fractile	Un numero compreso tra 0 e 1 corrispondente al frattale (quantile espresso come frazione) da calcolare.
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeFractile (0.24,1,2,4,6)`	Restituisce 1,72
`RangeFractile(0.5,1,2,3,4,6)`	Restituisce 3
`RangeFractile (0.5,1,2,5,6)`	Restituisce 3,5

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
RangeTab:
LOAD recno() as RangeID, RangeFractile(0.5,Field1,Field2,Field3) as MyRangeFrac INLINE [
Field1, Field2, Field3
10,5,6
2,3,7
8,2,8
18,11,9
5,5,9
9,4,2
];
```

La tabella risultante mostra i valori restituiti in MyRangeFrac per ciascun record della tabella.

Tabella risultante

RangeID	MyRangeFrac
1	6
2	3
3	8
4	11
5	5
6	4

Esempio con espressione:

```
RangeFractile (0.5, Above(Sum(MyField),0,3))
```

In questo esempio, la funzione intra-record **Above()** contiene gli argomenti offset e count opzionali. Ciò restituisce una scala di risultati che può essere utilizzata come input per una qualsiasi delle funzioni scala. In questo caso, `Above(Sum(MyField),0,3)` restituisce il valore di `MyField` per la riga attuale e le due righe che la precedono. Questi valori forniscono l'input per la funzione **RangeFractile()**. Per la riga inferiore della tabella seguente, ciò equivale a `RangeFractile(0.5, 3,4,6)`, vale a dire il calcolo del frattale 0,5 per le serie 3, 4 e 6. Per le prime due righe della tabella seguente, il numero dei valori nella scala viene ridotto di conseguenza, se non vi sono righe che precedono la riga attuale. Vengono restituiti risultati simili per le altre funzioni intra-record.

Dati campione

MyField	RangeFractile(0.5, Above(Sum(MyField),0,3))
1	1
2	1.5
3	2
4	3
5	4
6	5

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
1
2
3
4
5
6
] ;
```

Vedere anche:

p *Above - funzione per grafici (page 1267)*
p *Fractile - funzione per grafici (page 407)*

RangeIRR

RangeIRR() restituisce il tasso di rendimento interno per una serie di flussi di cassa rappresentati dai valori di input.

Il tasso di rendimento interno è il tasso di interesse ricevuto per un investimento che consiste in pagamenti (valori negativi) ed entrate (valori positivi) che ricorrono ad intervalli regolari.

Sintassi:

```
RangeIRR(value[, value][, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	Un valore singolo oppure una scala di valori restituita da una funzione intra-record con un terzo parametro opzionale. Per essere calcolata, la funzione necessita almeno di un valore positivo e di un valore negativo.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Tabella di esempio

Esempi	Risultati
`RangeIRR(-70000,12000,15000,18000,21000,26000)`	Restituisce 0,0866

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.
`RangeTab3:`
`LOAD *,`
`recno() as RangeID,`
`RangeIRR(Field1,Field2,Field3) as RangeIRR;`
`LOAD * INLINE [`
`Field1|Field2|Field3`
`-10000|5000|6000`
`-2000|NULL|7000`
`-8000|'abc'|8000`
`-1800|11000|9000`
`-5000|5000|9000`
`-9000|4000|2000`
`] (delimiter is '|');`
	La tabella risultante mostra i valori restituiti in RangeIRR per ciascun record della tabella.
	RangeID — RangeIRR
	1 — 0.0639
	2 — 0.8708
	3 — -
	4 — 5.8419
	5 — 0.9318
	6 — -0.2566

Vedere anche:

p *Funzioni intra-record (page 1264)*

RangeKurtosis

RangeKurtosis() restituisce il valore che corrisponde al kurtosis di una scala di numeri.

Sintassi:

```
RangeKurtosis(first_expr[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

Se non viene trovato nessun valore numerico, viene restituito NULL.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeKurtosis (1,2,4,7)`	Restituisce -0,28571428571429

Vedere anche:

p *Kurtosis - funzione per grafici (page 415)*

RangeMax

RangeMax() restituisce i valori numerici più alti trovati nel campo o nell'espressione.

Sintassi:

```
RangeMax(first_expr[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

Se non viene trovato nessun valore numerico, viene restituito NULL.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeMax (1,2,4)`	Restituisce 4
`RangeMax (1,'xyz')`	Restituisce 1
`RangeMax (null( ), 'abc')`	Restituisce NULL

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
RangeTab3:
LOAD recno() as RangeID, RangeMax(Field1,Field2,Field3) as MyRangeMax INLINE [
Field1, Field2, Field3
10,5,6
2,3,7
8,2,8
18,11,9
5,5,9
9,4,2
];
```

La tabella risultante mostra i valori restituiti in MyRangeMax per ciascun record della tabella.

Tabella risultante

RangeID	MyRangeMax
1	10
2	7
3	8
4	18
5	9
6	9

Esempio con espressione:

```
RangeMax (Above(MyField,0,3))
```

Restituisce il valore massimo nella scala di tre valori di **MyField** calcolati in base alla riga attuale e alle due righe che la precedono. Specificando 3 per il terzo argomento, la funzione **Above()** restituisce tre valori, se il numero di righe che precedono la riga attuale è sufficiente, che vengono considerati come input per la funzione **RangeMax()**.

Dati utilizzati negli esempi:

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati campione

MyField	RangeMax (Above(Sum(MyField),1,3))
10	10
2	10
8	10
18	18
5	18
9	18

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
2
8
18
5
9
] ;
```

RangeMaxString

RangeMaxString() restituisce l'ultimo valore nell'ordine del testo che trova nel campo o nell'espressione.

Sintassi:

```
RangeMaxString(first_expr[, Expression])
```

Tipo di dati restituiti: stringa

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeMaxString (1,2,4)`	Restituisce 4
`RangeMaxString ('xyz','abc')`	Restituisce 'xyz'
`RangeMaxString (5,'abc')`	Restituisce 'abc'
`RangeMaxString (null( ))`	Restituisce NULL

Esempio con espressione:

```
RangeMaxString (Above(MaxString(MyField),0,3))
```

Restituisce l'ultimo (nell'ordine alfabetico del testo) dei tre risultati della funzione **MaxString(MyField)** valutati in base alla riga attuale e alle due righe che la precedono.

Dati utilizzati negli esempi:

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati campione

MyField	RangeMaxString(Above(MaxString(MyField),0,3))
10	10
abc	abc
8	abc
def	def
xyz	xyz
9	xyz

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
'abc'
8
'def'
```

```
'xyz'
9
] ;
```

Vedere anche:

p *MaxString - funzione per grafici (page 534)*

RangeMin

RangeMin() restituisce i valori numerici più bassi trovati nel campo o nell'espressione.

Sintassi:

```
RangeMin(first_expr[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

Se non viene trovato nessun valore numerico, viene restituito NULL.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeMin (1,2,4)`	Restituisce 1
`RangeMin (1,'xyz')`	Restituisce 1
`RangeMin (null( ), 'abc')`	Restituisce NULL

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
RangeTab3:
LOAD recno() as RangeID, RangeMin(Field1,Field2,Field3) as MyRangeMin INLINE [
Field1, Field2, Field3
10,5,6
2,3,7
8,2,8
18,11,9
```

```
5,5,9
9,4,2
];
```

La tabella risultante mostra i valori restituiti in MyRangeMin per ciascun record della tabella.

Tabella risultante

RangeID	MyRangeMin
1	5
2	2
3	2
4	9
5	5
6	2

Esempio con espressione:

```
RangeMin (Above(MyField,0,3)
```

Restituisce il valore minimo nella scala di tre valori di **MyField** calcolati in base alla riga attuale e alle due righe che la precedono. Specificando 3 per il terzo argomento, la funzione **Above()** restituisce tre valori, se il numero di righe che precedono la riga attuale è sufficiente, che vengono considerati come input per la funzione **RangeMin()**.

Dati utilizzati negli esempi:

Dati campione

MyField	RangeMin(Above(MyField,0,3))
10	10
2	2
8	2
18	2
5	5
9	5

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
2
8
18
5
9
```

```
] ;
```

Vedere anche:

p *Min - funzione per grafici (page 342)*

RangeMinString

RangeMinString() restituisce il primo valore nell'ordine del testo che trova nel campo o nell'espressione.

Sintassi:

```
RangeMinString(first_expr[, Expression])
```

Tipo di dati restituiti: stringa

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeMinString (1,2,4)`	Restituisce 1
`RangeMinString ('xyz','abc')`	Restituisce 'abc'
`RangeMinString (5,'abc')`	Restituisce 5
`RangeMinString (null( ))`	Restituisce NULL

Esempio con espressione:

```
RangeMinString (Above(MinString(MyField),0,3))
```

Restituisce il primo (nell'ordine alfabetico del testo) dei tre risultati della funzione **MinString(MyField)** valutati in base alla riga attuale e alle due righe che la precedono.

Dati utilizzati negli esempi:

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati campione

MyField	RangeMinString(Above(MinString(MyField),0,3))
10	10
abc	10
8	8
def	8
xyz	8
9	9

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
'abc'
8
'def'
'xyz'
9
] ;
```

Vedere anche:

p *MinString - funzione per grafici (page 536)*

RangeMissingCount

RangeMissingCount() restituisce il numero di valori non numerici (incluso NULL) nell'espressione o nel campo.

Sintassi:

```
RangeMissingCount(first_expr[, Expression])
```

Tipo di dati restituiti: numero intero

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da conteggiare.
Expression	Le espressioni o i campi opzionali contenenti la scala di dati da conteggiare.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeMissingCount (1,2,4)`	Restituisce 0
`RangeMissingCount (5,'abc')`	Restituisce 1
`RangeMissingCount (null( ))`	Restituisce 1

Esempio con espressione:

```
RangeMissingCount (Above(MinString(MyField),0,3))
```

Restituisce il numero di valori non numerici trovati nei tre risultati della funzione **MinString(MyField)** valutati in base alla riga corrente e alle due righe che la precedono.

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati campione

MyField	RangeMissingCount (Above(MinString (MyField),0,3))	Explanation
10	2	Restituisce 2 perché non ci sono righe che precedono questa riga, quindi 2 dei 3 valori sono mancanti.
abc	2	Restituisce 2 perché la riga corrente è preceduta da 1 sola riga e la riga corrente non è numerica ('abc').
8	1	Restituisce 1 perché 1 delle 3 righe include un valore non numerico ('abc').
def	2	Restituisce 2 perché 2 delle 3 righe includono valori non numerici ('def' e 'abc').
xyz	2	Restituisce 2 perché 2 delle 3 righe includono valori non numerici (' xyz' e 'def').
9	2	Restituisce 2 perché 2 delle 3 righe includono valori non numerici (' xyz' e 'def').

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
'abc'
8
```

```
'def'
'xyz'
9
] ;
```

Vedere anche:

p *MissingCount - funzione per grafici (page 359)*

RangeMode

RangeMode() trova il valore più comune (valore della modalità) nel campo o nell'espressione.

Sintassi:

```
RangeMode(first_expr {, Expression})
```

Tipo di dati restituiti: numerico

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

Se più di un valore condivide la frequenza più alta, viene restituito NULL.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeMode (1,2,9,2,4)`	Restituisce 2
`RangeMode ('a',4,'a',4)`	Restituisce NULL
`RangeMode (null( ))`	Restituisce NULL

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
RangeTab3:
LOAD recno() as RangeID, RangeMode(Field1,Field2,Field3) as MyRangeMode INLINE [
```

```
Field1, Field2, Field3
10,5,6
2,3,7
8,2,8
18,11,9
5,5,9
9,4,2
];
```

La tabella risultante mostra i valori restituiti da **MyRangMode** per ciascun record della tabella.

Tabella dei risultati

RangeID	MyRangMode
1	-
2	-
3	8
4	-
5	5
6	-

Esempio con espressione:

```
RangeMode (Above(MyField,0,3))
```

Restituisce il valore più ricorrente all'interno dei tre risultati di **MyField** valutati in base alla riga attuale e alle due righe che la precedono. Specificando 3 per il terzo argomento, la funzione **Above()** restituisce tre valori, se il numero di righe che precedono la riga attuale è sufficiente, che vengono considerati come input per la funzione **RangeMode()**.

Dati utilizzati nell'esempio:

```
RangeTab:
LOAD * INLINE [
MyField
10
2
8
18
5
9
] ;
```

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati campione

MyField	RangeMode(Above(MyField,0,3))
10	Restituisce 10 perché non vi sono righe che precedono quella attuale, quindi il singolo valore è quello più ricorrente.
2	-
8	-
18	-
5	-
9	-

Vedere anche:

p *Mode - funzione per grafici (page 345)*

RangeNPV

RangeNPV() restituisce il valore attuale netto di un investimento basato su un tasso di sconto e una serie di futuri pagamenti periodici (valori negativi) ed entrate (valori positivi). Il risultato presenta un formato numerico predefinito di **money**.

Per i flussi di cassa che non sono necessariamente periodici, fare riferimento a *RangeXNPV (page 1357)*.

Sintassi:

```
RangeNPV(discount_rate, value[,value][, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
discount_rate	Il tasso di interesse per periodo.
value	Un pagamento o un'entrata ricorrente alla fine di ogni periodo. Ciascun valore può essere un valore singolo oppure una scala di valori restituita da una funzione intra-record con un terzo parametro opzionale.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Esempi	Risultati
`RangeNPV(0.1,-10000,3000,4200,6800)`	Restituisce 1188,44

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.
`RangeTab3:`
`LOAD *,`
`recno() as RangeID,`
`RangeNPV(Field1,Field2,Field3) as RangeNPV;`
`LOAD * INLINE [`
`Field1|Field2|Field3`
`10|5|-6000`
`2|NULL|7000`
`8|'abc'|8000`
`18|11|9000`
`5|5|9000`
`9|4|2000`
`] (delimiter is '|');`
	La tabella risultante mostra i valori restituiti in RangeNPV per ciascun record della tabella.
	RangeID — RangeNPV
	1 — $-49.13
	2 — $777.78
	3 — $98.77
	4 — $25.51
	5 — $250.83
	6 — $20.40

Vedere anche:

p *Funzioni intra-record (page 1264)*

RangeNullCount

RangeNullCount() trova il numero di valori NULL nel campo o nell'espressione.

Sintassi:

```
RangeNullCount(first_expr [, Expression])
```

Tipo di dati restituiti: numero intero

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeNullCount (1,2,4)`	Restituisce 0
`RangeNullCount (5,'abc')`	Restituisce 0
`RangeNullCount (null( ), null( ))`	Restituisce 2

Esempio con espressione:

```
RangeNullCount (Above(Sum(MyField),0,3))
```

Restituisce il numero di valori NULL trovati nei tre risultati della funzione **Sum(MyField)** valutati in base alla riga attuale e alle due righe che la precedono.

*Se si copia **MyField** nell'esempio seguente, non verrà restituito il valore NULL.*

Dati campione

MyField	RangeNullCount(Above(Sum(MyField),0,3))
10	Restituisce 2 perché non ci sono righe che precedono questa riga, quindi 2 dei 3 valori sono mancanti (=NULL).
'abc'	Restituisce 1 perché la riga attuale è preceduta solo da una riga, quindi uno dei tre valori è mancante (=NULL).
8	Restituisce 0 perché nessuna delle tre righe corrisponde a un valore NULL.

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
'abc'
8
] ;
```

Vedere anche:

p *NullCount - funzione per grafici (page 362)*

RangeNumericCount

RangeNumericCount() trova il numero di valori numerici nel campo o nell'espressione.

Sintassi:

```
RangeNumericCount(first_expr[, Expression])
```

Tipo di dati restituiti: numero intero

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeNumericCount (1,2,4)`	Restituisce 3
`RangeNumericCount (5,'abc')`	Restituisce 1
`RangeNumericCount (null( ))`	Restituisce 0

Esempio con espressione:

```
RangeNumericCount (Above(MaxString(MyField),0,3))
```

Restituisce il numero di valori numerici trovati nei tre risultati della funzione **MaxString(MyField)** valutati in base alla riga attuale e alle due righe che la precedono.

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati campione

MyField	RangeNumericCount(Above(MaxString(MyField),0,3))
10	1
abc	1
8	2
def	1
xyz	1
9	1

Dati utilizzati negli esempi:

```
RangeTab:
```

```
LOAD * INLINE [
MyField
10
'abc'
8
def
xyz
9
] ;
```

Vedere anche:

p *NumericCount - funzione per grafici (page 365)*

RangeOnly

RangeOnly() è una funzione duale che restituisce un valore se l'espressione viene valutata in base a un valore univoco. In caso contrario, viene restituito **NULL**.

Sintassi:

```
RangeOnly(first_expr[, Expression])
```

Tipo di dati restituiti: duale

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Esempi e risultati:

Esempi	Risultati
`RangeOnly (1,2,4)`	Restituisce NULL
`RangeOnly (5,'abc')`	Restituisce NULL
`RangeOnly (null( ), 'abc')`	Restituisce 'abc'
`RangeOnly(10,10,10)`	Restituisce 10

Vedere anche:

p *Only - funzione per grafici (page 348)*

RangeSkew

RangeSkew() restituisce il valore che corrisponde all'asimmetria di una scala di numeri.

Sintassi:

```
RangeSkew(first_expr[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

Se non viene trovato nessun valore numerico, viene restituito NULL.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`rangeskew (1,2,4)`	Restituisce 0,93521952958283
`rangeskew (above (SalesValue,0,3))`	Restituisce il grado di asimmetria mobile della scala dei tre valori restituiti dalla funzione above() calcolata in base alla riga attuale e alle due righe che la precedono.

Dati utilizzati nell'esempio:

Dati campione

CustID	RangeSkew(Above(SalesValue,0,3))
1-20	-, -, 0.5676, 0.8455, 1.0127, -0.8741, 1.7243, -1.7186, 1.5518, 1.4332, 0, 1.1066, 1.3458, 1.5636, 1.5439, 0.6952, -0.3766

```
SalesTable:
LOAD recno() as CustID, * inline [
SalesValue
101
163
126
```

```
139
167
86
83
22
32
70
108
124
176
113
95
32
42
92
61
21
] ;
```

Vedere anche:

p *Skew - funzione per grafici (page 445)*

RangeStdev

RangeStdev() trova la deviazione standard di una scala di numeri.

Sintassi:

```
RangeStdev(first_expr[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

Se non viene trovato nessun valore numerico, viene restituito NULL.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeStdev (1,2,4)`	Restituisce 1,5275252316519
`RangeStdev (null( ))`	Restituisce NULL
`RangeStdev (above (SalesValue),0,3))`	Restituisce la deviazione standard mobile della scala dei tre valori restituiti dalla funzione above() calcolata in base alla riga attuale e alle due righe che la precedono.

Dati utilizzati nell'esempio:

Dati campione

CustID	RangeStdev(SalesValue, 0,3))
1-20	-,43.841, 34.192, 18.771, 20.953, 41.138, 47.655, 36.116, 32.716, 25.325, 38,000, 27.737, 35.553, 33.650, 42.532, 33.858, 32.146, 25.239, 35.595

```
SalesTable:
LOAD recno() as CustID, * inline [
SalesValue
101
163
126
139
167
86
83
22
32
70
108
124
176
113
95
32
42
92
61
21
] ;
```

Vedere anche:

p *Stdev - funzione per grafici (page 448)*

RangeSum

RangeSum() restituisce la somma di una scala di valori. Tutti i valori non numerici vengono trattati come 0.

Sintassi:

```
RangeSum(first_expr[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomenti

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Limiti:

La funzione **RangeSum** tratta tutti i valori non numerici come 0.

Esempi e risultati:

Esempi

Esempi	Risultati
`RangeSum (1,2,4)`	Restituisce 7
`RangeSum (5,'abc')`	Restituisce 5
`RangeSum (null( ))`	Restituisce 0

Esempio:

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.

```
RangeTab3:

LOAD recno() as RangeID, Rangesum(Field1,Field2,Field3) as MyRangeSum INLINE [

Field1, Field2, Field3

10,5,6

2,3,7

8,2,8

18,11,9
```

```
5,5,9

9,4,2
];
```

La tabella risultante mostra i valori restituiti in MyRangeSum per ciascun record della tabella.

Tabella risultante

RangeID	MyRangeSum
1	21
2	12
3	18
4	38
5	19
6	15

Esempio con espressione:

```
RangeSum (Above(MyField,0,3))
```

Restituisce la somma dei tre valori di **MyField)**: dalla riga attuale e dalle due righe che la precedono. Specificando 3 per il terzo argomento, la funzione **Above()** restituisce tre valori, se il numero di righe che precedono la riga attuale è sufficiente, che vengono considerati come input per la funzione **RangeSum()**.

Dati utilizzati negli esempi:

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati campione

MyField	RangeSum(Above(MyField,0,3))
10	10
2	12
8	20
18	28
5	31
9	32

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
2
```

```
8
18
5
9
] ;
```

Vedere anche:

p *Sum - funzione per grafici (page 351)*
p *Above - funzione per grafici (page 1267)*

RangeTextCount

RangeTextCount() restituisce il numero di valori di testo nel campo o nell'espressione.

Sintassi:

```
RangeTextCount(first_expr[, Expression])
```

Tipo di dati restituiti: numero intero

Argomenti:

Gli argomenti di questa funzione possono contenere funzioni intra-record, che a loro volta restituiscono un elenco di valori.

Argomento

Argomento	Descrizione
first_expr	L'espressione o il campo contenente i dati da misurare.
Expression	Le espressioni o campi opzionali contenenti la scala di dati da misurare.

Esempi e risultati:

Esempi di funzioni

Esempi	Risultati
`RangeTextCount (1,2,4)`	Restituisce 0
`RangeTextCount (5,'abc')`	Restituisce 1
`RangeTextCount (null( ))`	Restituisce 0

Esempio con espressione:

```
RangeTextCount (Above(MaxString(MyField),0,3))
```

Restituisce il numero di valori di testo all'interno dei tre risultati della funzione **MaxString(MyField)** valutati in base alla riga attuale e alle due righe che la precedono.

Dati utilizzati negli esempi:

*Disattivare l'ordinamento di **MyField** per garantire il funzionamento corretto dell'esempio.*

Dati di esempio

MyField	MaxString(MyField)	RangeTextCount(Above(Sum(MyField),0,3))
10	10	0
abc	abc	1
8	8	1
def	def	2
xyz	xyz	2
9	9	2

Dati utilizzati negli esempi:

```
RangeTab:
LOAD * INLINE [
MyField
10
'abc'
8
null()
'xyz'
9
] ;
```

Vedere anche:

p *TextCount - funzione per grafici (page 369)*

RangeXIRR

RangeXIRR() restituisce il tasso interno di ritorno per una programmazione di flussi di denaro che non è necessariamente su base periodica. Per calcolare il tasso di rendimento interno di una serie di flussi di cassa periodici, utilizzare la funzione **RangeIRR**.

Sintassi:

```
RangeXIRR(value, date{, value, date})
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
value	Un flusso di cassa o una serie di flussi di cassa che corrisponde a una programmazione di date di pagamento. La serie di valori deve contenere almeno un valore positivo e uno negativo.
date	Una data di pagamento o una programmazione di date di pagamento che corrisponde ai pagamenti con flusso di cassa.

Limiti:

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Tutti i pagamenti sono scontati in base ad un anno composto da 365 giorni.

Esempi	Risultati
`RangeXIRR(-2500,'2008-01-01',2750,'2008-09-01')`	Restituisce 0,1532

Vedere anche:

p *RangeIRR (page 1332)*

RangeXNPV

RangeXNPV() restituisce il valore attuale netto per una programmazione di flussi di cassa non necessariamente periodica. Il risultato è in un formato numerico predefinito della valuta. Per calcolare il valore attuale netto per una serie di flussi di cassa periodici, utilizzare la funzione **RangeNPV**.

Sintassi:

```
RangeXNPV(discount_rate, values, dates[, Expression])
```

Tipo di dati restituiti: numerico

Argomenti:

Argomenti

Argomento	Descrizione
discount_rate	Il tasso di interesse per periodo.

Argomento	Descrizione
values	Un flusso di cassa o una serie di flussi di cassa che corrisponde a una programmazione di date di pagamento. Ciascun valore può essere un valore singolo oppure una scala di valori restituita da una funzione intra-record con un terzo parametro opzionale. La serie di valori deve contenere almeno un valore positivo e uno negativo.
dates	Una data di pagamento o una programmazione di date di pagamento che corrisponde ai pagamenti con flusso di cassa.

Limiti:

I valori di testo, i valori NULL e i valori mancanti vengono ignorati.

Tutti i pagamenti sono scontati in base ad un anno composto da 365 giorni.

Tabella di esempio

Esempi	Risultati
`RangeXNPV(0.1, -2500,'2008-01-01',2750,'2008-09-01')`	Restituisce 80,25

Aggiungere lo script di esempio all'app ed eseguirlo. Per visualizzare il risultato, aggiungere i campi elencati nella colonna risultati a un foglio nell'app dell'utente.
`RangeTab3:`
`LOAD *,`
`recno() as RangeID,`
`RangeXNPV(Field1,Field2,Field3) as RangeNPV;`
`LOAD * INLINE [`
`Field1|Field2|Field3`
`10|5|-6000`
`2|NULL|7000`
`8|'abc'|8000`
`18|11|9000`
`5|5|9000`
`9|4|2000`
`] (delimiter is '|');`
	La tabella risultante mostra i valori restituiti in RangeXNPV per ciascun record della tabella.
	RangeID — RangeXNPV
	1 — $-49.13
	2 — $777.78
	3 — $98.77
	4 — $25.51
	5 — $250.83
	6 — $20.40

5.22 Funzioni relazionali

Si tratta di un gruppo di funzioni che calcolano le proprietà dei singoli valori dimensionali in un grafico, utilizzando numeri già aggregati.

Si tratta di un gruppo di funzioni che calcolano le proprietà dei singoli valori dimensionali in un grafico, utilizzando numeri già aggregati. Ad esempio, non è possibile calcolare una classificazione senza un confronto con altri valori dimensionali.

Queste funzioni possono essere utilizzate solo nelle espressioni grafiche. Non possono essere utilizzati nello script di caricamento.

Nel grafico è richiesta una dimensione, che definisce gli altri punti di dati necessari per il confronto. Di conseguenza, una funzione relazionale non è significativa in un grafico senza dimensioni (ad esempio, un oggetto KPI).

Funzioni di classificazione

Se si utilizzano queste funzioni, verrà automaticamente disabilitata la soppressione dei valori zero. I valori NULL vengono ignorati.

Rank

Rank() valuta le righe del grafico nell'espressione, e per ciascuna riga, visualizza la posizione relativa del valore della dimensione valutata nell'espressione. Quando valuta l'espressione, la funzione confronta il risultato con quello delle altre righe contenenti il segmento di colonna attuale e restituisce la classificazione della riga attuale all'interno del segmento.

```
Rank - funzione per grafici([TOTAL [<fld {, fld}>]] expr[, mode[, fmt]])
```

HRank

HRank() valuta l'espressione, confronta il risultato con il risultato delle altre colonne contenenti il segmento di riga attuale di una tabella pivot. La funzione quindi restituisce la classificazione della colonna attuale all'interno del segmento.

```
HRank - funzione per grafici([TOTAL] expr[, mode[, fmt]])
```

Funzioni di raggruppamento

KMeans2D

Il gruppo di proprietà **Licenza sito** contiene proprietà correlate alla licenza per il sistema Qlik Sense. Tutti i campi sono obbligatori e non devono essere vuoti.

Proprietà licenza sito

Nome della proprietà	Descrizione
Nome proprietario	Il nome utente del proprietario del prodotto Qlik Sense.
Organizzazione proprietario	Il nome dell'organizzazione di cui il proprietario prodotto Qlik Sense è un membro.
Numero di serie	Il numero di serie assegnato al software Qlik Sense.
Codice di controllo	Il numero di controllo assegnato al software Qlik Sense.
Accesso LEF	Il License Enabler File (LEF) assegnato al software Qlik Sense.

KMeans2D() valuta le righe del grafico applicando il clustering K-means, e per ciascuna riga del grafico visualizza l'id cluster del cluster a cui è stato assegnato questo punto dati. Le colonne utilizzate dall'algoritmo di clustering sono determinate rispettivamente dai parametri coordinate_1 e coordinate_2. Sono entrambe aggregazioni. Il numero di cluster creati è determinato dal parametro num_clusters. I dati possono essere normalizzati in via opzionale dal parametro norm.

```
KMeans2D - funzione per grafici(num_clusters, coordinate_1, coordinate_2 [,
norm])
```

KMeansND

KMeansND() valuta le righe del grafico applicando il clustering K-means, e per ciascuna riga del grafico visualizza l'id cluster del cluster a cui è stato assegnato questo punto dati. Le colonne utilizzate dall'algoritmo di clustering sono determinate dai parametri coordinate_1, coordinate_2, ecc. fino a n colonne. Sono tutte aggregazioni. Il numero di cluster creati è determinato dal parametro num_clusters.

```
KMeansND - funzione per grafici(num_clusters, num_iter, coordinate_1,
coordinate_2 [,coordinate_3 [, ...]])
```

KMeansCentroid2D

KMeansCentroid2D() valuta le righe del grafico applicando il clustering K-means, e per ciascuna riga del grafico visualizza la coordinata desiderata del cluster a cui è stato assegnato questo punto dati. Le colonne utilizzate dall'algoritmo di clustering sono determinate rispettivamente dai parametri coordinate_1 e coordinate_2. Sono entrambe aggregazioni. Il numero di cluster creati è determinato dal parametro num_clusters. I dati possono essere normalizzati in via opzionale dal parametro norm.

```
KMeansCentroid2D - funzione per grafici(num_clusters, coordinate_no,
coordinate_1, coordinate_2 [, norm])
```

KMeansCentroidND

KMeansCentroidND() valuta le righe del grafico applicando il clustering K-means, e per ciascuna riga del grafico visualizza la coordinata desiderata del cluster a cui è stato assegnato questo punto dati. Le colonne utilizzate dall'algoritmo di clustering sono determinate dai parametri coordinate_1, coordinate_2, ecc. fino a n colonne. Sono tutte aggregazioni. Il numero di cluster creati è determinato dal parametro num_clusters.

```
KMeansCentroidND - funzione per grafici(num_clusters, num_iter, coordinate_
no, coordinate_1, coordinate_2 [,coordinate_3 [, ...]])
```

Funzioni di scomposizione serie temporale

STL_Trend

STL_Trend è una funzione di scomposizione delle serie temporali. Insieme a STL_Seasonal e a STL_Residual, questa funzione viene utilizzata per scomporre una serie temporale in componenti stagionali, di tendenza e residuali. Nel contesto dell'algoritmo STL, la scomposizione delle serie temporali viene utilizzata per identificare sia un modello stagionale ricorrente sia una tendenza generale, data una metrica di ingresso e altri parametri. La funzione STL_Trend identifica una tendenza generale, indipendente da schemi o cicli stagionali, dai dati delle serie temporali.

```
STL_Trend - funzione per grafici(Expression, period [,seasonal_smoother
[,trend_smoother]])
```

STL_Seasonal

STL_Seasonal è una funzione di scomposizione delle serie temporali. Insieme a STL_Trend e a STL_Residual, questa funzione viene utilizzata per scomporre una serie temporale in componenti stagionali, di tendenza e residuali. Nel contesto dell'algoritmo STL, la scomposizione delle serie temporali viene

utilizzata per identificare sia un modello stagionale ricorrente sia una tendenza generale, data una metrica di ingresso e altri parametri. La funzione **STL_Seasonal** è in grado di identificare un modello stagionale all'interno di una serie temporale, separandolo dalla tendenza generale mostrata dai dati.

```
STL_Seasonal - funzione per grafici(Expression, period [,seasonal_smoother
[,trend_smoother]])
```

STL_Residual

STL_Residual è una funzione di scomposizione delle serie temporali. Insieme a **STL_Seasonal** e a **STL_Trend**, questa funzione viene utilizzata per scomporre una serie temporale in componenti stagionali, di tendenza e residuali. Nel contesto dell'algoritmo STL, la scomposizione delle serie temporali viene utilizzata per identificare sia un modello stagionale ricorrente sia una tendenza generale, data una metrica di ingresso e altri parametri. Nell'eseguire questa operazione, una parte della variazione della metrica di input non rientrerà né nella componente stagionale né in quella di tendenza e sarà definita come componente residua. La funzione grafico **STL_Residual** acquisisce questa parte del calcolo.

```
STL_Residual - funzione per grafici(Expression, period [,seasonal_smoother
[,trend_smoother]])
```

Rank - funzione per grafici

Rank() valuta le righe del grafico nell'espressione, e per ciascuna riga, visualizza la posizione relativa del valore della dimensione valutata nell'espressione. Quando valuta l'espressione, la funzione confronta il risultato con quello delle altre righe contenenti il segmento di colonna attuale e restituisce la classificazione della riga attuale all'interno del segmento.

Segmenti delle colonne

Region	Country	Population	Rank(Population)
Americas	Mexico	128,932,753	2
Americas	Canada	37,742,154	3
Americas	United States of America	331,002,651	1
Europe	Sweden	10,099,265	4
Europe	United Kingdom	67,886,011	2
Europe	France	65,273,511	3
Europe	Germany	83,783,942	1

Per i grafici diversi dalla tabelle, il segmento di colonna attuale è definito come visualizzato nell'equivalente di tabella lineare del grafico.

Sintassi:

```
Rank([TOTAL] expr[, mode[, fmt]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
mode	Specifica la rappresentazione numerica del risultato della funzione.
fmt	Specifica la rappresentazione testuale del risultato della funzione.

Argomento	Descrizione
TOTAL	Se il grafico è unidimensionale o se l'espressione è preceduta dal qualificatore **TOTAL**, la funzione viene valutata lungo l'intera colonna. Se la tabella o l'equivalente della tabella presenta più dimensioni verticali, il segmento di colonna attuale includerà solo righe con lo stesso valore della riga attuale in tutte le colonne di dimensione, eccetto la colonna che mostra l'ultima dimensione nell'ordinamento tra campi.

La classificazione viene restituita come valore duale che, nel caso in cui ogni riga presenti una classificazione univoca, sarà un numero intero compreso tra 1 e il numero di righe nel segmento di colonna attuale.

Nel caso in cui molte righe condividano la stessa classificazione, la rappresentazione testuale e numerica potrà essere controllata mediante i parametri **mode** e **fmt**.

mode

Il secondo argomento, **mode**, può assumere i seguenti valori:

Esempi **mode**

Valore	Descrizione
0 (predefinito)	Se tutte le classificazioni all'interno del gruppo condiviso rientrano nella parte bassa del valore medio dell'intera classificazione, tutte le righe all'interno di tale gruppo riceveranno la classificazione più bassa. Se tutte le classificazioni all'interno del gruppo condiviso rientrano nella parte alta del valore medio dell'intera classificazione, tutte le righe all'interno di tale gruppo riceveranno la classificazione più alta. Se tutte le classificazioni all'interno del gruppo condiviso si distribuiscono lungo il valore medio dell'intera classificazione, tutte le righe all'interno di tale gruppo riceveranno il valore corrispondente alla media calcolata in base alla classificazione massima e minima dell'intero segmento di colonna.
1	Classificazione minima su tutte le righe.
2	Classificazione media su tutte le righe.
3	Classificazione massima su tutte le righe.
4	Classificazione minima sulla prima riga, quindi incrementata di uno per ogni riga.

fmt

Il terzo argomento, **fmt**, può assumere i seguenti valori:

Esempi **fmt**

Valore	Descrizione
0 (predefinito)	Valore basso **-** valore alto su tutte le righe (ad esempio, 3 - 4).

Valore	Descrizione
1	Valore basso su tutte le righe.
2	Valore basso sulla prima riga, vuoto sulle righe successive.

L'ordine delle righe per **mode** 4 e **fmt** 2 è determinato dall'ordine delle dimensioni del grafico.

Esempi e risultati:

Creare due visualizzazioni dalle dimensioni Product e Sales e un'altra da Product e UnitSales. Aggiungere le misure come mostrato nella seguente tabella.

Esempi di classificazione

Esempi	Risultati
Esempio 1. Creare una tabella con le dimensioni `Customer` e `Sales` e la misura `Rank(Sales)`	Il risultato dipende dall'ordinamento delle dimensioni. Se viene ordinata in base a Customer, la tabella elencherà tutti i valori di Sales per Astrida, quindi Betacab e così via. Il risultato per Rank(Sales) mostrerà 10 per il valore Sales 12, 9 per il valore Sales 13 e così via, con il valore di classificazione 1 restituito per il valore Sales 78. Il segmento di colonna successivo inizia con Betacab, per il quale, il primo valore di Sales nel segmento è 12. Il valore di classificazione di Rank(Sales) per questo è fornito come 11. Se la tabella viene ordinata in base a Sales, i segmenti colonna consisteranno nei valori di Sales e del Customer corrispondente. Poiché vi sono due valori Sales di 12 (per Astrida e Betacab), il valore di Rank(Sales) per tale segmento di colonna è 1-2, per ciascun valore di Customer. Questo perché sono presenti due valori di Customer per il valore Sales 12. Se fossero stati presenti 4 valori, il risultato sarebbe stato 1-4 per tutte le righe. Questo mostra l'aspetto del risultato per il valore predefinito (0) dell'argomento fmt.
Esempio 2. Sostituire la dimensione Customer con Product e aggiungere la misura `Rank(Sales,1,2)`	Viene restituito 1 sulla prima riga di ciascun segmento di colonna, mentre tutte le altre righe vengono lasciate vuote perché gli argomenti **mode** e **fmt** sono impostati rispettivamente su 1 e 2.

Risultati per l'esempio 1, con la tabella ordinata per Customer:

Tabella dei risultati

Customer	Sales	Rank(Sales)
Astrida	12	10
Astrida	13	9

Customer	Sales	Rank(Sales)
Astrida	20	8
Astrida	22	7
Astrida	45	6
Astrida	46	5
Astrida	60	4
Astrida	65	3
Astrida	70	2
Astrida	78	1
Betcab	12	11

Risultati per l'esempio 1, con la tabella ordinata per Sales:

Tabella dei risultati

Customer	Sales	Rank(Sales)
Astrida	12	1-2
Betacab	12	1-2
Astrida	13	1
Betacab	15	1
Astrida	20	1
Astrida	22	1-2
Betacab	22	1-2
Betacab	24	1-2
Canutility	24	1-2

Dati utilizzati negli esempi:

```
ProductData:
Load * inline [
Customer|Product|UnitSales|UnitPrice
Astrida|AA|4|16
Astrida|AA|10|15
Astrida|BB|9|9
Betacab|BB|5|10
Betacab|CC|2|20
Betacab|DD|0|25
Canutility|AA|8|15
Canutility|CC|0|19
] (delimiter is '|');

Sales2013:
```

```
crosstable (Month, Sales) LOAD * inline [
Customer|Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec
Astrida|46|60|70|13|78|20|45|65|78|12|78|22
Betacab|65|56|22|79|12|56|45|24|32|78|55|15
Canutility|77|68|34|91|24|68|57|36|44|90|67|27
Divadip|57|36|44|90|67|27|57|68|47|90|80|94
] (delimiter is '|');
```

Vedere anche:

p *Sum - funzione per grafici (page 351)*

HRank - funzione per grafici

HRank() valuta l'espressione, confronta il risultato con il risultato delle altre colonne contenenti il segmento di riga attuale di una tabella pivot. La funzione quindi restituisce la classificazione della colonna attuale all'interno del segmento.

Sintassi:

```
HRank([ TOTAL ] expr [ , mode [, fmt ] ])
```

Tipo di dati restituiti: duale

Questa funzione è valida solo per le tabelle pivot. In tutti gli altri tipi di grafici restituisce NULL.

Argomenti:

Argomenti

Argomento	Descrizione
expr	L'espressione o il campo contenente i dati da misurare.
mode	Specifica la rappresentazione numerica del risultato della funzione.
fmt	Specifica la rappresentazione testuale del risultato della funzione.
TOTAL	Se il grafico è unidimensionale o se l'espressione è preceduta dal qualificatore **TOTAL**, la funzione viene valutata lungo l'intera colonna. Se la tabella o l'equivalente della tabella presenta più dimensioni verticali, il segmento di colonna attuale includerà solo righe con lo stesso valore della riga attuale in tutte le colonne di dimensione, eccetto la colonna che mostra l'ultima dimensione nell'ordinamento tra campi.

Se la tabella pivot è unidimensionale o se è preceduta dal qualificatore **total**, il segmento di riga attuale sarà sempre uguale all'intera riga. Se una tabella pivot include più dimensioni orizzontali, il segmento della riga attuale includerà solo le colonne con gli stessi valori della colonna attuale in tutte le righe della dimensione, tranne per la riga che visualizza l'ultima dimensione orizzontale dell'ordinamento tra campi.

La classificazione viene restituita come valore duale che, nel caso in cui ciascuna colonna presenti una classificazione univoca, avrà come valore un numero intero compreso tra 1 e il numero di colonne nel segmento della riga attuale.

Nel caso in cui molte righe condividano la stessa classificazione, la rappresentazione testuale e numerica potrà essere controllata mediante gli argomenti **mode** e **format**.

Il secondo argomento **mode** specifica la rappresentazione numerica del risultato della funzione:

Esempi **mode**

Valore	Descrizione
0 (predefinito)	Se tutte le classificazioni all'interno del gruppo condiviso ricadono nella parte bassa del valore medio dell'intera classificazione, tutte le colonne all'interno di tale gruppo riceveranno la classificazione più bassa. Se tutte le classificazioni all'interno del gruppo condiviso ricadono nella parte alta del valore medio dell'intera classificazione, tutte le colonne all'interno di tale gruppo riceveranno la classificazione più alta. Se tutte le classificazioni all'interno del gruppo condiviso si distribuiscono lungo il valore medio dell'intera classificazione, tutte le righe all'interno di tale gruppo riceveranno il valore corrispondente alla media calcolata in base alla classificazione massima e minima dell'intero segmento di colonna.
1	Classificazione minima su tutte le colonne nel gruppo.
2	Classificazione media su tutte le colonne nel gruppo.
3	Classificazione massima su tutte le colonne nel gruppo.
4	Classificazione minima sulla prima colonna, quindi incrementata di uno per ciascuna colonna nel gruppo.

Il terzo argomento, **format**, specifica la rappresentazione testuale del risultato della funzione:

Esempi **format**

Valore	Descrizione
0 (predefinito)	Valore basso**&' - '&**valore alto su tutte le colonne (ad esempio 3 - 4).
1	Valore basso su tutte le colonne nel gruppo.
2	Valore basso nella prima colonna, vuoto nelle colonne successive nel gruppo.

L'ordine delle colonne per **mode** 4 e **format** 2 è determinato dall'ordinamento delle dimensioni del grafico.

Esempi:

```
HRank( sum( Sales ))
HRank( sum( Sales ), 2 )
HRank( sum( Sales ), 0, 1 )
```

Ottimizzazione con k-means: Un esempio del mondo reale

L'esempio seguente illustra un caso d'uso del mondo reale in cui le funzioni di clustering KMeans e Centroide sono applicate a un set di dati. La funzione KMeans segrega i punti dati in cluster che condividono somiglianze. I cluster diventano più compatti e differenziati dato che l'algoritmo KMeans viene applicato su un numero configurabile di iterazioni.

KMeans è una funzione utilizzata in molti campi in un'ampia varietà di casi di utilizzo; alcuni esempi di casi di utilizzo del clustering includono la segmentazione dei clienti, il rilevamento di frodi, la previsione di attriti negli account, il targeting degli incentivi ai clienti, l'identificazione del cybercrimine e l'ottimizzazione dei percorsi di consegna. L'algoritmo di clustering KMeans risulta sempre più utilizzato laddove le aziende cercano di inferire schemi e ottimizzare le offerte di servizi.

Funzioni KMeans e Centroid di Qlik Sense

Qlik Sense fornisce due funzioni KMeans che raggruppano i punti dati in cluster in base alla somiglianza. Vedere *KMeans2D - funzione per grafici (page 1376)* e *KMeansND - funzione per grafici (page 1391)*. La funzione **KMeans2D** accetta due dimensioni e funziona bene al momento di visualizzare risultati attraverso un **grafico a dispersione**. La funzione **KMeansND** accetta più di due dimensioni. Essendo semplice concettualizzare un esito 2D su grafici standard, la seguente dimostrazione applica KMeans a un **grafico a dispersione** usando due dimensioni. Il clustering KMeans può essere visualizzato attraverso il coloramento per espressione; oppure per dimensione come descritto in questo esempio.

Le funzioni centroide di Qlik Sense determinano la posizione aritmetica media di tutti i punti dati nel cluster e identificano un punto centrale, o centroide, per tale cluster. Per ciascuna riga del grafico (o record), la funzione centroide visualizza la coordinata del cluster a cui è stato assegnato questo punto dati. Vedere *KMeansCentroid2D - funzione per grafici (page 1406)* e *KMeansCentroidND - funzione per grafici (page 1407)*.

Panoramica sui casi di utilizzo ed esempio

Il seguente esempio si riferisce a uno scenario simulato del mondo reale. Un'azienda tessile nello stato di New York, USA, interessata a ridurre le spese minimizzando i costi di consegna. Un modo per farlo è trasferire i magazzini più vicino ai loro distributori. L'azienda impiega 118 distributori in tutto lo stato di New York. La seguente dimostrazione simula come un operations manager potrebbe segmentare i distributori in cinque geografie raggruppate usando la funzione KMeans e poi identificare cinque posizioni di magazzino ottimali centrali rispetto a quei cluster usando la funzione centroide. L'obiettivo è individuare le coordinate di mappatura utilizzabili per identificare cinque posizioni centrali del magazzino.

Il set di dati

Il set di dati è basato su nomi e indirizzi generati casualmente nello stato di New York con coordinate reali di latitudine e longitudine. Il set di dati contiene le seguenti dieci colonne: id, first_name, last_name, telefono, indirizzo, città, stato, codice postale, latitudine, longitudine. Il set di dati è disponibile in basso come file da scaricare localmente e poi caricare in Qlik Sense o inline per l'editor caricamento dati. L'app creata viene denominata *KMeans e Centroide distributori* e il primo foglio nell'app viene denominato *Analisi cluster di distribuzione*.

Selezionare il collegamento seguente per scaricare il file di dati campione: DistributorData.csv

Set di dati Distributor: Caricamento inline per l'editor caricamento dati in Qlik Sense (page 1374)

Titolo: DistributorData

Numero totale di record: 118

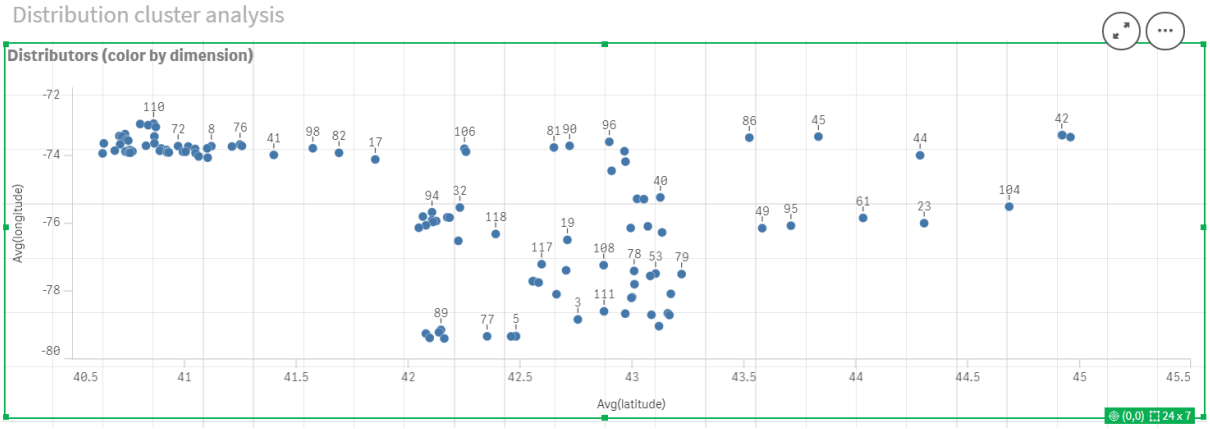
Applicazione della funzione KMeans2D

In questo esempio, la configurazione di un **grafico a dispersione** viene dimostrata utilizzando il set di dati *DistributorData*, viene applicata la funzione **KMeans2D**, e il grafico viene colorato per dimensione.

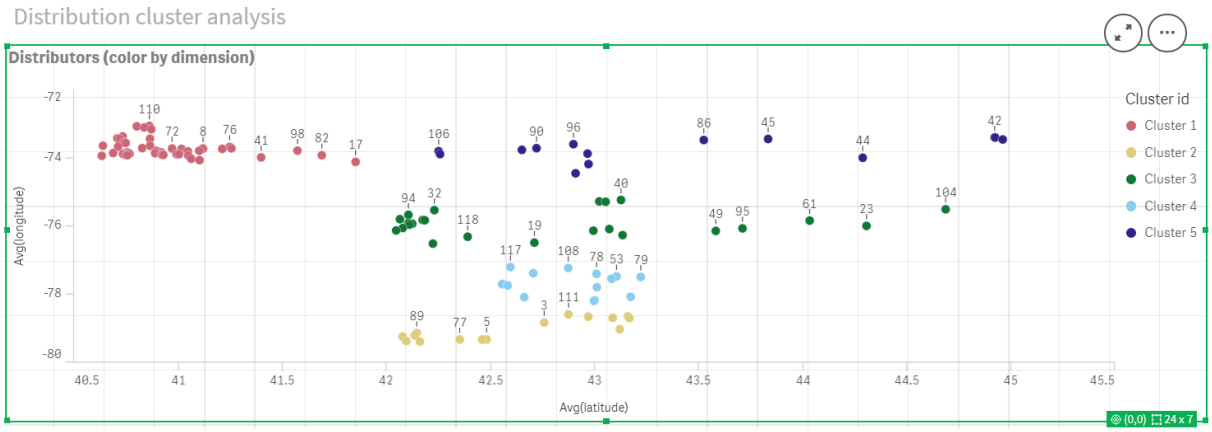
Notare che le funzioni KMeans Qlik Sense supportano il clustering automatico usando un metodo definito differenza di profondità (DeD). Quando un utente imposta lo 0 per il numero di cluster, viene determinato il numero ottimale di cluster per tale set di dati. Tuttavia, per questo esempio viene creata una variabile per l'argomento **num_clusters** (fare riferimento a *KMeans2D - funzione per grafici (page 1376)* per la sintassi). Pertanto, il numero desiderato di cluster (k=5) viene specificato mediante una variabile.

1. Un **Grafico a dispersione** è trascinato sul foglio e denominato *Distributori (per dimensione)*.
2. Viene creata una **variabile** per specificare il numero di cluster. La **variabile** viene denominata *vDistClusters*. Per la variabile **Definizione**, viene inserito *5*.
3. Configurazione **Dati** per il grafico:
    a. Sotto **Dimensioni**, il campo *id* viene selezionato per **Bolla**. *Id cluster* viene inserito per l'**Etichetta**.
    b. Sotto **Misure**, *Avg([latitude])* rappresenta l'espressione per l'**asse X**.
    c. Sotto **Misure**, *Avg([longitude])* rappresenta l'espressione per l'**asse Y**.
4. Configurazione dell'**Aspetto**:
    a. Sotto **Colori e legenda**, viene scelto **Personalizzato** per **Colori**.
    b. **Per dimensione** viene selezionato per colorare il grafico.
    c. Viene inserita la seguente espressione: *=pick(aggr(KMeans2D(vDistClusters,only (latitude),only(longitude)),id)+1, 'Cluster 1', 'Cluster 2', 'Cluster 3', 'Cluster 4', 'Cluster 5')*
    d. Viene selezionata la casella di controllo per **Colori persistenti**.

Viene applicato Grafico a dispersione prima del coloramento KMeans per dimensione



Viene applicato Grafico a dispersione dopo il coloramento KMeans per dimensione



Aggiunta di una **tabella**: *Distributori*

Può essere utile avere a disposizione una tabella per l'accesso rapido ai dati pertinenti. Il grafico a **dispersione** mostra gli *id* attraverso una tabella con i nomi dei distributori corrispondenti che viene aggiunta per riferimento.

1. Una **tabella** denominata *Distributori* viene trascinata sul foglio con le seguenti **Colonne** (Dimensioni) aggiunte: *id*, *first_name* e *last_name*.

Tabella: Nomi distributore

Distributors

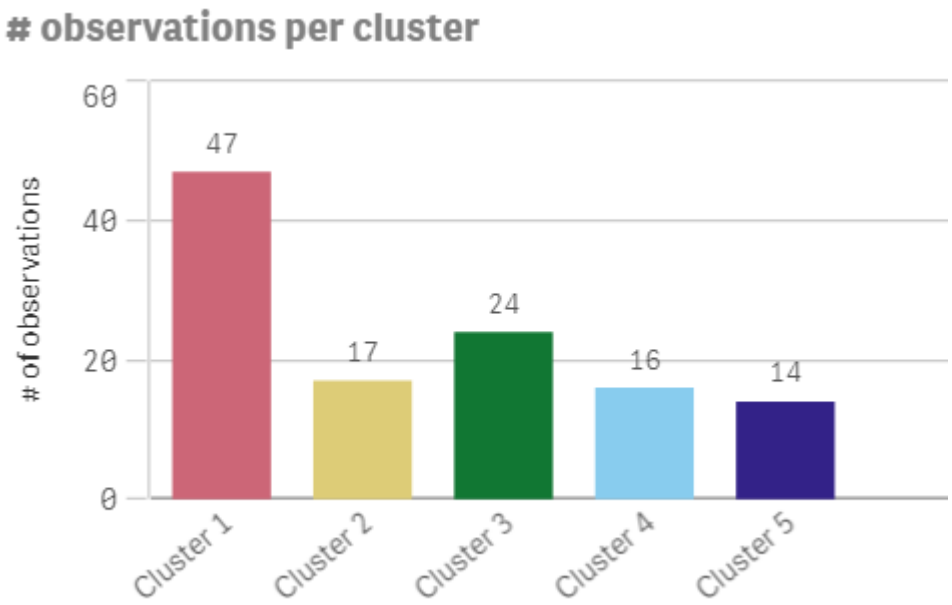
id	first_name	last_name
1	Kaiya	Snow
2	Dean	Roy
3	Eden	Paul
4	Bryanna	Higgins
5	Elisabeth	Lee
6	Skylar	Robinson
7	Cody	Bailey
8	Dario	Sims
9	Deacon	Hood

Aggiunta di un **grafico a barre**: *# osservazioni per cluster*

Per uno scenario di distribuzione magazzino, è utile conoscere quanti distributori verranno serviti da ciascun magazzino. Pertanto, viene creato un **grafico a barre** che misuri quanti distributori vengono assegnati a ciascun cluster.

1. Un **grafico a barre** viene trascinato sul foglio. Il grafico è denominato: *# osservazioni per cluster*.
2. Configurazione **Dati** per il **grafico a barre**:
    a. Viene aggiunta una **Dimensione** denominata *Cluster* (l'etichetta può essere aggiunta dopo l'applicazione dell'espressione). Viene inserita la seguente espressione: *=pick(aggr (KMeans2D(vDistClusters,only(latitude),only(longitude)),id)+1, 'Cluster 1', 'Cluster 2', 'Cluster 3', 'Cluster 4', 'Cluster 5')*
    b. Viene aggiunta una **Misura** denominata *# di osservazioni*. Viene inserita la seguente espressione: *=count(aggr(KMeans2D(vDistClusters,only(latitude),only(longitude)),id))*
3. Configurazione dell'**Aspetto**:
    a. Sotto **Colori e legenda**, viene scelto **Personalizzato** per **Colori**.
    b. **Per dimensione** viene selezionato per colorare il grafico.
    c. Viene inserita la seguente espressione: *=pick(aggr(KMeans2D(vDistClusters,only (latitude),only(longitude)),id)+1, 'Cluster 1', 'Cluster 2', 'Cluster 3', 'Cluster 4', 'Cluster 5')*
    d. Viene selezionata la casella di controllo per **Colori persistenti**.
    e. **Mostra legenda** è disattivato.
    f. Sotto **Presentazione**, **Etichette valore** è impostato su **Auto**.
    g. Sotto **Asse X**: È selezionato **Cluster**, **Solo etichette**.

Grafico a barre: # osservazioni per cluster



Applicazione della funzione **Centroid2D**

Viene aggiunta una seconda tabella per la funzione **Centroid2D** che identificherà le coordinate per le potenziali posizioni del magazzino. Questa tabella mostra la posizione centrale (valori centroide) per i cinque gruppi distributori identificati.

1. Una **Tabella** viene trascinata sul foglio e denominata *Centroidi cluster* con le seguenti colonne aggiunte:

a. Viene aggiunta una **Dimensione** denominata *Cluster*. Viene inserita la seguente espressione: *=pick(aggr(KMeans2D(vDistClusters,only(latitude),only (longitude)),id)+1,'Warehouse 1','Warehouse 2','Warehouse 3','Warehouse 4','Warehouse 5'*)
b. Viene aggiunta una **Misura** denominata *latitudine (D1)*. Viene inserita la seguente espressione: *=only(aggr(KMeansCentroid2D(vDistClusters,0,only(latitude),only (longitude)),id))*
   Notare che il parametro **coordinate_no** corrisponde alla prima dimensione(0). In questo caso la dimensione *latitudine* viene tracciata rispetto all'asse x. Se stessimo lavorando con la funzione **CentroidND** e ci fossero fino a sei dimensioni, tali voci di parametro potrebbero corrispondere a uno qualsiasi di sei valori: 0,1,2,3,4,o 5.
c. Viene aggiunta una **Misura** denominata *longitudine (D2)*. Viene inserita la seguente espressione: *=only(aggr(KMeansCentroid2D(vDistClusters,1,only(latitude),only (longitude)),id))*
   Il parametro **coordinate_no** in questa espressione corrisponde alla seconda dimensione(1). La dimensione *longitudine* viene tracciata rispetto all'asse y.

Tabella: Calcoli centroide cluster

Cluster centroids

Clusters	latitude (D1)	longitude (D2)
Totals	-	-
Warehouse 1	40.945422240426	-73.719966482979
Warehouse 2	42.590538729412	-79.067889217647
Warehouse 3	42.805089516667	-75.901621883333
Warehouse 4	42.8581692625	-77.6800485875
Warehouse 5	43.436770771429	-73.734622635714

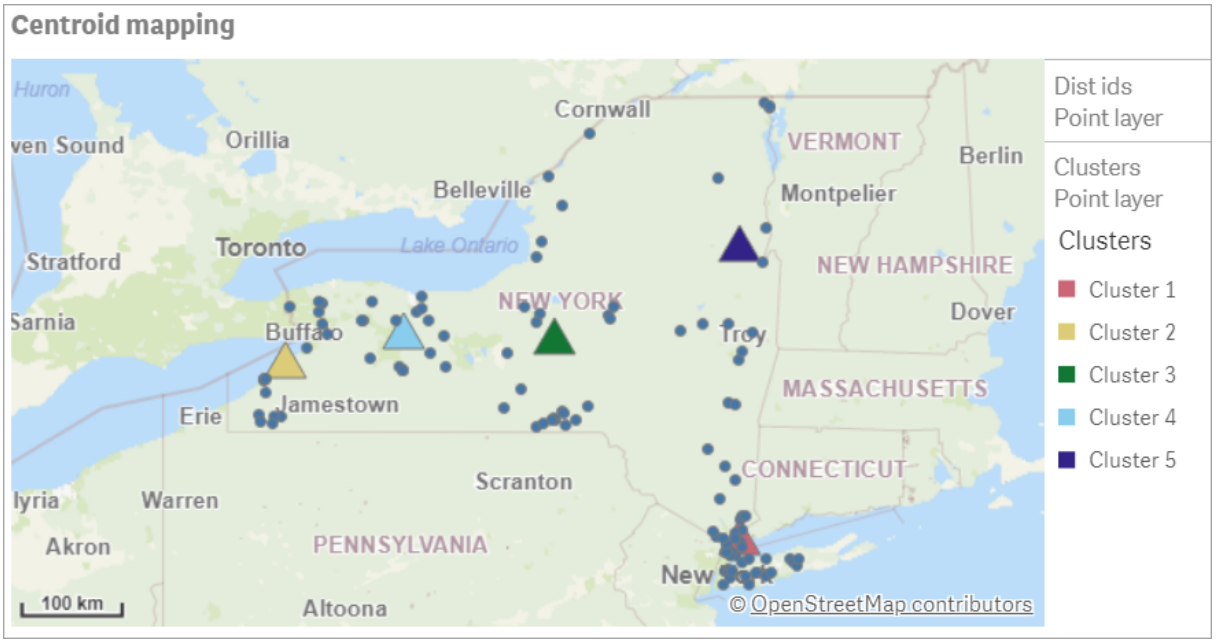
Mapping centroide

Il passaggio successivo consiste nel mappare i centroidi. Sta allo sviluppatore di app scegliere se preferisce posizionare la visualizzazione su fogli separati.

1. Una **mappa** denominata *Mapping centroide* viene trascinata sul foglio.
2. Nella sezione **Livelli**. Viene selezionato **Aggiungi livello**, quindi viene selezionato **Livello punti**.
   a. L'**Id** *campo* viene selezionato e si aggiunge l'*etichetta* **Dist ids**.
   b. Nella sezione **Posizione**, la casella di controllo per **Campi latitudine e longitudine** risulta selezionata.
   c. Per **Latitudine**, viene selezionato il campo *latitudine*.
   d. Per **Longitudine**, viene selezionato il campo *longitudine*.

   e. Nella sezione **Dimensione e forma**, viene selezionato **Bolla** per **Forma**, quindi la **Dimensione** viene ridotta in base alla preferenza sul cursore.
   f. Nella sezione **Colori**, viene selezionato **Colore singolo** e viene selezionato blu per il **Colore** e grigio per il colore **Contorno** (tali scelte rappresentano anche una preferenza).
3. Nella sezione **Livelli**, un secondo **Livello punti** viene aggiunto selezionando **Aggiungi livello** e quindi selezionando **Livello punti**.
   a. Viene inserita la seguente espressione: *=aggr(KMeans2D(vDistClusters,only(latitude),only (longitude)),id)*
   b. Viene aggiunta l'**Etichetta** *Cluster*.
   c. Nella sezione **Posizione**, la casella di controllo per **Campi latitudine e longitudine** risulta selezionata.
   d. Per la **Latitudine** che in questo caso viene tracciata lungo l'asse x, viene aggiunta la seguente espressione: *=aggr(KMeansCentroid2D(vDistClusters,0,only(latitude),only (longitude)),id)*
   e. Per la **Longitudine** che in questo caso viene tracciata lungo l'asse y, viene aggiunta la seguente espressione: *=aggr(KMeansCentroid2D(vDistClusters,1,only(latitude),only (longitude)),id)*
   f. Nella sezione **Dimensioni e forma**, viene selezionato **Triangolo** per **Forma** e le **Dimensioni** vengono ridotte sul cursore alla preferenza.
   g. Sotto **Colori e legenda**, viene selezionato **Personalizzato** per **Colori**.
   h. **Per dimensione** viene selezionato per colorare il grafico. Viene inserita la seguente espressione: *=pick(aggr(KMeans2D(vDistClusters,only(latitude),only (longitude)),id)+1,'Cluster 1','Cluster 2','Cluster 3','Cluster 4','Cluster 5')*
   i. La dimensione viene etichettata *Cluster*.
4. In **Impostazioni mappa**, viene selezionato **Adattiva** per **Proiezione**. **Metrica** viene selezionato per **Unità di misura.**

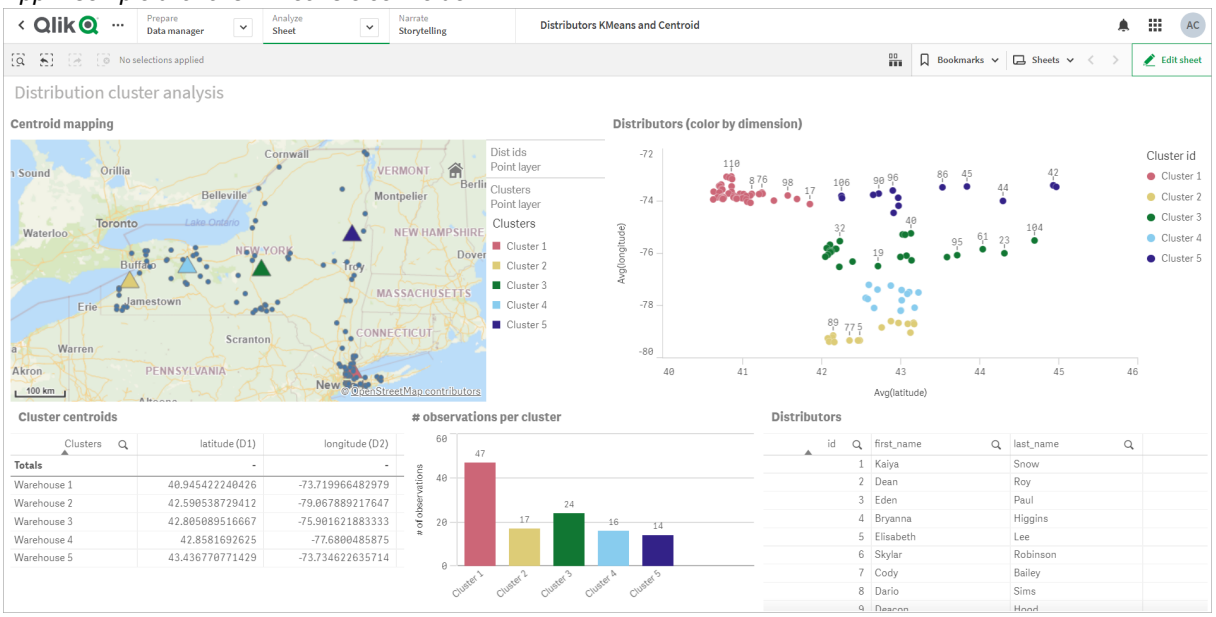
Mappa: Centroidi mappati per cluster



Conclusione

Utilizzando la funzione KMeans per questo scenario del mondo reale, i distributori sono stati segmentati in gruppi simili o cluster basati sulla somiglianza; in questo caso, la prossimità l'uno con l'altro. La funzione Centroide è stata applicata a tali cluster per identificare cinque coordinate di mappatura. Tali coordinate forniscono una posizione centrale iniziale su cui costruire o localizzare i magazzini. La funzione centroide si applica al grafico **mappa**, in modo che gli utenti delle app possano visualizzare dove si trovano i centroidi rispetto ai punti dati cluster circostanti. Le coordinate risultanti rappresentano potenziali posizioni dei magazzini che potrebbero ridurre al minimo i costi di consegna ai distributori nello stato di New York.

App: Esempio di analisi KMeans e centroide



Set di dati Distributor: Caricamento inline per l'editor caricamento dati in Qlik Sense

```
DistributorData:
Load * Inline [
id,first_name,last_name,telephone,address,city,state,zip,latitude,longitude
1,Kaiya,Snow,(716) 201-1212,6231 Tonawanda Creek Rd #APT 308,Lockport,NY,14094,43.08926,-
78.69313
2,Dean,Roy,(716) 201-1588,6884 E High St,Lockport,NY,14094,43.16245,-78.65036
3,Eden,Paul,(716) 202-4596,4647 Southwestern Blvd #APT 350,Hamburg,NY,14075,42.76003,-78.83194
4,Bryanna,Higgins,(716) 203-7041,418 Park Ave,Dunkirk,NY,14048,42.48279,-79.33088
5,Elisabeth,Lee,(716) 203-7043,36 E Courtney St,Dunkirk,NY,14048,42.48299,-79.31928
6,Skylar,Robinson,(716) 203-7166,26 Greco Ln,Dunkirk,NY,14048,42.4612095,-79.3317925
7,Cody,Bailey,(716) 203-7201,114 Lincoln Ave,Dunkirk,NY,14048,42.4801269,-79.322232
8,Dario,Sims,(408) 927-1606,N Castle Dr,Armonk,NY,10504,41.11979,-73.714864
9,Deacon,Hood,(410) 244-6221,4856 44th St,Woodside,NY,11377,40.748372,-73.905445
10,Zackery,Levy,(410) 363-8874,61 Executive Blvd,Farmingdale,NY,11735,40.7197457,-73.430239
11,Rey,Hawkins,(412) 344-8687,4585 Shimerville Rd,Clarence,NY,14031,42.972075,-78.6592452
12,Phillip,Howard,(413) 269-4049,464 Main St #101,Port Washington,NY,11050,40.8273756,-
73.7009971
13,Shirley,Tyler,(434) 985-8943,114 Glann Rd,Apalachin,NY,13732,42.0482515,-76.1229725
14,Aniyah,Jarvis,(440) 244-1808,87 N Middletown Rd,Pearl River,NY,10965,41.0629,-74.0159
15,Alayna,Woodard,(478) 335-3704,70 W Red Oak Ln,West Harrison,NY,10604,41.0162722,-73.7234926
16,Jermaine,Lambert,(508) 561-9836,24 Kellogg Rd,New Hartford,NY,13413,43.0555739,-75.2793197
17,Harper,Gibbs,(239) 466-0238,Po Box 33,Cottekill,NY,12419,41.853392,-74.106082
18,Osvaldo,Graham,(252) 246-0816,6878 Sand Hill Rd,East Syracuse,NY,13057,43.073215,-76.081448
19,Roberto,Wade,(270) 469-1211,3936 Holley Rd,Moravia,NY,13118,42.713044,-76.481227
20,Kate,Mcguire,(270) 788-3080,6451 State 64 Rte #3,Naples,NY,14512,42.707366,-77.380489
21,Dale,Andersen,(281) 480-5690,205 W Service Rd,Champlain,NY,12919,44.9645392,-73.4470831
22,Lorelai,Burch,(302) 644-2133,1 Brewster St,Glen Cove,NY,11542,40.865177,-73.633019
23,Amiyah,Flowers,(303) 223-0055,46600 Us Interstate 81 Rte,Alexandria
Bay,NY,13607,44.309626,-75.988365
24,Mckinley,Clements,(303) 918-3230,200 Summit Lake Dr,Valhalla,NY,10595,41.101145,-73.778298
25,Marc,Gibson,(607) 203-1233,25 Robinson St,Binghamton,NY,13901,42.107416,-75.901614
26,Kali,Norman,(607) 203-1400,1 Ely Park Blvd #APT 15,Binghamton,NY,13905,42.125866,-75.925026
27,Laci,Cain,(607) 203-1437,16 Zimmer Road,Kirkwood,NY,13795,42.066516,-75.792627
28,Mohammad,Perez,(607) 203-1652,71 Endicott Ave #APT 12,Johnson City,NY,13790,42.111894,-
75.952187
29,Izabelle,Pham,(607) 204-0392,434 State 369 Rte,Port Crane,NY,13833,42.185838,-75.823074
30,Kiley,Mays,(607) 204-0870,244 Ballyhack Rd #14,Port Crane,NY,13833,42.175612,-75.814917
31,Peter,Trevino,(607) 205-1374,125 Melbourne St.,Vestal,NY,13850,42.080254,-76.051124
32,Ani,Francis,(607) 208-4067,48 Caswell St,Afton,NY,13730,42.232065,-75.525674
33,Jared,Sheppard,(716) 386-3002,4709 430th Rte,Bemus Point,NY,14712,42.162175,-79.39176
34,Dulce,Atkinson,(914) 576-2266,501 Pelham Rd,New Rochelle,NY,10805,40.895449,-73.782602
35,Jayla,Beasley,(716) 526-1054,5010 474th Rte,Ashville,NY,14710,42.096859,-79.375561
36,Dane,Donovan,(718) 545-3732,5014 31st Ave,Woodside,NY,11377,40.756967,-73.909506
37,Brendon,Clay,(585) 322-7780,133 Cummings Ave,Gainesville,NY,14066,42.664309,-78.085651
38,Asia,Nunez,(718) 426-1472,2407 Gilmore ,East Elmhurst,NY,11369,40.766662,-73.869185
39,Dawson,Odonnell,(718) 342-2179,5019 H Ave,Brooklyn,NY,11234,40.633245,-73.927591
40,Kyle,Collins,(315) 733-7078,502 Rockhaven Rd,Utica,NY,13502,43.129184,-75.226726
41,Eliza,Hardin,(315) 331-8072,502 Sladen Place,West Point,NY,10996,41.3993,-73.973003
42,Kasen,Klein,(518) 298-4581,2407 Lake Shore Rd,Chazy,NY,12921,44.925561,-73.387373
43,Reuben,Bradford,(518) 298-4581,33 Lake Flats Dr,Champlain,NY,12919,44.928092,-73.387884
44,Henry,Grimes,(518) 523-3990,2407 Main St,Lake Placid,NY,12946,44.291487,-73.98474
45,Kyan,Livingston,(518) 585-7364,241 Alexandria Ave,Ticonderoga,NY,12883,43.836553,-73.43155
```

```
46,Kaitlyn,Short,(516) 678-3189,241 Chance Dr,Oceanside,NY,11572,40.638534,-73.63079
47,Damaris,Jacobs,(914) 664-5331,241 Claremont Ave,Mount Vernon,NY,10552,40.919852,-73.827848
48,Alivia,Schroeder,(315) 469-4473,241 Lafayette Rd,Syracuse,NY,13205,42.996446,-76.12957
49,Bridget,Strong,(315) 298-4355,241 Maltby Rd,Pulaski,NY,13142,43.584966,-76.136317
50,Francis,Lee,(585) 201-7021,166 Ross St,Batavia,NY,14020,43.0031502,-78.17487
51,Makaila,Phelps,(585) 201-7422,58 S Main St,Batavia,NY,14020,42.99941,-78.1939285
52,Jazlynn,Stephens,(585) 203-1087,1 Sinclair Dr,Pittsford,NY,14534,43.084157,-77.545452
53,Ryann,Randolph,(585) 203-1519,331 Eaglehead Rd,East Rochester,NY,14445,43.10785,-77.475552
54,Rosa,Baker,(585) 204-4011,42 Ossian St,Dansville,NY,14437,42.560761,-77.70088
55,Marcel,Barry,(585) 204-4013,42 Jefferson St,Dansville,NY,14437,42.557735,-77.702983
56,Dennis,Schmitt,(585) 204-4061,750 Dansville Mount Morris Rd,Dansville,NY,14437,42.584458,-
77.741648
57,Cassandra,Kim,(585) 204-4138,3 Perine Ave APT1,Dansville,NY,14437,42.562865,-77.69661
58,Kolton,Jacobson,(585) 206-5047,4925 Upper Holly Rd,Holley,NY,14470,43.175957,-78.074465
59,Nathanael,Donovan,(718) 393-3501,9604 57th Ave,Corona,NY,11373,40.736077,-73.864858
60,Robert,Frazier,(718) 271-3067,300 56th Ave,Corona,NY,11373,40.735304,-73.873997
61,Jessie,Mora,(315) 405-8991,9607 Forsyth Loop,Watertown,NY,13603,44.036466,-75.833437
62,Martha,Rollins,(347) 242-2642,22 Main St,Corona,NY,11373,40.757727,-73.829331
63,Emely,Townsend,(718) 699-0751,60 Sanford Ave,Corona,NY,11373,40.755466,-73.831029
64,Kylie,Cooley,(347) 561-7149,9608 95th Ave,Ozone Park,NY,11416,40.687564,-73.845715
65,Wendy,Cameron,(585) 571-4185,9608 Union St,Scottsville,NY,14546,43.013327,-77.7907839
66,Kayley,Peterson,(718) 654-5027,961 E 230th St,Bronx,NY,10466,40.889275,-73.850555
67,Camden,Ochoa,(718) 760-8699,59 Vark St,Yonkers,NY,10701,40.929322,-73.89957
68,Priscilla,Castillo,(910) 326-7233,9359 Elm St,Chadwicks,NY,13319,43.024902,-75.26886
69,Dana,Schultz,(913) 322-4580,99 Washington Ave,Hastings on Hudson,NY,10706,40.99265,-
73.879748
70,Blaze,Medina,(914) 207-0015,60 Elliott Ave,Yonkers,NY,10705,40.921498,-73.896682
71,Finnegan,Tucker,(914) 207-0015,90 Hillside Drive,Yonkers,NY,10705,40.922514,-73.892911
72,Pranav,Palmer,(914) 214-8376,5 Bruce Ave,Harrison,NY,10528,40.970916,-73.711493
73,Kolten,Wong,(914) 218-8268,70 Barker St,Mount Kisco,NY,10549,41.211993,-73.723202
74,Jasiah,Vazquez,(914) 231-5199,30 Broadway,Dobbs Ferry,NY,10522,41.004629,-73.879825
75,Lamar,Pierce,(914) 232-0380,68 Ridge Rd,Katonah,NY,10536,41.256662,-73.707964
76,Carla,Coffey,(914) 232-0469,197 Beaver Dam Rd,Katonah,NY,10536,41.247934,-73.664363
77,Brooklynn,Harmon,(716) 595-3227,8084 Glasgow Rd,Cassadega,NY,14718,42.353861,-79.329558
78,Raquel,Hodges,(585) 398-8125,809 County Road ,Victor,NY,14564,43.011745,-77.398806
79,Jerimiah,Gardner,(585) 787-9127,809 Houston Rd,Webster,NY,14580,43.224204,-77.491353
80,Clarence,Hammond,(720) 746-1619,809 Pierpont Ave,Piermont,NY,10968,41.0491181,-73.918622
81,Rhys,Gill,(518) 427-7887,81 Columbia St,Albany,NY,12210,42.652824,-73.752096
82,Edith,Parrish,(845) 452-7621,81 Glenwood Ave,Poughkeepsie,NY,12603,41.691058,-73.910829
83,Kobe,Mcintosh,(845) 371-1101,81 Heitman Dr,Spring Valley,NY,10977,41.103227,-74.054396
84,Ayden,Waters,(516) 796-2722,81 Kingfisher Rd,Levittown,NY,11756,40.738939,-73.52826
85,Francis,Rogers,(631) 427-7728,81 Knollwood Ave,Huntington,NY,11743,40.864905,-73.426107
86,Jaden,Landry,(716) 496-4038,12839 39th Rte,Chaffee,NY,14030,43.527396,-73.462786
87,Giancarlo,Campos,(518) 885-5717,1284 Saratoga Rd,Ballston Spa,NY,12020,42.968594,-73.862847
88,Eduardo,Contreras,(716) 285-8987,1285 Saunders Sett Rd,Niagara Falls,NY,14305,43.122963,-
79.029274
89,Gabriela,Davidson,(716) 267-3195,1286 Mee Rd,Falconer,NY,14733,42.147339,-79.137976
90,Evangeline,Case,(518) 272-9435,1287 2nd Ave,Watervliet,NY,12189,42.723132,-73.703818
91,Tyrone,Ellison,(518) 843-4691,1287 Midline Rd,Amsterdam,NY,12010,42.9730876,-74.1700608
92,Bryce,Bass,(518) 943-9549,1288 Leeds Athens Rd,Athens,NY,12015,42.259381,-73.876897
93,Londyn,Butler,(518) 922-7095,129 Argersinger Rd,Fultonville,NY,12072,42.910969,-74.441917
94,Graham,Becker,(607) 655-1318,129 Baker Rd,Windsor,NY,13865,42.107271,-75.66408
95,Rolando,Fitzgerald,(315) 465-4166,17164 County 90 Rte,Mannsville,NY,13661,43.713443,-
76.06232
96,Grant,Hoover,(518) 692-8363,1718 County 113 Rte,Schaghticote,NY,12154,42.900648,-73.585036
97,Mark,Goodwin,(631) 584-6761,172 Cambon Ave,Saint James,NY,11780,40.871152,-73.146032
```

```
98,Deacon,Cantu,(845) 221-7940,172 Carpenter Rd,Hopewell Junction,NY,12533,41.57388,-73.77609
99,Tristian,Walsh,(516) 997-4750,172 E Cabot Ln,Westbury,NY,11590,40.7480397,-73.54819
100,Abram,Alexander,(631) 588-3817,172 Lorenzo Cir,Ronkonkoma,NY,11779,40.837123,-73.09367
101,Lesly,Bush,(516) 489-3791,172 Nassau Blvd,Garden City,NY,11530,40.71147,-73.660753
102,Pamela,Espinoza,(716) 201-1520,172 Niagara St ,Lockport,NY,14094,43.169871,-78.70093
103,Bryanna,Newton,(914) 328-4332,172 Warren Ave,White Plains,NY,10603,41.047207,-73.79572
104,Marcelo,Schmitt,(315) 393-4432,319 Mansion Ave,Ogdensburg,NY,13669,44.690246,-75.49992
105,Layton,Valenzuela,(631) 676-2113,319 Singingwood Dr,Holbrook,NY,11741,40.801391,-73.058993
106,Roderick,Rocha,(518) 671-6037,319 Warren St,Hudson,NY,12534,42.252527,-73.790629
107,Camryn,Terrell,(315) 635-1680,3192 Olive Dr,Baldinsville,NY,13027,43.136843,-76.260303
108,Summer,Callahan,(585) 394-4195,3192 Smith Road,Canandaigua,NY,14424,42.875457,-77.228039
109,Pierre,Novak,(716) 665-2524,3194 Falconer Kimball Stand Rd,Falconer,NY,14733,42.138439,-
79.211091
110,Kennedi,Fry,(315) 543-2301,32 College Rd,Selden,NY,11784,40.861624,-73.04757
111,Wyatt,Pruitt,(716) 681-4042,277 Ransom Rd,Lancaster ,NY,14086,42.87702,-78.591302
112,Lilly,Jensen,(631) 841-0859,2772 Schliegel Blvd,Amityville,NY,11701,40.708021,-73.413015
113,Tristin,Hardin,(631) 920-0927,278 Fulton Street,West Babylon,NY,11704,40.733578,-73.357321
114,Tanya,Stafford,(716) 484-0771,278 Sampson St,Jamestown,NY,14701,42.0797,-79.247805
115,Paris,Cordova,(607) 589-4857,278 Washburn Rd,Spencer,NY,14883,42.225046,-76.510257
116,Alfonso,Morse,(718) 359-5582,200 Colden St,Flushing,NY,11355,40.750403,-73.822752
117,Maurice,Hooper,(315) 595-6694,4435 Italy Hill Rd,Branchport,NY,14418,42.597957,-77.199267
118,Iris,Wolf,(607) 539-7288,444 Harford Rd,Brooktondale,NY,14817,42.392164,-76.30756
];
```

KMeans2D - funzione per grafici

KMeans2D() valuta le righe del grafico applicando il clustering K-means, e per ciascuna riga del grafico visualizza l'id cluster del cluster a cui è stato assegnato questo punto dati. Le colonne utilizzate dall'algoritmo di clustering sono determinate rispettivamente dai parametri coordinate_1 e coordinate_2. Sono entrambe aggregazioni. Il numero di cluster creati è determinato dal parametro num_clusters. I dati possono essere normalizzati in via opzionale dal parametro norm.

KMeans2D restituisce un valore per punto dati. Il valore restituito è duale ed è un valore intero corrispondente al cluster a cui ciascun punto dati è stato assegnato.

Sintassi:

```
KMeans2D(num_clusters, coordinate_1, coordinate_2 [, norm])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
num_clusters	Intero che specifica il numero di cluster.
coordinate_1	L'aggregazione che calcola la prima coordinata, in genere l'asse x del grafico a dispersione che può essere effettuato dal grafico. Il parametro aggiuntivo, coordinate_2, calcola la seconda coordinata.

Argomento	Descrizione
norm	Il metodo di normalizzazione opzionale applicato alle serie di dati prima del clustering K-means. Possibili valori: 0 o 'nessuno' per l'assenza di normalizzazione 1 o 'zscore' per la normalizzazione z-score 2 o 'minmax' per la normalizzazione min-max Se non viene fornito alcun parametro o se il parametro fornito risulta errato, non viene applicata alcuna normalizzazione. Z-score normalizza i dati in base alla deviazione standard e media della funzionalità. Z-score non assicura che ciascuna funzionalità abbia la stessa scala, ma rappresenta un approccio migliore a min-max quando si ha a che fare con outlier. La normalizzazione min-max assicura che le funzionalità abbiano la stessa scala prelevando i valori minimo e massimo di ciascuna di esse e ricalcolando ciascun datapoint.

Esempio: espressione del grafico
In questo esempio, creiamo un grafico a dispersione usando la serie di dati *Iris*, quindi utilizziamo KMeans per colorare i dati per espressione.

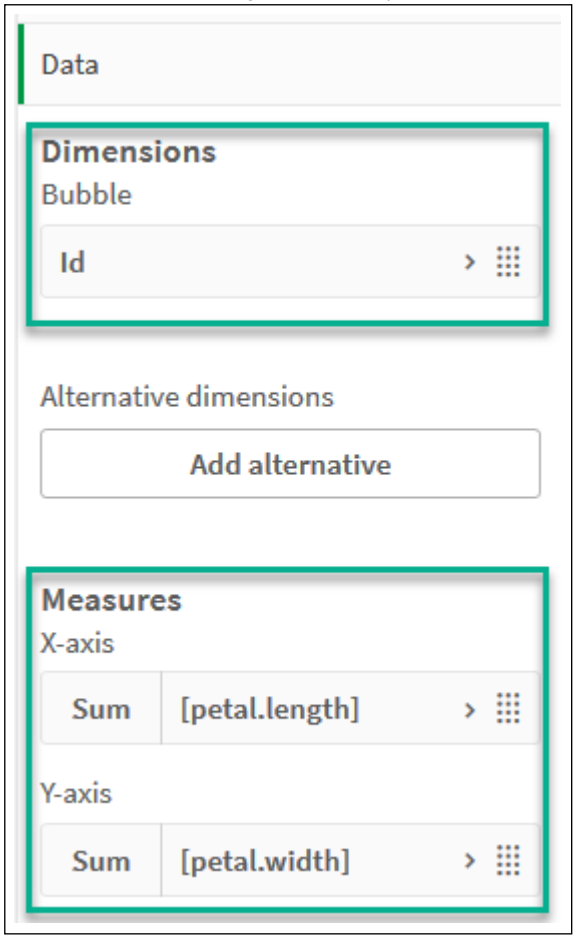
Creiamo inoltre una variabile per l'argomento *num_clusters*, quindi utilizziamo una casella di input variabile per modificare il numero di cluster.

La serie di dati *Iris* è disponibile pubblicamente in una serie di formati. I dati sono stati forniti come tabella inline da caricare usando l'editor caricamento dati in Qlik Sense. Notare che è stata aggiunta una colonna *Id* alla tabella dati per questo esempio.

Dopo il caricamento dei dati in Qlik Sense, è possibile compiere le seguenti operazioni:

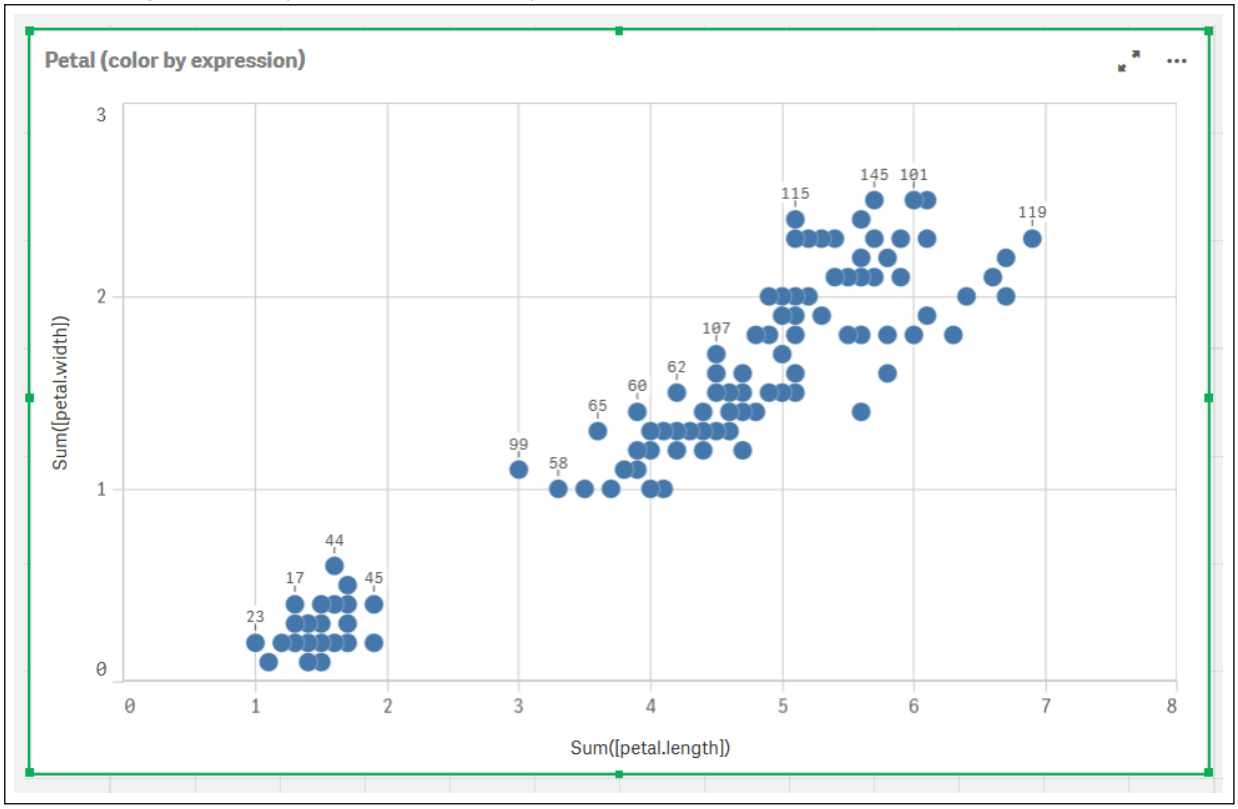
1. Trascinare un **Grafico a dispersione** in un nuovo foglio. Denominare il grafico *Petal (colore per espressione)*.
2. Creare una variabile per specificare il numero di cluster. Per la variabile **Nome**, inserire *KmeansPetalClusters*. Per la variabile **Definizione**, inserire *=2*.
3. Configurare **Dati** per il grafico:
    i. Sotto **Dimensioni**, scegliere *id* per il campo per **Bolla**. Inserire Id cluster per l'etichetta.
    ii. Sotto **Misure**, scegliere *Sum([petal.length])* per l'espressione per **asse X**.
    iii. Sotto **Misure**, scegliere *Sum([petal.width])* per l'espressione per **asse Y**.

Impostazioni dati per il grafico Petal (colore per espressione)



I punti dati vengono riportati sul grafico.

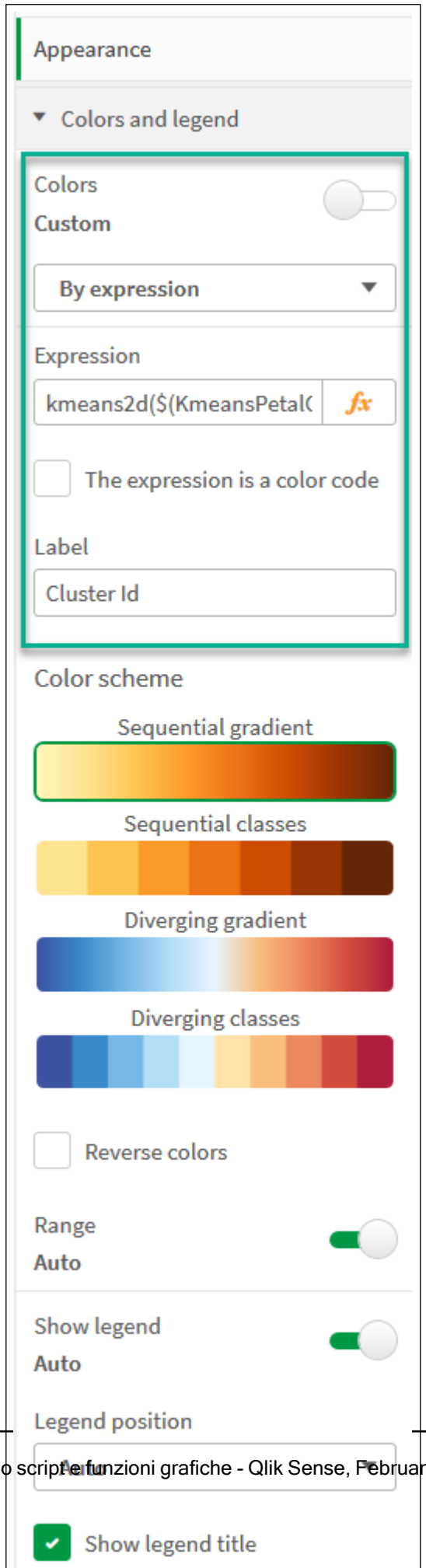
Punti dati sul grafico Petal (colore per espressione)



4. Configurare **Aspetto** per il grafico:
    i. Sotto **Colori e legenda**, scegliere **Personalizzato** per **Colori**.
    ii. Scegliere di colorare il grafico **Per espressione**.
    iii. Inserire quanto segue per **Espressione**: *kmeans2d($(KmeansPetalClusters), Sum ([petal.length]), Sum([petal.width]))*
    Notare che *KmeansPetalClusters* è la variabile che impostiamo a *2*.
    In alternativa, inserire quanto segue: *kmeans2d(2, Sum([petal.length]), Sum([petal.width]))*
    iv. Deselezionare la casella di controllo per **L'espressione è un codice cromatico**.

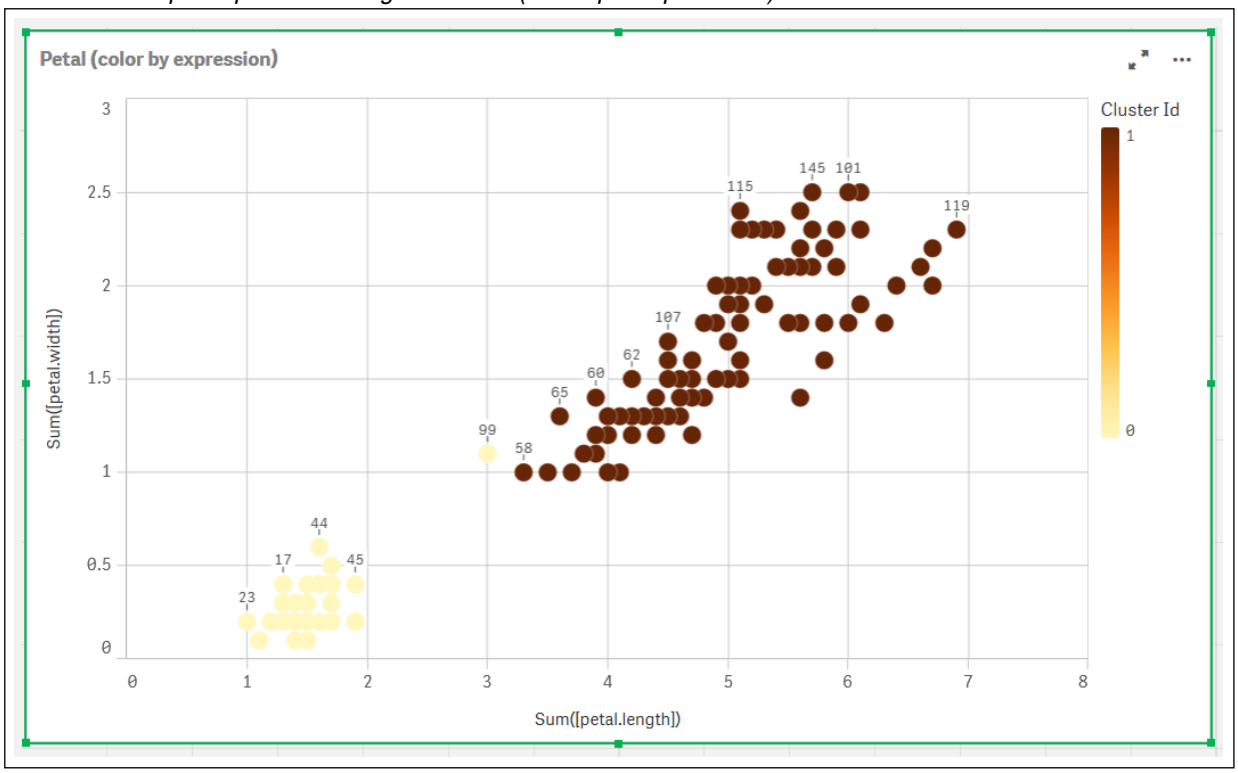
v. Inserire quanto segue per **Etichetta**: *Id cluster*

Impostazioni aspetto per il grafico Petal (colore per espressione)


Appearance
Colors and legend
Colors
Custom
By expression
Expression
kmeans2d($(KmeansPetal(
The expression is a color code
Label
Cluster Id
Color scheme
Sequential gradient
Sequential classes
Diverging gradient
Diverging classes
Reverse colors
Range
Auto
Show legend
Auto
Legend position
Auto
Show legend title

I due cluster sul grafico sono colorati in base all'espressione KMeans.

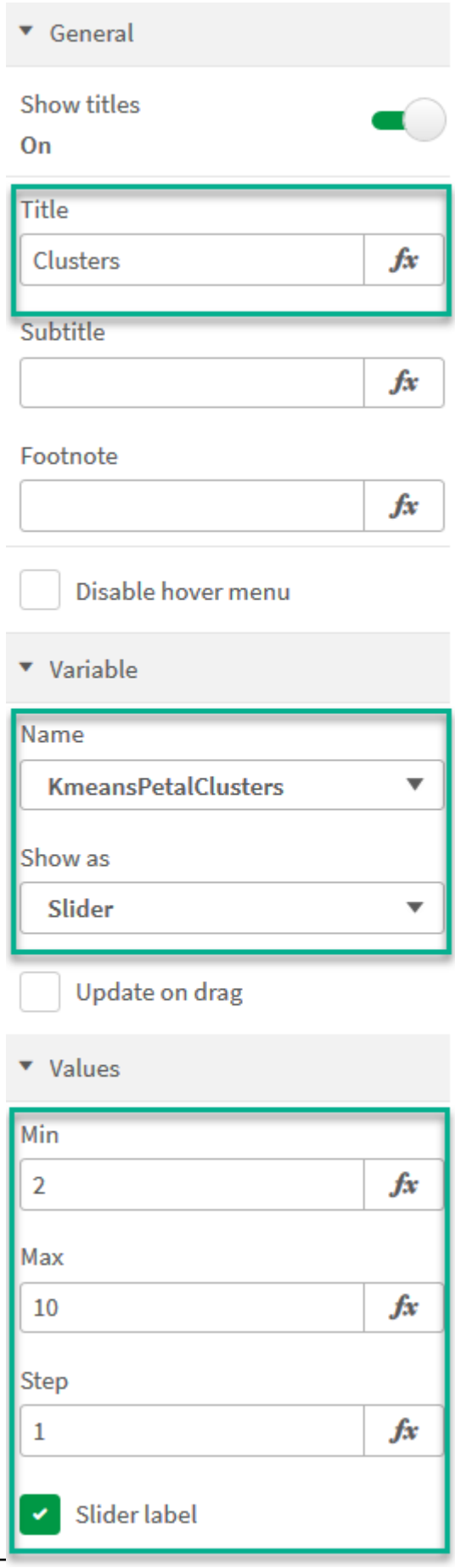
Cluster colorati per espressione sul grafico Petal (colore per espressione)



5. Aggiungere una casella **Input variabile** per il numero di cluster.
    i. Sotto **Oggetti personalizzati** nel pannello **Asset**, scegliere **Qlik Dashboard bundle**. Se non si ha accesso al dashboard bundle, è comunque possibile modificare il numero di cluster utilizzando la variabile che abbiamo creato, oppure direttamente come intero nell'espressione.
    ii. Trascinare una casella **Input variabile** sul foglio.
    iii. Sotto **Aspetto**, fare clic su **Generale**.
    iv. Inserire quanto segue per **Titolo**: *Cluster*
    v. Fare clic su **Variabile**.
    vi. Scegliere la seguente variabile per **Nome**: *KmeansPetalClusters*.
    vii. Scegliere **Cursore** per **Mostra come**.

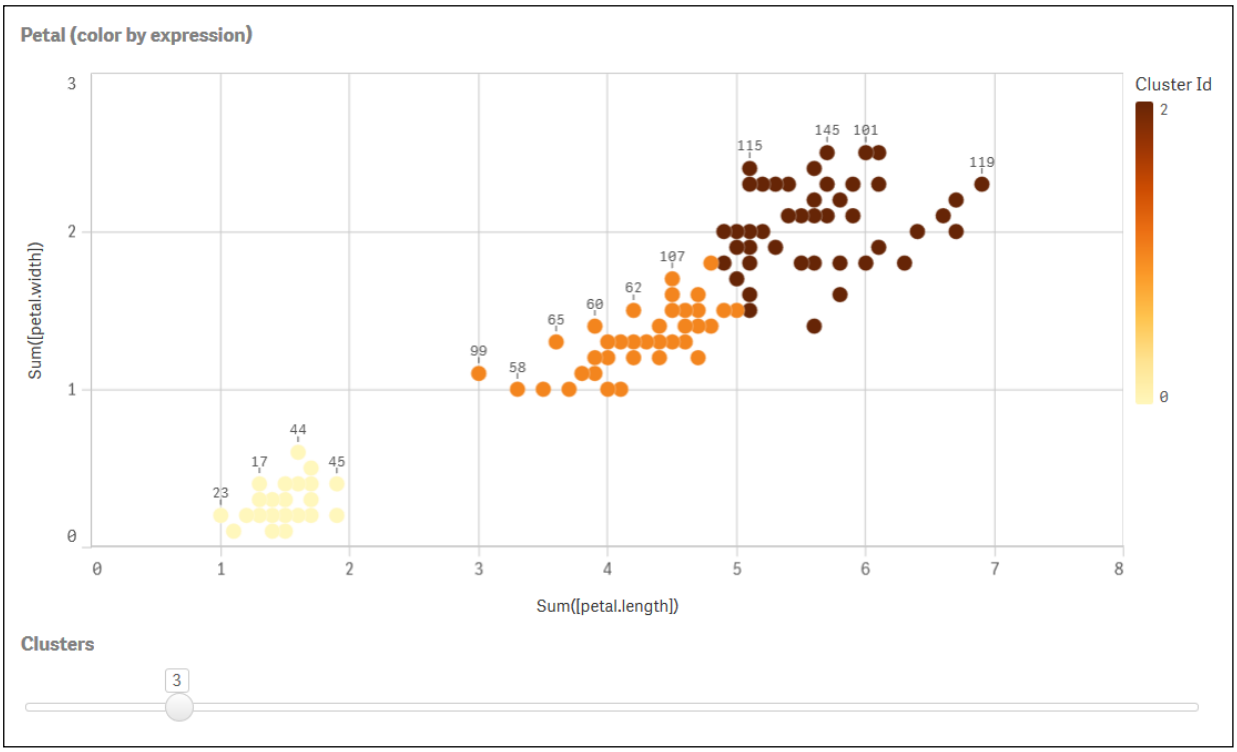
viii. Scegliere **Valori**, quindi configurare le impostazioni come richiesto.

Aspetto per la casella di input variabile Cluster


General
Show titles
On
Title
Clusters
Subtitle
Footnote
Disable hover menu
Variable
Name
KmeansPetalClusters
Show as
Slider
Update on drag
Values
Min
2
Max
10
Step
1
Slider label

Al termine della modifica, è possibile modificare il numero di cluster usando il cursore nella casella di input variabile *Cluster*.

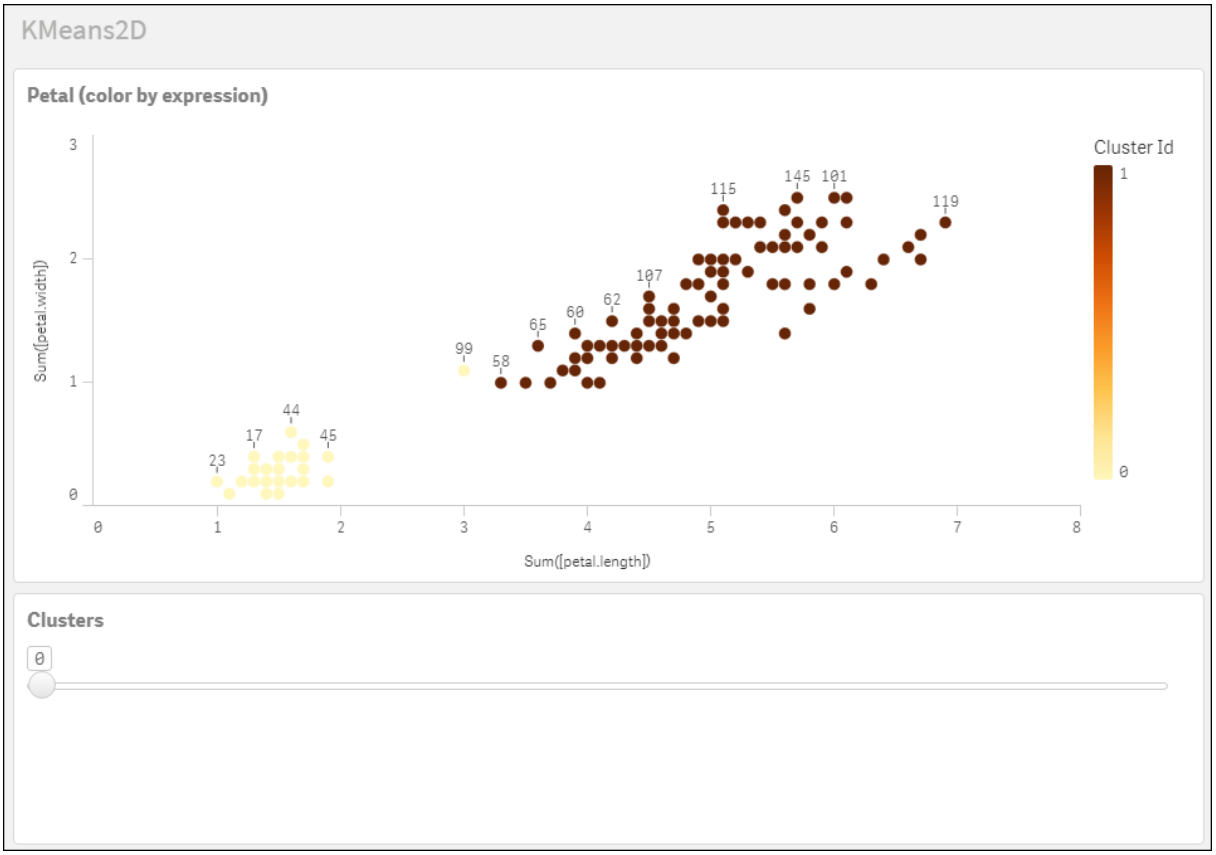
Cluster colorati per espressione sul grafico Petal (colore per espressione)



Clustering automatico

Le funzioni **KMeans** supportano il clustering automatico mediante un metodo chiamato differenza di profondità (DeD, Depth Difference). Quando un utente imposta lo 0 per il numero di cluster, viene determinato un numero ottimale di cluster per tale set di dati. Notare che mentre un valore intero per il numero di cluster (*k*) non viene restituito esplicitamente, viene calcolato all'interno dell'algoritmo KMeans. Ad esempio, se viene specificato 0 nella funzione del valore di *KmeansPetalClusters* o se viene impostato mediante una casella di input variabile, le assegnazioni cluster vengono calcolate automaticamente per il set di dati in base a un numero ottimale di cluster.

Il metodo di differenza di profondità KMeans determina il numero ottimale di cluster quando (k) viene impostato a 0



Serie di dati Iris: Caricamento inline per l'editor caricamento dati in Qlik Sense

```
IrisData:
Load * Inline [
sepal.length, sepal.width, petal.length, petal.width, variety, id
5.1, 3.5, 1.4, 0.2, Setosa, 1
4.9, 3, 1.4, 0.2, Setosa, 2
4.7, 3.2, 1.3, 0.2, Setosa, 3
4.6, 3.1, 1.5, 0.2, Setosa, 4
5, 3.6, 1.4, 0.2, Setosa, 5
5.4, 3.9, 1.7, 0.4, Setosa, 6
4.6, 3.4, 1.4, 0.3, Setosa, 7
5, 3.4, 1.5, 0.2, Setosa, 8
4.4, 2.9, 1.4, 0.2, Setosa, 9
4.9, 3.1, 1.5, 0.1, Setosa, 10
5.4, 3.7, 1.5, 0.2, Setosa, 11
4.8, 3.4, 1.6, 0.2, Setosa, 12
4.8, 3, 1.4, 0.1, Setosa, 13
4.3, 3, 1.1, 0.1, Setosa, 14
5.8, 4, 1.2, 0.2, Setosa, 15
5.7, 4.4, 1.5, 0.4, Setosa, 16
5.4, 3.9, 1.3, 0.4, Setosa, 17
5.1, 3.5, 1.4, 0.3, Setosa, 18
5.7, 3.8, 1.7, 0.3, Setosa, 19
5.1, 3.8, 1.5, 0.3, Setosa, 20
5.4, 3.4, 1.7, 0.2, Setosa, 21
```

```
5.1, 3.7, 1.5, 0.4, Setosa, 22
4.6, 3.6, 1, 0.2, Setosa, 23
5.1, 3.3, 1.7, 0.5, Setosa, 24
4.8, 3.4, 1.9, 0.2, Setosa, 25
5, 3, 1.6, 0.2, Setosa, 26
5, 3.4, 1.6, 0.4, Setosa, 27
5.2, 3.5, 1.5, 0.2, Setosa, 28
5.2, 3.4, 1.4, 0.2, Setosa, 29
4.7, 3.2, 1.6, 0.2, Setosa, 30
4.8, 3.1, 1.6, 0.2, Setosa, 31
5.4, 3.4, 1.5, 0.4, Setosa, 32
5.2, 4.1, 1.5, 0.1, Setosa, 33
5.5, 4.2, 1.4, 0.2, Setosa, 34
4.9, 3.1, 1.5, 0.1, Setosa, 35
5, 3.2, 1.2, 0.2, Setosa, 36
5.5, 3.5, 1.3, 0.2, Setosa, 37
4.9, 3.1, 1.5, 0.1, Setosa, 38
4.4, 3, 1.3, 0.2, Setosa, 39
5.1, 3.4, 1.5, 0.2, Setosa, 40
5, 3.5, 1.3, 0.3, Setosa, 41
4.5, 2.3, 1.3, 0.3, Setosa, 42
4.4, 3.2, 1.3, 0.2, Setosa, 43
5, 3.5, 1.6, 0.6, Setosa, 44
5.1, 3.8, 1.9, 0.4, Setosa, 45
4.8, 3, 1.4, 0.3, Setosa, 46
5.1, 3.8, 1.6, 0.2, Setosa, 47
4.6, 3.2, 1.4, 0.2, Setosa, 48
5.3, 3.7, 1.5, 0.2, Setosa, 49
5, 3.3, 1.4, 0.2, Setosa, 50
7, 3.2, 4.7, 1.4, Versicolor, 51
6.4, 3.2, 4.5, 1.5, Versicolor, 52
6.9, 3.1, 4.9, 1.5, Versicolor, 53
5.5, 2.3, 4, 1.3, Versicolor, 54
6.5, 2.8, 4.6, 1.5, Versicolor, 55
5.7, 2.8, 4.5, 1.3, Versicolor, 56
6.3, 3.3, 4.7, 1.6, Versicolor, 57
4.9, 2.4, 3.3, 1, Versicolor, 58
6.6, 2.9, 4.6, 1.3, Versicolor, 59
5.2, 2.7, 3.9, 1.4, Versicolor, 60
5, 2, 3.5, 1, Versicolor, 61
5.9, 3, 4.2, 1.5, Versicolor, 62
6, 2.2, 4, 1, Versicolor, 63
6.1, 2.9, 4.7, 1.4, Versicolor, 64
5.6, 2.9, 3.6, 1.3, Versicolor, 65
6.7, 3.1, 4.4, 1.4, Versicolor, 66
5.6, 3, 4.5, 1.5, Versicolor, 67
5.8, 2.7, 4.1, 1, Versicolor, 68
6.2, 2.2, 4.5, 1.5, Versicolor, 69
5.6, 2.5, 3.9, 1.1, Versicolor, 70
5.9, 3.2, 4.8, 1.8, Versicolor, 71
6.1, 2.8, 4, 1.3, Versicolor, 72
6.3, 2.5, 4.9, 1.5, Versicolor, 73
6.1, 2.8, 4.7, 1.2, Versicolor, 74
6.4, 2.9, 4.3, 1.3, Versicolor, 75
6.6, 3, 4.4, 1.4, Versicolor, 76
```

```
6.8, 2.8, 4.8, 1.4, Versicolor, 77
6.7, 3, 5, 1.7, Versicolor, 78
6, 2.9, 4.5, 1.5, Versicolor, 79
5.7, 2.6, 3.5, 1, Versicolor, 80
5.5, 2.4, 3.8, 1.1, Versicolor, 81
5.5, 2.4, 3.7, 1, Versicolor, 82
5.8, 2.7, 3.9, 1.2, Versicolor, 83
6, 2.7, 5.1, 1.6, Versicolor, 84
5.4, 3, 4.5, 1.5, Versicolor, 85
6, 3.4, 4.5, 1.6, Versicolor, 86
6.7, 3.1, 4.7, 1.5, Versicolor, 87
6.3, 2.3, 4.4, 1.3, Versicolor, 88
5.6, 3, 4.1, 1.3, Versicolor, 89
5.5, 2.5, 4, 1.3, Versicolor, 90
5.5, 2.6, 4.4, 1.2, Versicolor, 91
6.1, 3, 4.6, 1.4, Versicolor, 92
5.8, 2.6, 4, 1.2, Versicolor, 93
5, 2.3, 3.3, 1, Versicolor, 94
5.6, 2.7, 4.2, 1.3, Versicolor, 95
5.7, 3, 4.2, 1.2, Versicolor, 96
5.7, 2.9, 4.2, 1.3, Versicolor, 97
6.2, 2.9, 4.3, 1.3, Versicolor, 98
5.1, 2.5, 3, 1.1, Versicolor, 99
5.7, 2.8, 4.1, 1.3, Versicolor, 100
6.3, 3.3, 6, 2.5, Virginica, 101
5.8, 2.7, 5.1, 1.9, Virginica, 102
7.1, 3, 5.9, 2.1, Virginica, 103
6.3, 2.9, 5.6, 1.8, Virginica, 104
6.5, 3, 5.8, 2.2, Virginica, 105
7.6, 3, 6.6, 2.1, Virginica, 106
4.9, 2.5, 4.5, 1.7, Virginica, 107
7.3, 2.9, 6.3, 1.8, Virginica, 108
6.7, 2.5, 5.8, 1.8, Virginica, 109
7.2, 3.6, 6.1, 2.5, Virginica, 110
6.5, 3.2, 5.1, 2, Virginica, 111
6.4, 2.7, 5.3, 1.9, Virginica, 112
6.8, 3, 5.5, 2.1, Virginica, 113
5.7, 2.5, 5, 2, Virginica, 114
5.8, 2.8, 5.1, 2.4, Virginica, 115
6.4, 3.2, 5.3, 2.3, Virginica, 116
6.5, 3, 5.5, 1.8, Virginica, 117
7.7, 3.8, 6.7, 2.2, Virginica, 118
7.7, 2.6, 6.9, 2.3, Virginica, 119
6, 2.2, 5, 1.5, Virginica, 120
6.9, 3.2, 5.7, 2.3, Virginica, 121
5.6, 2.8, 4.9, 2, Virginica, 122
7.7, 2.8, 6.7, 2, Virginica, 123
6.3, 2.7, 4.9, 1.8, Virginica, 124
6.7, 3.3, 5.7, 2.1, Virginica, 125
7.2, 3.2, 6, 1.8, Virginica, 126
6.2, 2.8, 4.8, 1.8, Virginica, 127
6.1, 3, 4.9, 1.8, Virginica, 128
6.4, 2.8, 5.6, 2.1, Virginica, 129
7.2, 3, 5.8, 1.6, Virginica, 130
7.4, 2.8, 6.1, 1.9, Virginica, 131
```

```
7.9, 3.8, 6.4, 2, Virginica, 132
6.4, 2.8, 5.6, 2.2, Virginica, 133
6.3, 2.8, 5.1, 1.5, Virginica, 134
6.1, 2.6, 5.6, 1.4, Virginica, 135
7.7, 3, 6.1, 2.3, Virginica, 136
6.3, 3.4, 5.6, 2.4, Virginica, 137
6.4, 3.1, 5.5, 1.8, Virginica, 138
6, 3, 4.8, 1.8, Virginica, 139
6.9, 3.1, 5.4, 2.1, Virginica, 140
6.7, 3.1, 5.6, 2.4, Virginica, 141
6.9, 3.1, 5.1, 2.3, Virginica, 142
5.8, 2.7, 5.1, 1.9, Virginica, 143
6.8, 3.2, 5.9, 2.3, Virginica, 144
6.7, 3.3, 5.7, 2.5, Virginica, 145
6.7, 3, 5.2, 2.3, Virginica, 146
6.3, 2.5, 5, 1.9, Virginica, 147
6.5, 3, 5.2, 2, Virginica, 148
6.2, 3.4, 5.4, 2.3, Virginica, 149
5.9, 3, 5.1, 1.8, Virginica, 150
];
```

KMeansND - funzione per grafici

KMeansND() valuta le righe del grafico applicando il clustering K-means, e per ciascuna riga del grafico visualizza l'id cluster del cluster a cui è stato assegnato questo punto dati. Le colonne utilizzate dall'algoritmo di clustering sono determinate dai parametri coordinate_1, coordinate_2, ecc. fino a n colonne. Sono tutte aggregazioni. Il numero di cluster creati è determinato dal parametro num_clusters.

KMeansND restituisce un valore per punto dati. Il valore restituito è duale ed è un valore intero corrispondente al cluster a cui ciascun punto dati è stato assegnato.

Sintassi:

```
KMeansND(num_clusters, num_iter, coordinate_1, coordinate_2 [,coordinate_3 [,...]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
num_clusters	Intero che specifica il numero di cluster.
num_iter	Il numero di ripetizioni di clustering con centri cluster reinizializzati.
coordinate_1	L'aggregazione che calcola la prima coordinata, in genere l'asse x (di un grafico a dispersione che può essere effettuato dal grafico). I parametri aggiuntivi calcolano le seconde, terze e quarte coordinate, ecc.

Esempio: espressione del grafico
In questo esempio, creiamo un grafico a dispersione usando la serie di dati *Iris*, quindi utilizziamo KMeans per colorare i dati per espressione.

Creiamo inoltre una variabile per l'argomento *num_clusters*, quindi utilizziamo una casella di input variabile per modificare il numero di cluster.

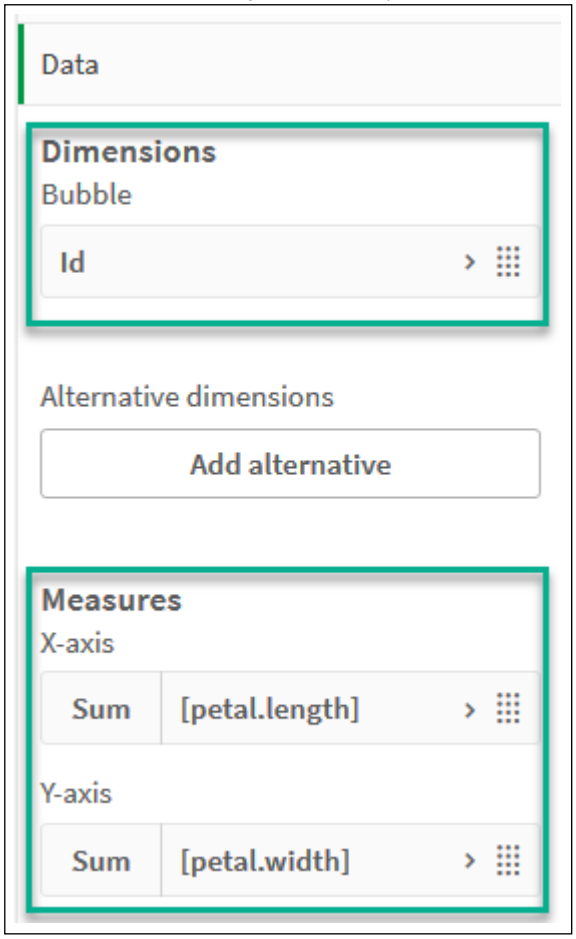
Inoltre, creiamo una variabile per l'argomento *num_iter*, quindi utilizziamo una seconda casella di input variabile per modificare il numero di ripetizioni.

La serie di dati *Iris* è disponibile pubblicamente in una serie di formati. I dati sono stati forniti come tabella inline da caricare usando l'editor caricamento dati in Qlik Sense. Notare che è stata aggiunta una colonna *Id* alla tabella dati per questo esempio.

Dopo il caricamento dei dati in Qlik Sense, è possibile compiere le seguenti operazioni:

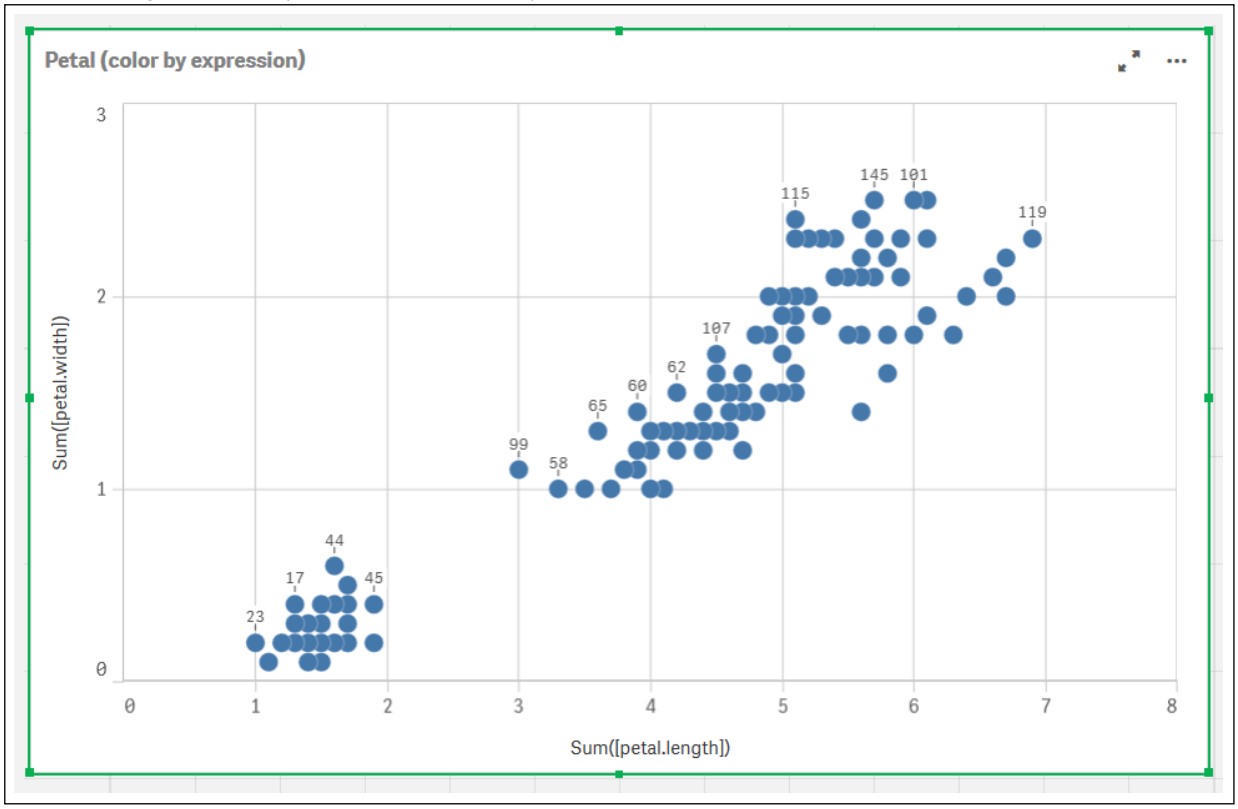
1. Trascinare un **Grafico a dispersione** in un nuovo foglio. Denominare il grafico *Petal (colore per espressione)*.
2. Creare una variabile per specificare il numero di cluster. Per la variabile **Nome**, inserire *KmeansPetalClusters*. Per la variabile **Definizione**, inserire *=2*.
3. Creare una variabile per specificare il numero di ripetizioni. Per la variabile **Nome**, inserire *KmeansNumberIterations*. Per la variabile **Definizione**, inserire *=1*.
4. Configurare **Dati** per il grafico:
    i. Sotto **Dimensioni**, scegliere *id* per il campo per **Bolla**. Inserire Id cluster per l'etichetta.
    ii. Sotto **Misure**, scegliere *Sum([petal.length])* per l'espressione per **asse X**.
    iii. Sotto **Misure**, scegliere *Sum([petal.width])* per l'espressione per **asse Y**.

Impostazioni dati per il grafico Petal (colore per espressione)



I punti dati vengono riportati sul grafico.

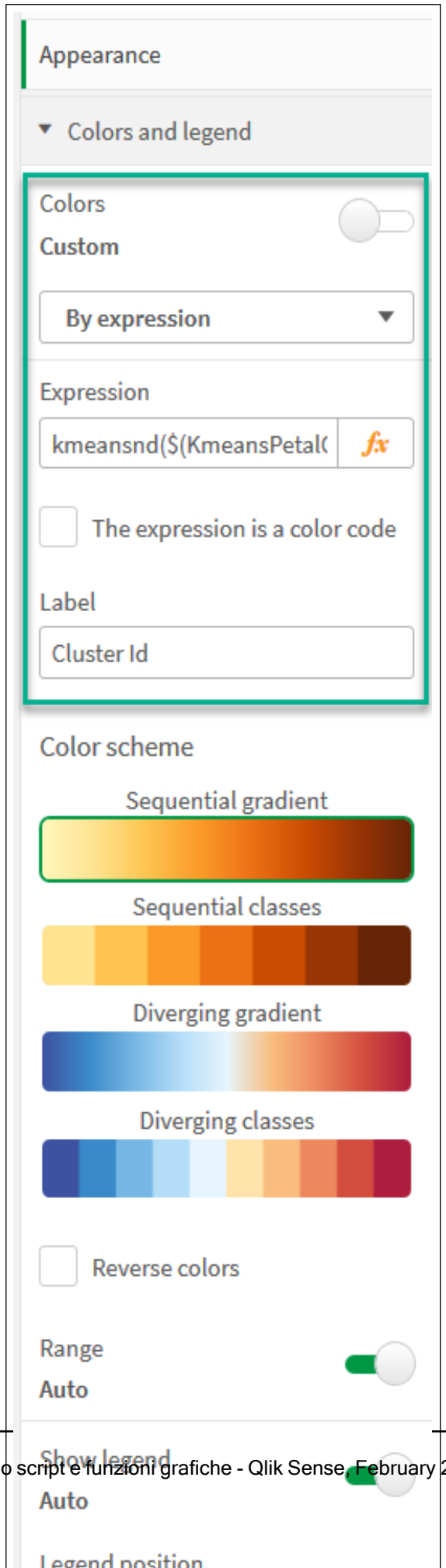
Punti dati sul grafico Petal (colore per espressione)



5. Configurare **Aspetto** per il grafico:
   i. Sotto **Colori e legenda**, scegliere **Personalizzato** per **Colori**.
   ii. Scegliere di colorare il grafico **Per espressione**.
   iii. Inserire quanto segue per **Expression**: *kmeansnd ($(KmeansPetalClusters),$(KmeansNumberIterations), Sum([petal.length]), Sum ([petal.width]),Sum([sepal.length]), Sum([sepal.width]))*
   Notare che *KmeansPetalClusters* è la variabile che impostiamo a *2*.
   KmeansNumberIterations è la variabile che impostiamo a *1*.
   In alternativa, inserire quanto segue: *kmeansnd(2, 2, Sum([petal.length]), Sum ([petal.width]),Sum([sepal.length]), Sum([sepal.width]))*
   iv. Deselezionare la casella di controllo per **L'espressione è un codice cromatico**.

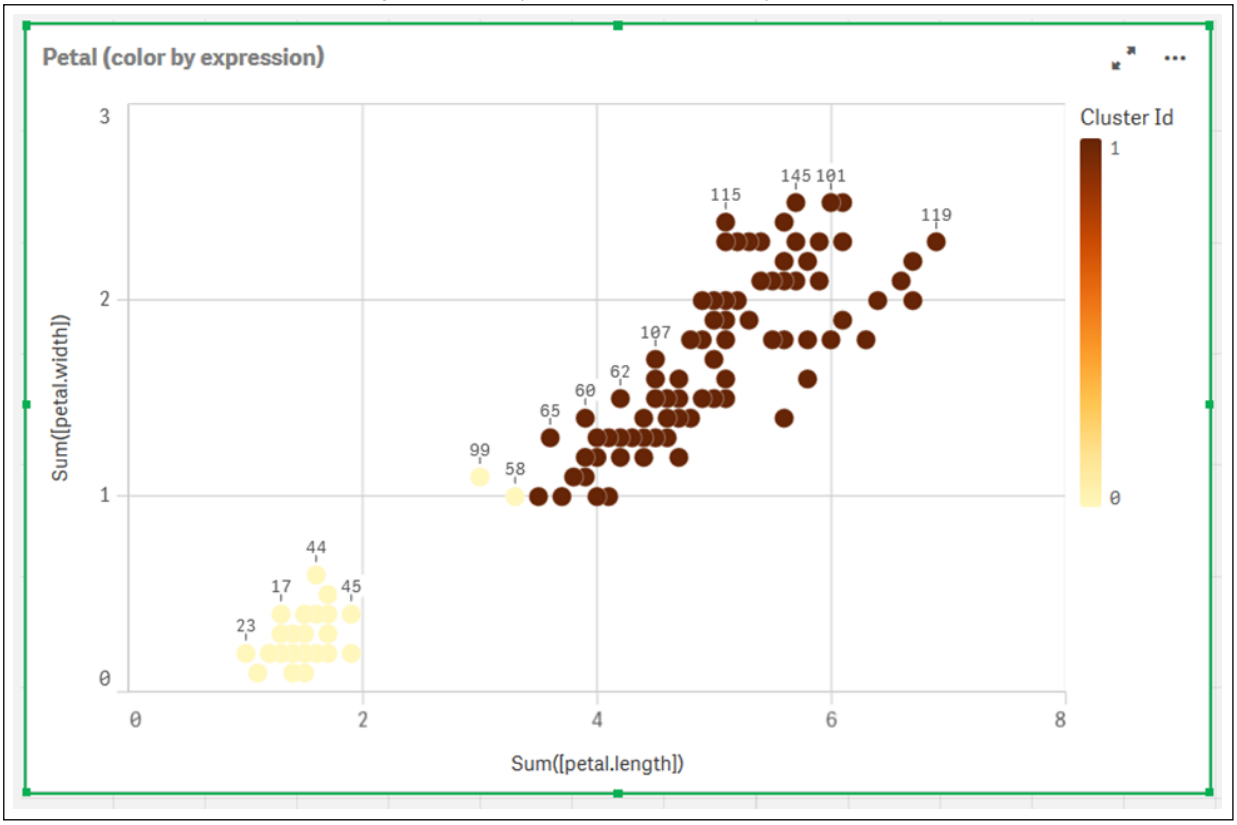
v. Inserire quanto segue per **Etichetta**: *Id cluster*

Impostazioni aspetto per il grafico Petal (colore per espressione)


Appearance
Colors and legend
Colors
Custom
By expression
Expression
kmeansnd($(KmeansPetal(
The expression is a color code
Label
Cluster Id
Color scheme
Sequential gradient
Sequential classes
Diverging gradient
Diverging classes
Reverse colors
Range
Auto
Show legend
Auto
Legend position

I due cluster sul grafico sono colorati in base all'espressione KMeans.

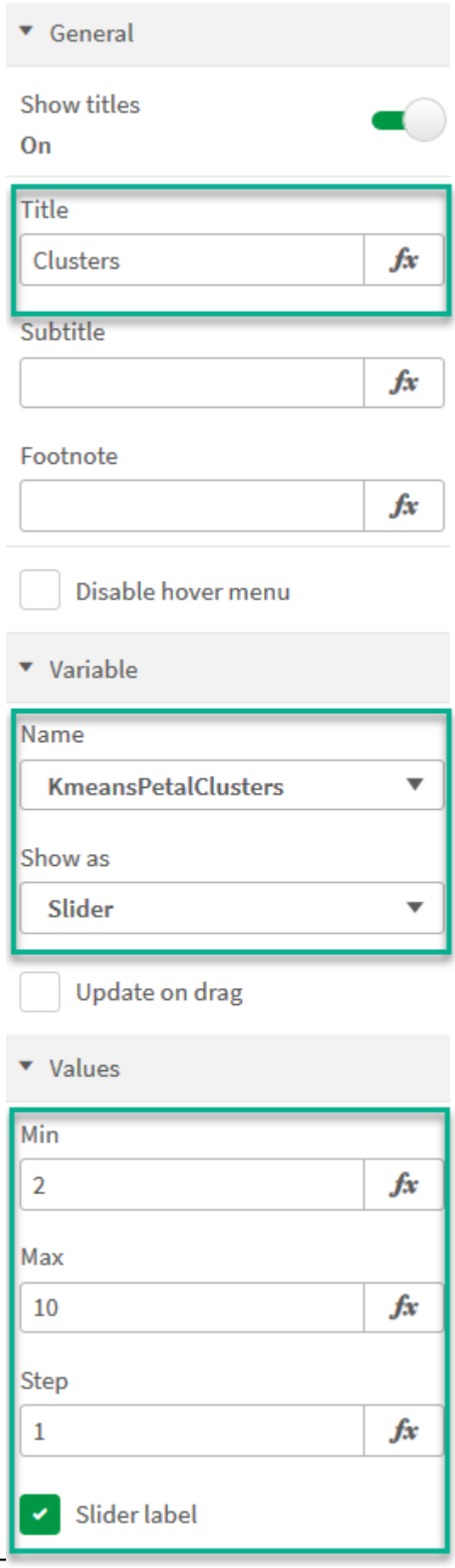
Cluster colorati per espressione sul grafico Petal (colore per espressione)



6. Aggiungere una casella **Input variabile** per il numero di cluster.
    i. Sotto **Oggetti personalizzati** nel pannello **Asset**, scegliere **Qlik Dashboard bundle**. Se non si ha accesso al dashboard bundle, è comunque possibile modificare il numero di cluster utilizzando la variabile che abbiamo creato, oppure direttamente come intero nell'espressione.
    ii. Trascinare una casella **Input variabile** sul foglio.
    iii. Sotto **Aspetto**, fare clic su **Generale**.
    iv. Inserire quanto segue per **Titolo**: *Cluster*
    v. Fare clic su **Variabile**.
    vi. Scegliere la seguente variabile per **Nome**: *KmeansPetalClusters*.
    vii. Scegliere **Cursore** per **Mostra come**.

viii. Scegliere **Valori**, quindi configurare le impostazioni come richiesto.

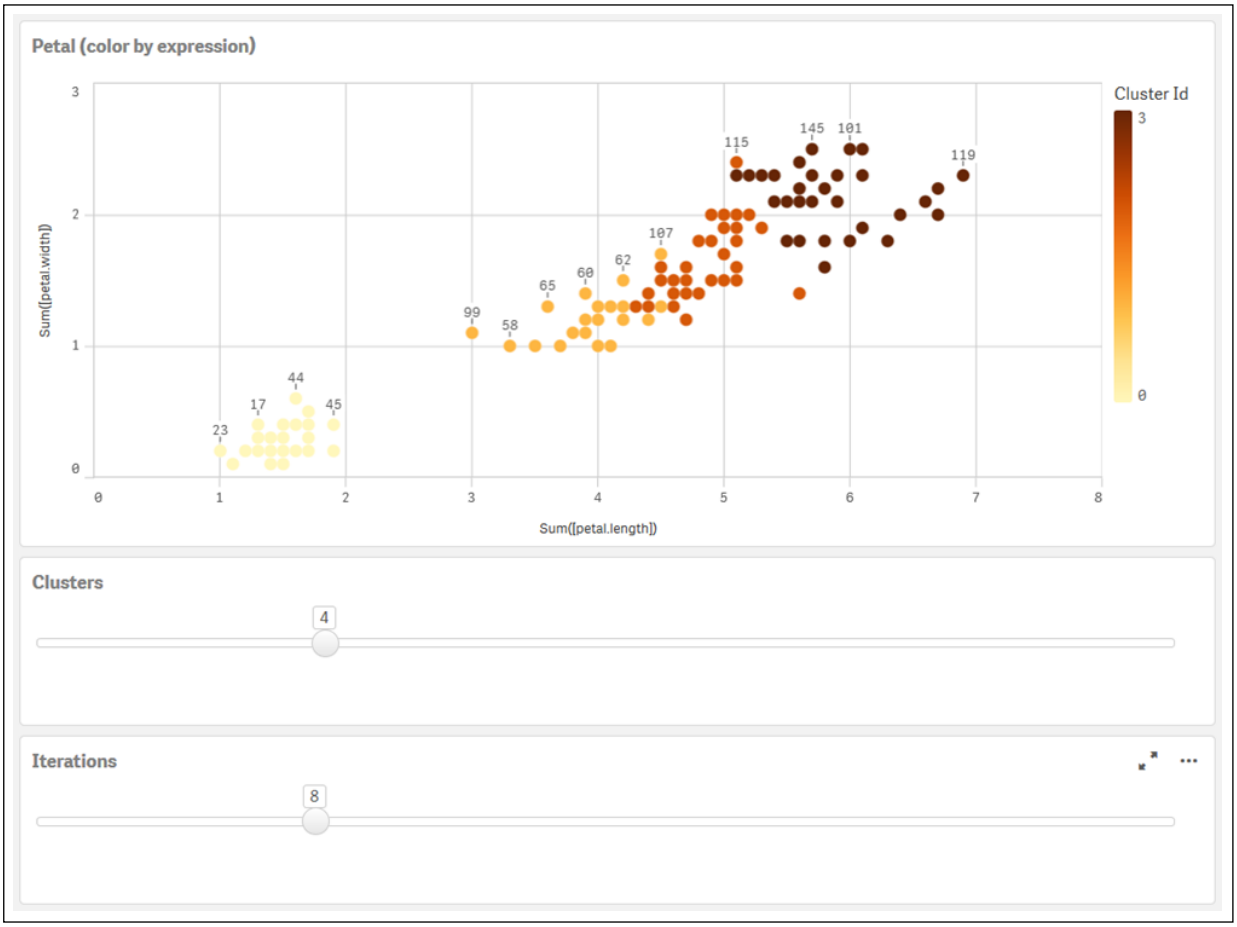
Aspetto per la casella di input variabile Cluster


General
Show titles
On
Title
Clusters
Subtitle
Footnote
Disable hover menu
Variable
Name
KmeansPetalClusters
Show as
Slider
Update on drag
Values
Min
2
Max
10
Step
1
Slider label

7. Aggiungere una casella **Input variabile** per il numero di ripetizioni.
    i. Trascinare una casella **Input variabile** sul foglio.
    ii. Sotto **Aspetto**, scegliere **Generale**.
    iii. Inserire quanto segue per **Titolo**: *Ripetizioni*
    iv. Sotto **Aspetto**, scegliere **Variabile**.
    v. Scegliere la seguente variabile sotto **Nome**: *KmeansNumberIterations*.
    vi. Configurare le impostazioni aggiuntive come richiesto,

    Ora è possibile modificare il numero di cluster e di ripetizioni usando i cursori nelle caselle di input variabile.

    Cluster colorati per espressione sul grafico Petal (colore per espressione)

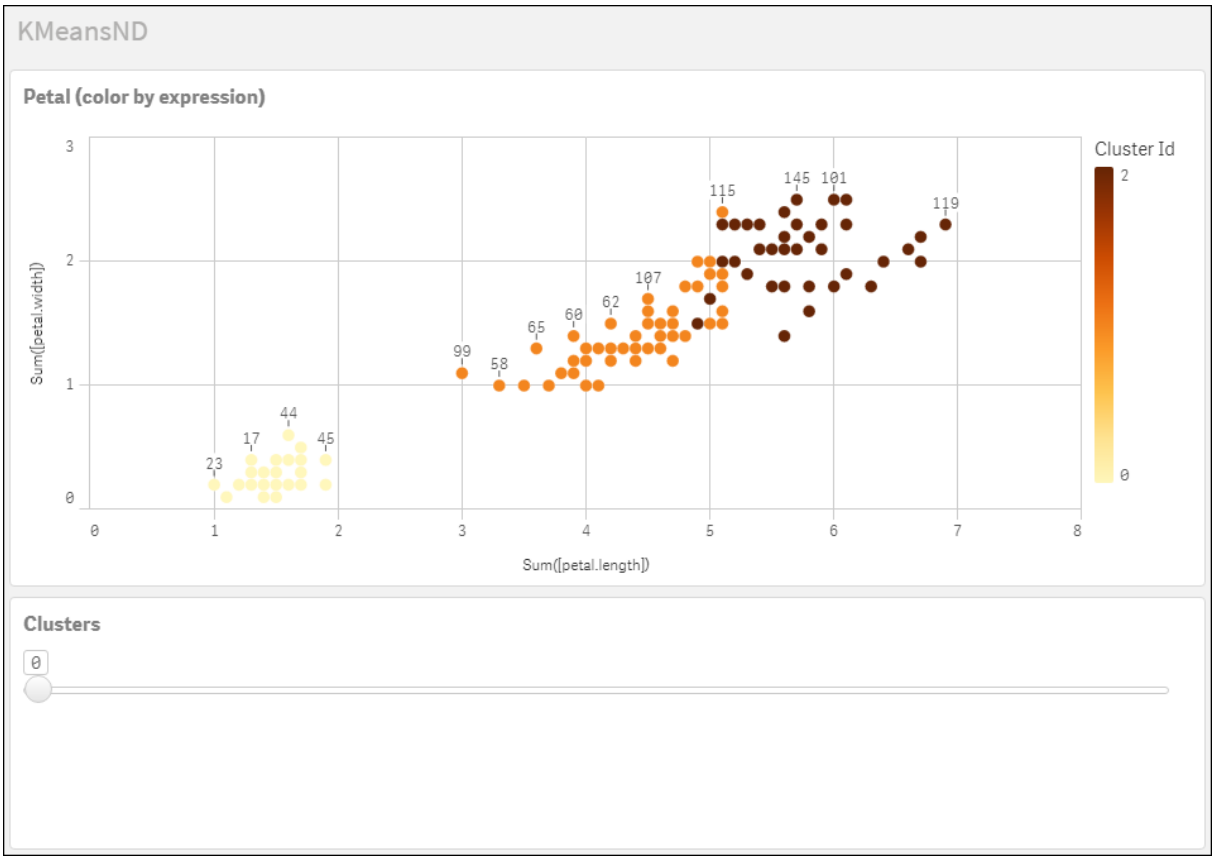


Clustering automatico

Le funzioni **KMeans** supportano il clustering automatico mediante un metodo chiamato differenza di profondità (DeD, Depth Difference). Quando un utente imposta lo 0 per il numero di cluster, viene determinato un numero ottimale di cluster per tale set di dati. Notare che mentre un valore intero per il numero di cluster (*k*) non viene restituito esplicitamente, viene calcolato all'interno dell'algoritmo KMeans. Ad esempio, se viene specificato 0 nella funzione del valore di *KmeansPetalClusters* o se viene impostato

mediante una casella di input variabile, le assegnazioni cluster vengono calcolate automaticamente per il set di dati in base a un numero ottimale di cluster. In base al set di dati Iris, se viene selezionato 0 per il numero di cluster, l'algoritmo determinerà (clustering automatico) un numero ottimale di cluster (3) per questo set di dati.

Il metodo di differenza di profondità KMeans determina il numero ottimale di cluster quando (k) viene impostato a 0.



Serie di dati Iris: Caricamento inline per l'editor caricamento dati in Qlik Sense

```
IrisData:
Load * Inline [
sepal.length, sepal.width, petal.length, petal.width, variety, id
5.1, 3.5, 1.4, 0.2, Setosa, 1
4.9, 3, 1.4, 0.2, Setosa, 2
4.7, 3.2, 1.3, 0.2, Setosa, 3
4.6, 3.1, 1.5, 0.2, Setosa, 4
5, 3.6, 1.4, 0.2, Setosa, 5
5.4, 3.9, 1.7, 0.4, Setosa, 6
4.6, 3.4, 1.4, 0.3, Setosa, 7
5, 3.4, 1.5, 0.2, Setosa, 8
4.4, 2.9, 1.4, 0.2, Setosa, 9
4.9, 3.1, 1.5, 0.1, Setosa, 10
5.4, 3.7, 1.5, 0.2, Setosa, 11
4.8, 3.4, 1.6, 0.2, Setosa, 12
4.8, 3, 1.4, 0.1, Setosa, 13
4.3, 3, 1.1, 0.1, Setosa, 14
5.8, 4, 1.2, 0.2, Setosa, 15
```

```
5.7, 4.4, 1.5, 0.4, Setosa, 16
5.4, 3.9, 1.3, 0.4, Setosa, 17
5.1, 3.5, 1.4, 0.3, Setosa, 18
5.7, 3.8, 1.7, 0.3, Setosa, 19
5.1, 3.8, 1.5, 0.3, Setosa, 20
5.4, 3.4, 1.7, 0.2, Setosa, 21
5.1, 3.7, 1.5, 0.4, Setosa, 22
4.6, 3.6, 1, 0.2, Setosa, 23
5.1, 3.3, 1.7, 0.5, Setosa, 24
4.8, 3.4, 1.9, 0.2, Setosa, 25
5, 3, 1.6, 0.2, Setosa, 26
5, 3.4, 1.6, 0.4, Setosa, 27
5.2, 3.5, 1.5, 0.2, Setosa, 28
5.2, 3.4, 1.4, 0.2, Setosa, 29
4.7, 3.2, 1.6, 0.2, Setosa, 30
4.8, 3.1, 1.6, 0.2, Setosa, 31
5.4, 3.4, 1.5, 0.4, Setosa, 32
5.2, 4.1, 1.5, 0.1, Setosa, 33
5.5, 4.2, 1.4, 0.2, Setosa, 34
4.9, 3.1, 1.5, 0.1, Setosa, 35
5, 3.2, 1.2, 0.2, Setosa, 36
5.5, 3.5, 1.3, 0.2, Setosa, 37
4.9, 3.1, 1.5, 0.1, Setosa, 38
4.4, 3, 1.3, 0.2, Setosa, 39
5.1, 3.4, 1.5, 0.2, Setosa, 40
5, 3.5, 1.3, 0.3, Setosa, 41
4.5, 2.3, 1.3, 0.3, Setosa, 42
4.4, 3.2, 1.3, 0.2, Setosa, 43
5, 3.5, 1.6, 0.6, Setosa, 44
5.1, 3.8, 1.9, 0.4, Setosa, 45
4.8, 3, 1.4, 0.3, Setosa, 46
5.1, 3.8, 1.6, 0.2, Setosa, 47
4.6, 3.2, 1.4, 0.2, Setosa, 48
5.3, 3.7, 1.5, 0.2, Setosa, 49
5, 3.3, 1.4, 0.2, Setosa, 50
7, 3.2, 4.7, 1.4, Versicolor, 51
6.4, 3.2, 4.5, 1.5, Versicolor, 52
6.9, 3.1, 4.9, 1.5, Versicolor, 53
5.5, 2.3, 4, 1.3, Versicolor, 54
6.5, 2.8, 4.6, 1.5, Versicolor, 55
5.7, 2.8, 4.5, 1.3, Versicolor, 56
6.3, 3.3, 4.7, 1.6, Versicolor, 57
4.9, 2.4, 3.3, 1, Versicolor, 58
6.6, 2.9, 4.6, 1.3, Versicolor, 59
5.2, 2.7, 3.9, 1.4, Versicolor, 60
5, 2, 3.5, 1, Versicolor, 61
5.9, 3, 4.2, 1.5, Versicolor, 62
6, 2.2, 4, 1, Versicolor, 63
6.1, 2.9, 4.7, 1.4, Versicolor, 64
5.6, 2.9, 3.6, 1.3, Versicolor, 65
6.7, 3.1, 4.4, 1.4, Versicolor, 66
5.6, 3, 4.5, 1.5, Versicolor, 67
5.8, 2.7, 4.1, 1, Versicolor, 68
6.2, 2.2, 4.5, 1.5, Versicolor, 69
5.6, 2.5, 3.9, 1.1, Versicolor, 70
```

```
5.9, 3.2, 4.8, 1.8, Versicolor, 71
6.1, 2.8, 4, 1.3, Versicolor, 72
6.3, 2.5, 4.9, 1.5, Versicolor, 73
6.1, 2.8, 4.7, 1.2, Versicolor, 74
6.4, 2.9, 4.3, 1.3, Versicolor, 75
6.6, 3, 4.4, 1.4, Versicolor, 76
6.8, 2.8, 4.8, 1.4, Versicolor, 77
6.7, 3, 5, 1.7, Versicolor, 78
6, 2.9, 4.5, 1.5, Versicolor, 79
5.7, 2.6, 3.5, 1, Versicolor, 80
5.5, 2.4, 3.8, 1.1, Versicolor, 81
5.5, 2.4, 3.7, 1, Versicolor, 82
5.8, 2.7, 3.9, 1.2, Versicolor, 83
6, 2.7, 5.1, 1.6, Versicolor, 84
5.4, 3, 4.5, 1.5, Versicolor, 85
6, 3.4, 4.5, 1.6, Versicolor, 86
6.7, 3.1, 4.7, 1.5, Versicolor, 87
6.3, 2.3, 4.4, 1.3, Versicolor, 88
5.6, 3, 4.1, 1.3, Versicolor, 89
5.5, 2.5, 4, 1.3, Versicolor, 90
5.5, 2.6, 4.4, 1.2, Versicolor, 91
6.1, 3, 4.6, 1.4, Versicolor, 92
5.8, 2.6, 4, 1.2, Versicolor, 93
5, 2.3, 3.3, 1, Versicolor, 94
5.6, 2.7, 4.2, 1.3, Versicolor, 95
5.7, 3, 4.2, 1.2, Versicolor, 96
5.7, 2.9, 4.2, 1.3, Versicolor, 97
6.2, 2.9, 4.3, 1.3, Versicolor, 98
5.1, 2.5, 3, 1.1, Versicolor, 99
5.7, 2.8, 4.1, 1.3, Versicolor, 100
6.3, 3.3, 6, 2.5, Virginica, 101
5.8, 2.7, 5.1, 1.9, Virginica, 102
7.1, 3, 5.9, 2.1, Virginica, 103
6.3, 2.9, 5.6, 1.8, Virginica, 104
6.5, 3, 5.8, 2.2, Virginica, 105
7.6, 3, 6.6, 2.1, Virginica, 106
4.9, 2.5, 4.5, 1.7, Virginica, 107
7.3, 2.9, 6.3, 1.8, Virginica, 108
6.7, 2.5, 5.8, 1.8, Virginica, 109
7.2, 3.6, 6.1, 2.5, Virginica, 110
6.5, 3.2, 5.1, 2, Virginica, 111
6.4, 2.7, 5.3, 1.9, Virginica, 112
6.8, 3, 5.5, 2.1, Virginica, 113
5.7, 2.5, 5, 2, Virginica, 114
5.8, 2.8, 5.1, 2.4, Virginica, 115
6.4, 3.2, 5.3, 2.3, Virginica, 116
6.5, 3, 5.5, 1.8, Virginica, 117
7.7, 3.8, 6.7, 2.2, Virginica, 118
7.7, 2.6, 6.9, 2.3, Virginica, 119
6, 2.2, 5, 1.5, Virginica, 120
6.9, 3.2, 5.7, 2.3, Virginica, 121
5.6, 2.8, 4.9, 2, Virginica, 122
7.7, 2.8, 6.7, 2, Virginica, 123
6.3, 2.7, 4.9, 1.8, Virginica, 124
6.7, 3.3, 5.7, 2.1, Virginica, 125
```

```
7.2, 3.2, 6, 1.8, Virginica, 126
6.2, 2.8, 4.8, 1.8, Virginica, 127
6.1, 3, 4.9, 1.8, Virginica, 128
6.4, 2.8, 5.6, 2.1, Virginica, 129
7.2, 3, 5.8, 1.6, Virginica, 130
7.4, 2.8, 6.1, 1.9, Virginica, 131
7.9, 3.8, 6.4, 2, Virginica, 132
6.4, 2.8, 5.6, 2.2, Virginica, 133
6.3, 2.8, 5.1, 1.5, Virginica, 134
6.1, 2.6, 5.6, 1.4, Virginica, 135
7.7, 3, 6.1, 2.3, Virginica, 136
6.3, 3.4, 5.6, 2.4, Virginica, 137
6.4, 3.1, 5.5, 1.8, Virginica, 138
6, 3, 4.8, 1.8, Virginica, 139
6.9, 3.1, 5.4, 2.1, Virginica, 140
6.7, 3.1, 5.6, 2.4, Virginica, 141
6.9, 3.1, 5.1, 2.3, Virginica, 142
5.8, 2.7, 5.1, 1.9, Virginica, 143
6.8, 3.2, 5.9, 2.3, Virginica, 144
6.7, 3.3, 5.7, 2.5, Virginica, 145
6.7, 3, 5.2, 2.3, Virginica, 146
6.3, 2.5, 5, 1.9, Virginica, 147
6.5, 3, 5.2, 2, Virginica, 148
6.2, 3.4, 5.4, 2.3, Virginica, 149
5.9, 3, 5.1, 1.8, Virginica, 150
];
```

KMeansCentroid2D - funzione per grafici

KMeansCentroid2D() valuta le righe del grafico applicando il clustering K-means, e per ciascuna riga del grafico visualizza la coordinata desiderata del cluster a cui è stato assegnato questo punto dati. Le colonne utilizzate dall'algoritmo di clustering sono determinate rispettivamente dai parametri coordinate_1 e coordinate_2. Sono entrambe aggregazioni. Il numero di cluster creati è determinato dal parametro num_clusters. I dati possono essere normalizzati in via opzionale dal parametro norm.

KMeansCentroid2D restituisce un valore per punto dati. Il valore restituito è duale ed è una delle coordinate della posizione corrispondente al centro del cluster a cui ciascun punto dati è stato assegnato.

Sintassi:

```
KMeansCentroid2D(num_clusters, coordinate_no, coordinate_1, coordinate_2 [, norm])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
num_clusters	Intero che specifica il numero di cluster.

Argomento	Descrizione
coordinate_no	Il numero di coordinata desiderato dei centroid (corrispondente, ad esempio, all'asse x, y o z).
coordinate_1	L'aggregazione che calcola la prima coordinata, in genere l'asse x del grafico a dispersione che può essere effettuato dal grafico. Il parametro aggiuntivo, coordinate_2, calcola la seconda coordinata.
norm	Il metodo di normalizzazione opzionale applicato alle serie di dati prima del clustering K-means. Possibili valori: 0 o 'nessuno' per l'assenza di normalizzazione 1 o 'zscore' per la normalizzazione z-score 2 o 'minmax' per la normalizzazione min-max Se non viene fornito alcun parametro o se il parametro fornito risulta errato, non viene applicata alcuna normalizzazione. Z-score normalizza i dati in base alla deviazione standard e media della funzionalità. Z-score non assicura che ciascuna funzionalità abbia la stessa scala, ma rappresenta un approccio migliore a min-max quando si ha a che fare con outlier. La normalizzazione min-max assicura che le funzionalità abbiano la stessa scala prelevando i valori minimo e massimo di ciascuna di esse e ricalcolando ciascun datapoint.

Clustering automatico

Le funzioni **KMeans** supportano il clustering automatico mediante un metodo chiamato differenza di profondità (DeD, Depth Difference). Quando un utente imposta lo 0 per il numero di cluster, viene determinato un numero ottimale di cluster per tale set di dati. Notare che mentre un valore intero per il numero di cluster (*k*) non viene restituito esplicitamente, viene calcolato all'interno dell'algoritmo KMeans. Ad esempio, se viene specificato 0 nella funzione del valore di *KmeansPetalClusters* o se viene impostato mediante una casella di input variabile, le assegnazioni cluster vengono calcolate automaticamente per il set di dati in base a un numero ottimale di cluster.

KMeansCentroidND - funzione per grafici

KMeansCentroidND() valuta le righe del grafico applicando il clustering K-means, e per ciascuna riga del grafico visualizza la coordinata desiderata del cluster a cui è stato assegnato questo punto dati. Le colonne utilizzate dall'algoritmo di clustering sono determinate dai parametri coordinate_1, coordinate_2, ecc. fino a n colonne. Sono tutte aggregazioni. Il numero di cluster creati è determinato dal parametro num_ clusters.

KMeansCentroidND restituisce un valore per riga. Il valore restituito è duale ed è una delle coordinate della posizione corrispondente al centro del cluster a cui ciascun punto dati è stato assegnato.

Sintassi:

```
KMeansCentroidND((num_clusters, num_iter, coordinate_no, coordinate_1,
coordinate_2 [,coordinate_3 [, ...]])
```

Tipo di dati restituiti: duale

Argomenti:

Argomenti

Argomento	Descrizione
num_clusters	Intero che specifica il numero di cluster.
num_iter	Il numero di ripetizioni di clustering con centri cluster reinizializzati.
coordinate_no	Il numero di coordinata desiderato dei centroid (corrispondente, ad esempio, all'asse x, y o z).
coordinate_1	L'aggregazione che calcola la prima coordinata, in genere l'asse x (di un grafico a dispersione che può essere effettuato dal grafico). I parametri aggiuntivi calcolano le seconde, terze e quarte coordinate, ecc.

Clustering automatico

Le funzioni **KMeans** supportano il clustering automatico mediante un metodo chiamato differenza di profondità (DeD, Depth Difference). Quando un utente imposta lo 0 per il numero di cluster, viene determinato un numero ottimale di cluster per tale set di dati. Notare che mentre un valore intero per il numero di cluster (*k*) non viene restituito esplicitamente, viene calcolato all'interno dell'algoritmo KMeans. Ad esempio, se viene specificato 0 nella funzione del valore di *KmeansPetalClusters* o se viene impostato mediante una casella di input variabile, le assegnazioni cluster vengono calcolate automaticamente per il set di dati in base a un numero ottimale di cluster.

STL_Trend - funzione per grafici

STL_Trend è una funzione di scomposizione delle serie temporali. Insieme a **STL_Seasonal** e a **STL_Residual**, questa funzione viene utilizzata per scomporre una serie temporale in componenti stagionali, di tendenza e residuali. Nel contesto dell'algoritmo STL, la scomposizione delle serie temporali viene utilizzata per identificare sia un modello stagionale ricorrente sia una tendenza generale, data una metrica di ingresso e altri parametri. La funzione **STL_Trend** identifica una tendenza generale, indipendente da schemi o cicli stagionali, dai dati delle serie temporali.

Le tre funzioni STL sono correlate alla metrica di input attraverso una semplice somma:

STL_Trend + **STL_Seasonal** + **STL_Residual** = Metrica di input

STL (seasonal and trend decomposition using Loess) impiega tecniche di smoothing dei dati e, attraverso i suoi parametri di input, consente all'utente di regolare la periodicità dei calcoli che esegue. Questa periodicità determina il modo in cui la dimensione temporale della metrica di input (una misura) viene segmentata nell'analisi.

Come minimo, **STL_Trend** prende una metrica di input (`Expression`) e un valore intero per il suo `Period`, e restituisce un valore in virgola mobile. La metrica di input avrà la forma di un'aggregazione che varia lungo la dimensione temporale. Facoltativamente, è possibile includere i valori per `seasonal_smoother` e `trend_smoother` al fine di regolare l'efficacia dell'algoritmo di smoothing.

Sintassi:

```
STL_Trend(Expression, period [,seasonal_smoother [,trend_smoother]])
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
Expression	L'aggregazione o la misura, variabile lungo la dimensione temporale, che contiene il calcolo che si vuole analizzare. Non deve essere un valore costante.
period	La periodicità del set di dati. Questo parametro è un valore intero che rappresenta il numero di passi discreti che costituiscono un periodo, o ciclo stagionale, del segnale. Ad esempio, se la serie temporale è segmentata in una sezione per ogni trimestre dell'anno, è necessario impostare **period** a un valore di 4 per definire la periodicità come Anno.
seasonal_ smoother	Durata del seasonal smoother. Deve essere un numero intero dispari. Il seasonal smoother utilizza i dati relativi a una particolare fase della variazione stagionale, per un certo numero di periodi. Per ogni periodo viene utilizzato un passo discreto della dimensione temporale. Il seasonal smoother indica il numero di periodi utilizzati per lo smoothing. Ad esempio, se la dimensione temporale è segmentata per mese e il periodo è l'anno (12), la componente stagionale sarà elaborata in modo che ogni particolare mese di ogni anno sia calcolato a partire dai dati dello stesso mese, sia in quell'anno che negli anni adiacenti. Il valore **seasonal_smoother** è il numero di anni utilizzato per lo smoothing.
trend_ smoother	Durata del trend smoother. Deve essere un numero intero dispari. Il trend smoother utilizza la stessa scala temporale del parametro **period** e il suo valore è il numero di granuli utilizzati per lo smoothing. Ad esempio, se una serie temporale è segmentata per mese, il trend smoother corrisponderà al numero di mesi utilizzato per lo smoothing.

La funzione grafico **STL_Trend** viene spesso utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
STL_Seasonal - funzione per grafici (page 1410)	È la funzione utilizzata per calcolare la componente stagionale di una serie temporale.
STL_Residual - funzione per grafici (page 1412)	Quando si scompone una metrica di input in componente stagionale e componente di tendenza, parte della variazione della misura non rientra in nessuna delle due componenti principali. La funzione **STL_Residual** calcola questa parte della scomposizione.

Per un tutorial con un esempio completo che mostra come utilizzare questa funzione, vedere *Tutorial - Scomposizione delle serie temporali in Qlik Sense (page 1414)*.

STL_Seasonal - funzione per grafici

STL_Seasonal è una funzione di scomposizione delle serie temporali. Insieme a **STL_Trend** e a **STL_Residual**, questa funzione viene utilizzata per scomporre una serie temporale in componenti stagionali, di tendenza e residuali. Nel contesto dell'algoritmo STL, la scomposizione delle serie temporali viene utilizzata per identificare sia un modello stagionale ricorrente sia una tendenza generale, data una metrica di ingresso e altri parametri. La funzione **STL_Seasonal** è in grado di identificare un modello stagionale all'interno di una serie temporale, separandolo dalla tendenza generale mostrata dai dati.

Le tre funzioni STL sono correlate alla metrica di input attraverso una semplice somma:

STL_Trend + **STL_Seasonal** + **STL_Residual** = Metrica di input

STL (seasonal and trend decomposition using Loess) impiega tecniche di smoothing dei dati e, attraverso i suoi parametri di input, consente all'utente di regolare la periodicità dei calcoli che esegue. Questa periodicità determina il modo in cui la dimensione temporale della metrica di input (una misura) viene segmentata nell'analisi.

Come minimo, **Seasonal** prende una metrica di input (`Expression`) e un valore intero per il suo `Period`, e restituisce un valore in virgola mobile. La metrica di input avrà la forma di un'aggregazione che varia lungo la dimensione temporale. Facoltativamente, è possibile includere i valori per `seasonal_smoother` e `trend_smoother` al fine di regolare l'efficacia dell'algoritmo di smoothing.

Sintassi:

```
STL_Seasonal(Expression, period [,seasonal_smoother [,trend_smoother]])
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
Expression	L'aggregazione o la misura, variabile lungo la dimensione temporale, che contiene il calcolo che si vuole analizzare. Non deve essere un valore costante.
period	La periodicità del set di dati. Questo parametro è un valore intero che rappresenta il numero di passi discreti che costituiscono un periodo, o ciclo stagionale, del segnale. Ad esempio, se la serie temporale è segmentata in una sezione per ogni trimestre dell'anno, è necessario impostare **period** a un valore di 4 per definire la periodicità come Anno.
seasonal_ smoother	Durata del seasonal smoother. Deve essere un numero intero dispari. Il seasonal smoother utilizza i dati relativi a una particolare fase della variazione stagionale, per un certo numero di periodi. Per ogni periodo viene utilizzato un passo discreto della dimensione temporale. Il seasonal smoother indica il numero di periodi utilizzati per lo smoothing. Ad esempio, se la dimensione temporale è segmentata per mese e il periodo è l'anno (12), la componente stagionale sarà elaborata in modo che ogni particolare mese di ogni anno sia calcolato a partire dai dati dello stesso mese, sia in quell'anno che negli anni adiacenti. Il valore **seasonal_smoother** è il numero di anni utilizzato per lo smoothing.
trend_ smoother	Durata del trend smoother. Deve essere un numero intero dispari. Il trend smoother utilizza la stessa scala temporale del parametro **period** e il suo valore è il numero di granuli utilizzati per lo smoothing. Ad esempio, se una serie temporale è segmentata per mese, il trend smoother corrisponderà al numero di mesi utilizzato per lo smoothing.

La funzione grafico **STL_Seasonal** viene spesso utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
STL_Trend - funzione per grafici (page 1408)	È la funzione utilizzata per calcolare la componente di tendenza di una serie temporale.
STL_Residual - funzione per grafici (page 1412)	Quando si scompone una metrica di input in componente stagionale e componente di tendenza, parte della variazione della misura non rientra in nessuna delle due componenti principali. La funzione **STL_Residual** calcola questa parte della scomposizione.

Per un tutorial con un esempio completo che mostra come utilizzare questa funzione, vedere *Tutorial - Scomposizione delle serie temporali in Qlik Sense (page 1414)*.

STL_Residual - funzione per grafici

STL_Residual è una funzione di scomposizione delle serie temporali. Insieme a **STL_Seasonal** e a **STL_Trend**, questa funzione viene utilizzata per scomporre una serie temporale in componenti stagionali, di tendenza e residuali. Nel contesto dell'algoritmo STL, la scomposizione delle serie temporali viene utilizzata per identificare sia un modello stagionale ricorrente sia una tendenza generale, data una metrica di ingresso e altri parametri. Nell'eseguire questa operazione, una parte della variazione della metrica di input non rientrerà né nella componente stagionale né in quella di tendenza e sarà definita come componente residua. La funzione grafico **STL_Residual** acquisisce questa parte del calcolo.

Le tre funzioni STL sono correlate alla metrica di input attraverso una semplice somma:

STL_Trend + **STL_Seasonal** + **STL_Residual** = Metrica di input

STL (seasonal and trend decomposition using Loess) impiega tecniche di smoothing dei dati e, attraverso i suoi parametri di input, consente all'utente di regolare la periodicità dei calcoli che esegue. Questa periodicità determina il modo in cui la dimensione temporale della metrica di input (una misura) viene segmentata nell'analisi.

Poiché la scomposizione delle serie temporali cerca principalmente la stagionalità e le variazioni generali dei dati, l'informazione nel residuo è considerata la meno significativa delle tre componenti. Tuttavia, una componente residua obliqua o periodica può aiutare a identificare problemi nel calcolo, come ad esempio un'impostazione errata della periodicità.

Come minimo, **STL_Residual** prende una metrica di input (`Expression`) e un valore intero per il suo `Period`, e restituisce un valore in virgola mobile. La metrica di input avrà la forma di un'aggregazione che varia lungo la dimensione temporale. Facoltativamente, è possibile includere i valori per `seasonal_smoother` e `trend_smoother` al fine di regolare l'efficacia dell'algoritmo di smoothing.

Sintassi:

```
STL_Residual(Expression, period [,seasonal_smoother [,trend_smoother]])
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
Expression	L'aggregazione o la misura, variabile lungo la dimensione temporale, che contiene il calcolo che si vuole analizzare. Non deve essere un valore costante.
period	La periodicità del set di dati. Questo parametro è un valore intero che rappresenta il numero di passi discreti che costituiscono un periodo, o ciclo stagionale, del segnale. Ad esempio, se la serie temporale è segmentata in una sezione per ogni trimestre dell'anno, è necessario impostare **period** a un valore di 4 per definire la periodicità come Anno.
seasonal_ smoother	Durata del seasonal smoother. Deve essere un numero intero dispari. Il seasonal smoother utilizza i dati relativi a una particolare fase della variazione stagionale, per un certo numero di periodi. Per ogni periodo viene utilizzato un passo discreto della dimensione temporale. Il seasonal smoother indica il numero di periodi utilizzati per lo smoothing. Ad esempio, se la dimensione temporale è segmentata per mese e il periodo è l'anno (12), la componente stagionale sarà elaborata in modo che ogni particolare mese di ogni anno sia calcolato a partire dai dati dello stesso mese, sia in quell'anno che negli anni adiacenti. Il valore **seasonal_smoother** è il numero di anni utilizzato per lo smoothing.
trend_ smoother	Durata del trend smoother. Deve essere un numero intero dispari. Il trend smoother utilizza la stessa scala temporale del parametro **period** e il suo valore è il numero di granuli utilizzati per lo smoothing. Ad esempio, se una serie temporale è segmentata per mese, il trend smoother corrisponderà al numero di mesi utilizzato per lo smoothing.

La funzione grafico **STL_Residual** viene spesso utilizzata in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
STL_Seasonal - funzione per grafici (page 1410)	È la funzione utilizzata per calcolare la componente stagionale di una serie temporale.
STL_Trend - funzione per grafici (page 1408)	È la funzione utilizzata per calcolare la componente di tendenza di una serie temporale.

Per un tutorial con un esempio completo che mostra come utilizzare questa funzione, vedere *Tutorial - Scomposizione delle serie temporali in Qlik Sense (page 1414)*.

Tutorial - Scomposizione delle serie temporali in Qlik Sense

Questo tutorial dimostra l'utilizzo di tre funzioni grafiche per scomporre una serie temporale utilizzando l'algoritmo STL.

Questo tutorial utilizza i dati delle serie temporali del numero di passeggeri di una compagnia aerea al mese per dimostrare la funzionalità dell'algoritmo STL. Le funzioni del grafico **STL_Trend**, **STL_Seasonal** e **STL_Residual** verranno utilizzate per creare le visualizzazioni. Per ulteriori informazioni sulla scomposizione delle serie temporali in Qlik Sense, vedere *Funzioni di scomposizione serie temporale (page 1360)*.

Creare un'app

Si inizia creando una nuova app e importandovi il set di dati.

Scaricare questo set di dati:

Tutorial - Scomposizione delle serie temporali

Questo file contiene i dati relativi al numero di passeggeri mensili di una compagnia aerea.

Procedere come indicato di seguito:

1. Nell'hub, fare clic su **Crea nuova app**.
2. Aprire l'app e rilasciare il file *Tutorial - Time series decomposition.csv* su di essa.

Preparare e caricare i dati

Per permettere a Qlik Sense di interpretare correttamente il campo YearMonth, potrebbe essere necessario utilizzare Gestione dati per riconoscere il campo come un campo data e non come un campo con valori stringa. In genere questo passaggio è gestito automaticamente, ma in questo caso le date sono presentate nel formato *YYYY-MM*, un po' insolito.

1. In Gestione dati selezionare una tabella e fare clic su .
2. Con il campo *YearMonth* selezionato, fare clic su e impostare il **Tipo campo** su **Data**.
3. Sotto **Formato input**, inserire *YYYY-MM*.
4. Sotto **Formato visualizzazione**, inserire *YYYY-MM* e fare clic su **OK**.
   Il campo dovrebbe ora mostrare l'icona del calendario.
5. Fare clic su **Carica dati**.

Ora si è pronti per iniziare a usare le funzioni STL per rappresentare visivamente i propri dati.

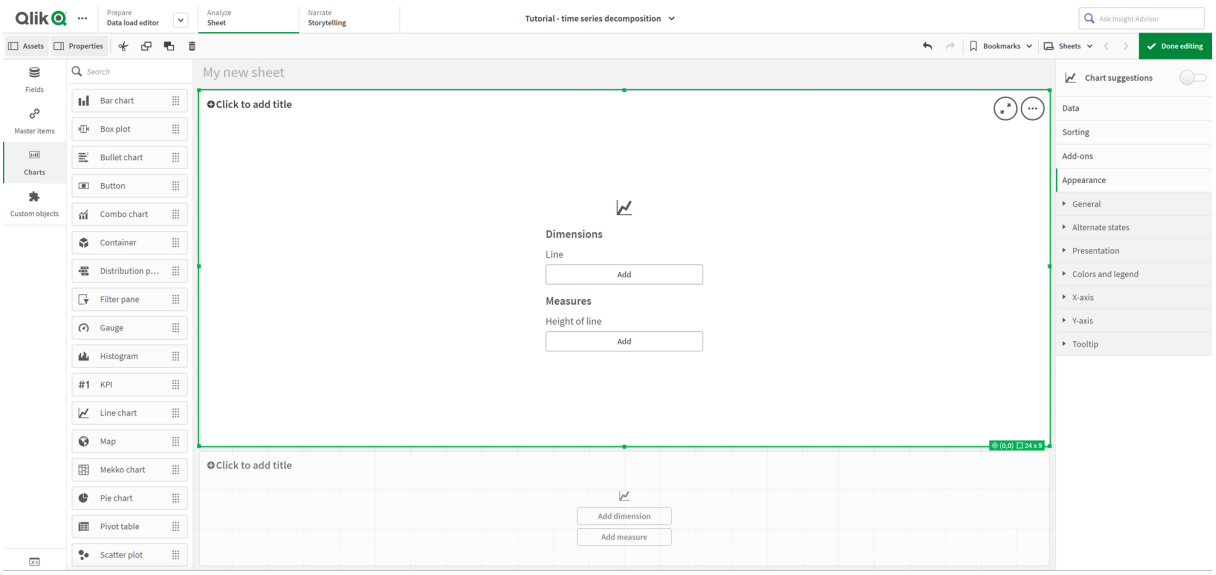
Creare le visualizzazioni

Ora si creeranno due grafici lineari per dimostrare la funzionalità delle funzioni grafico **STL_Trend**, **STL_Seasonal** e **STL_Residual**.

Aprire un nuovo foglio e dargli un titolo.

Aggiungere due grafici lineari al foglio. Ridimensionare e riposizionare i grafici in modo che corrispondano all'immagine seguente.

Qlik SenseContorno griglia del foglio app vuoto



Primo grafico lineare: Tendenza e componenti stagionali

Procedere come indicato di seguito:

1. Aggiungere il titolo *Stagionale e tendenza* al primo grafico lineare.
2. Aggiungere *YearMonth* come dimensione ed etichettarlo *Data*.
3. Aggiungere la seguente misura ed etichettarla *Passeggeri al mese*:
   =Sum(Passengers)
4. Sotto **Dati**, espandere la misura *Passeggeri al mese* e fare clic su **Aggiungi linea di tendenza**.
5. Impostare il **Tipo** su **Lineare**.
   Questa linea di tendenza verrà confrontata con l'output uniformato del componente di tendenza.
6. Aggiungere la seguente misura per tracciare la componente di tendenza ed etichettarla *Tendenza*:
   =STL_Trend(SUM(Passengers), 12)
7. Quindi, aggiungere la seguente misura per tracciare la componente stagionale ed etichettarla *Stagionale*:
   =STL_Seasonal(SUM(Passengers), 12)
8. Sotto **Aspetto** > **Presentazione**, impostare **Barra di scorrimento** su **Nessuna**.
9. Mantenere i colori predefiniti o modificarli in base alle proprie preferenze.

Secondo grafico lineare: Componente residuo

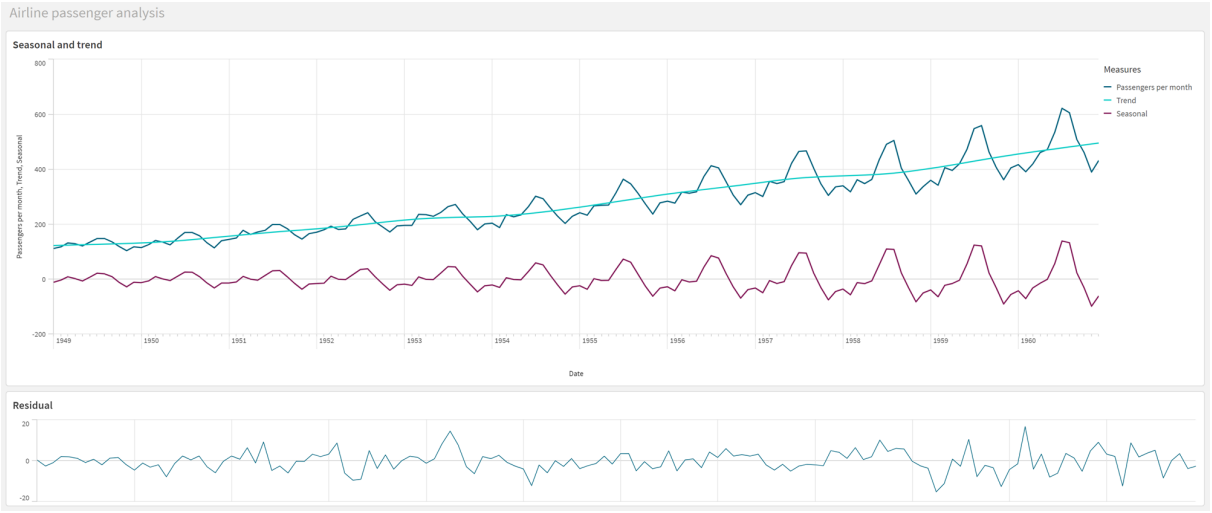
Quindi, configurare il secondo grafico lineare. Questa visualizzazione mostra la componente residua della serie temporale.

Procedere come indicato di seguito:

1. Trascinare un grafico lineare sul foglio. Aggiungere il titolo *Residuo*.
2. Aggiungere *Data* come dimensione.
3. Aggiungere la seguente misura ed etichettarla come *Residua*:
   =STL_Residual(SUM(Passengers), 12)
4. Sotto **Aspetto** > **Presentazione**, impostare **Barra di scorrimento** su **Nessuna**.

Il proprio foglio dovrebbe ora assomigliare a quello qui sotto.

Foglio Qlik Sense per l'analisi dei passeggeri delle compagnie aeree



Interpretazione e spiegazione dei dati

Con le funzioni STL, possiamo ottenere una serie di informazioni strategiche dai dati delle serie temporali.

Componente di tendenza

Le informazioni statistiche nella componente di tendenza sono destagionalizzate. In questo modo è più facile vedere le fluttuazioni generali e non ripetute nel tempo. Rispetto alla linea di tendenza rettilinea e lineare per i *Passeggeri al mese*, la componente di tendenza STL cattura le tendenze in evoluzione. Mostra alcune chiare deviazioni, pur presentando le informazioni in modo leggibile. I comportamenti di uniformità nell'algoritmo STL hanno contribuito a catturare questo aspetto.

Il calo del numero di passeggeri delle compagnie aeree, visibile nel grafico del trend di STL, può essere spiegato come parte dell'impatto economico delle recessioni che si sono verificate negli anni Cinquanta.

Componente stagionale

La componente stagionale priva di trend ha isolato le fluttuazioni ricorrenti nella serie temporale, eliminando le informazioni sulle tendenze generali da questa parte dell'analisi. Abbiamo iniziato con un set di dati composto da aggregazioni anno-mese. Con questi dati, è implicito che stiamo segmentando i dati in granuli di un mese. Definendo un valore di periodo pari a 12, si imposta il grafico per modellare i modelli stagionali nel corso di cicli di un anno (dodici mesi).

I dati mostrano un modello stagionale ripetuto di aumento dei passeggeri delle compagnie aeree nei mesi estivi, seguito da un calo nei mesi invernali. Questo è in linea con l'idea che l'estate sia un periodo molto gettonato per le vacanze e i viaggi. Si nota anche che, nel corso della serie temporale, questi cicli stagionali aumentano drasticamente in ampiezza.

Componente residuo

Il grafico della componente residua mostra tutte le informazioni che non sono state acquisite dalla scomposizione per trend e stagionalità. La componente residua comprende il rumore statistico, ma può anche indicare un'impostazione errata degli argomenti delle funzioni STL di tendenza e stagionali. In generale, se si verificano oscillazioni periodiche nella componente residua del segnale, o se le

informazioni visualizzate non sono chiaramente casuali, di solito si tratta di un segnale che indica la presenza di informazioni nella serie temporale che non sono attualmente acquisite nelle componenti stagionali o di tendenza. In questo caso, è necessario rivedere le definizioni di ciascun argomento della funzione ed eventualmente modificare la periodicità.

Valori smoother

Poiché non sono stati specificati valori per gli smoother di tendenza e stagionali, la funzione utilizzerà i valori predefiniti per questi parametri. In Qlik Sense, i valori predefiniti dello smoother nell'algoritmo STL producono risultati efficaci. Di conseguenza, nella maggior parte dei casi, questi argomenti possono essere lasciati fuori dalle espressioni.

Impostando gli argomenti dello smoother stagionale o di tendenza come 0 in una delle tre funzioni STL , l'algoritmo utilizzerà valori predefiniti, anziché valori pari a 0.

Il valore dello smoother di tendenza utilizza la dimensione specificata nel grafico. Poiché il campo *YearMonth* presenta i dati per mesi, il valore di trend smoother sarà il numero di mesi. Lo smoother stagionale rifletterà la periodicità definita. In questo caso, poiché abbiamo definito che un periodo dura dodici mesi (un anno), il valore dello smoother stagionale è il numero di anni. Ciò può sembrare confuso, ma in realtà significa che per trovare la stagionalità, dobbiamo esaminare un certo numero di stagioni. Questo numero è lo smoother stagionale.

Altre informazioni utili

Dato che i cicli stagionali aumentano di ampiezza nel tempo, un approccio analitico più avanzato potrebbe fare uso di funzioni logaritmiche per creare una scomposizione moltiplicativa. In pratica, in Qlik Sense è possibile creare una semplice misura dell'ampiezza relativa dividendo la componente stagionale per quella di tendenza. In questo modo, si nota che nel corso del tempo i picchi estivi di ciascun ciclo aumentano in ampiezza relativa. L'ampiezza dei punti bassi invernali, tuttavia, non aumenta nel tempo.

5.23 Funzioni di distribuzione statistica

Le funzioni di distribuzione statistica restituiscono le probabilità di accadimento dei diversi risultati possibili per una determinata variabile di input. È possibile utilizzare queste funzioni per calcolare i valori potenziali dei punti dati.

I tre gruppi delle funzioni di distribuzione statistica descritti di seguito sono tutte implementati in Qlik Sense mediante la libreria di funzioni Cephes. Per ottenere riferimenti e informazioni dettagliate sugli algoritmi utilizzati, l'accuratezza e così via, visitare il sito ≤ Cephes library. La libreria di funzioni Cephes è utilizzata su concessione.

- Le funzioni di probabilità calcolano la probabilità nel punto della distribuzione dato dal valore fornito.
    - Le funzioni di frequenza sono utilizzate per le distribuzioni discrete.
    - Le funzioni di densità vengono utilizzate per le funzioni continue.
- Le funzioni Dist calcolano la probabilità cumulativa della distribuzione nel punto della distribuzione

dato dal valore fornito.

- Le funzioni Inv calcolano il valore inverso, data la probabilità cumulativa della distribuzione.

Tutte le funzioni possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche.

Panoramica sulle funzioni di distribuzione statistica

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

BetaDensity

`BetaDensity()` restituisce la probabilità della distribuzione Beta.

```
BetaDensity (value, alpha, beta)
```

BetaDist

`BetaDist()` restituisce la probabilità cumulativa della distribuzione Beta.

```
BetaDist (value, alpha, beta)
```

BetaInv

`BetaINV()` restituisce l'inverso della probabilità cumulativa della distribuzione Beta.

```
BetaInv (prob, alpha, beta)
```

BinomDist

`BinomDist()` restituisce la probabilità cumulativa della distribuzione Binomiale.

```
BinomDist (value, trials, trial_probability)
```

BinomFrequency

`BinomFrequency()` restituisce la distribuzione di probabilità Binomiale.

```
BinomFrequency (value, trials, trial_probability)
```

BinomInv

`BinomInv()` restituisce l'inverso della probabilità cumulativa della distribuzione Binomiale.

```
BinomInv (prob, trials, trial_probability)
```

ChiDensity

`ChiDensity()` restituisce la probabilità a una coda della distribuzione chi^2. La funzione di densità chi^2 è associata a un test chi^2.

```
ChiDensity (value, degrees_freedom)
```

ChiDist

`ChiDist()` restituisce la probabilità a una coda della distribuzione chi^2. La distribuzione chi^2 è associata a un test chi^2.

```
ChiDist (value, degrees_freedom)
```

ChiInv

`ChiInv()` restituisce l'inverso della probabilità a una coda della distribuzione chi^2.

```
ChiInv (prob, degrees_freedom)
```

FDensity

`FDensity()` restituisce la probabilità della distribuzione `F`.

```
FDensity (value, degrees_freedom1, degrees_freedom2)
```

FDist

`FDist()` restituisce la probabilità cumulativa della distribuzione `F`.

```
FDist (value, degrees_freedom1, degrees_freedom2)
```

FInv

`FInv()` restituisce l'inverso della probabilità cumulativa della distribuzione `F`.

```
FInv (prob, degrees_freedom1, degrees_freedom2)
```

GammaDensity

`GammaDensity()` restituisce la probabilità della distribuzione Gamma.

```
GammaDensity (value, k, θ)
```

GammaDist

`GammaDist()` restituisce la probabilità accumulata della distribuzione Gamma.

```
GammaDist (value, k, θ)
```

GammaInv

`GammaInv()` restituisce l'inverso della probabilità cumulativa della distribuzione Gamma.

```
GammaInv (prob, k, θ)
```

NormDist

`NormDist()` restituisce la distribuzione cumulativa normale per la media e la deviazione standard specificate. Se mean = 0 e standard_dev = 1, la funzione restituisce la distribuzione normale standard.

```
NormDist (value, mean, standard_dev)
```

NormInv

`NormInv()` restituisce il valore contrario della distribuzione cumulativa normale per la media e la deviazione standard specificate.

```
NormInv (prob, mean, standard_dev)
```

PoissonDist

`PoissonDist()` restituisce la probabilità cumulativa della distribuzione Poisson.

```
PoissonDist (value, mean)
```

PoissonFrequency

`PoissonFrequency()` restituisce la distribuzione di probabilità di Poisson.

```
PoissonFrequency (value, mean)
```

PoissonInv

`PoissonInv()` restituisce l'inverso della probabilità cumulativa della distribuzione Poisson.

```
PoissonInv (prob, mean)
```

TDensity

`TDensity()` restituisce il valore della funzione di densità `t` per lo studente, dove un valore numerico è un valore calcolato di `t` per il quale la probabilità deve essere elaborata.

```
TDensity (value, degrees_freedom, tails)
```

TDist

`TDist()` restituisce la probabilità per la distribuzione `t` dello studente in cui un valore numerico è un valore calcolato di `t` per il quale deve essere calcolata la probabilità.

```
TDist (value, degrees_freedom, tails)
```

TInv

`TInv()` restituisce il valore `t` della distribuzione `t` dello studente in funzione della probabilità e dei gradi di libertà.

```
TInv (prob, degrees_freedom)
```

Vedere anche:

p *Funzioni di aggregazione statistica (page 392)*

BetaDensity

`BetaDensity()` restituisce la probabilità della distribuzione Beta.

Sintassi:

```
BetaDensity(value, alpha, beta)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
`value`	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore deve essere compreso tra 0 e 1.
`alpha`	Un numero positivo che definisce il primo parametro di forma. È l'esponente della variabile casuale

Argomento	Descrizione
beta	Un numero positivo che definisce il secondo parametro di forma. Indica il numero di gradi di libertà del denominatore.

BetaDist

`BetaDist()` restituisce la probabilità cumulativa della distribuzione Beta.

Sintassi:

```
BetaDist(value, alpha, beta)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
value	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore deve essere compreso tra 0 e 1.
alpha	Un numero positivo che definisce il primo parametro di forma. È l'esponente della variabile casuale
beta	Un numero positivo che definisce il secondo parametro di forma. È l'esponente che controlla la forma della distribuzione.

Questa funzione è correlata alla funzione `BetaInv` nel modo seguente:
`If prob = BetaDist(value, alpha, beta), then BetaInv(prob, alpha, beta) = value`

BetaInv

`BetaINV()` restituisce l'inverso della probabilità cumulativa della distribuzione Beta.

Sintassi:

```
BetaInv(prob, alpha, beta)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
prob	Una probabilità associata alla distribuzione di probabilità Beta. Deve essere un numero compreso tra 0 e 1.
alpha	Un numero positivo che definisce il primo parametro di forma. È l'esponente della variabile casuale
beta	Un numero positivo che definisce il secondo parametro di forma. È l'esponente che controlla la forma della distribuzione.

Questa funzione è correlata alla funzione `BetaDist` nel modo seguente:
`If prob = BetaDist(value, alpha, beta), then BetaInv(prob, alpha, beta) = value`

BinomDist

`BinomDist()` restituisce la probabilità cumulativa della distribuzione Binomiale.

Sintassi:

```
BinomDist(value, trials, trial_probability)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
`value`	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore deve essere un numero intero non inferiore a zero e non superiore al numero di prove.
`trials`	Un numero intero positivo che indica il numero di prove.
`trial_probability`	La probabilità di successo per ogni prova. È sempre un numero compreso tra 0 e 1.

Questa funzione è correlata alla funzione `BinomInv` nel modo seguente:
`If prob = BinomDIST(value, trials, trial_probability), then BinomInv(prob, trials, trial_probability) = value`

BinomFrequency

`BinomFrequency()` restituisce la distribuzione di probabilità Binomiale.

Sintassi:

```
BinomFrequency(value, trials, trial_probability)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
`value`	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore deve essere un numero intero non inferiore a zero e non superiore al numero di prove.
`trials`	Un numero intero positivo che indica il numero di prove
`trial_probability`	La probabilità di successo per ogni prova. È sempre un numero compreso tra 0 e 1.

BinomInv

`BinomInv()` restituisce l'inverso della probabilità cumulativa della distribuzione Binomiale.

Sintassi:

```
BinomInv(prob, trials, trial_probability)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
prob	Una probabilità associata alla distribuzione di probabilità Binomiale. Deve essere un numero compreso tra 0 e 1.
trials	Un numero intero positivo che indica il numero di prove.
trial_probability	La probabilità di successo per ogni prova. È sempre un numero compreso tra 0 e 1.

Questa funzione è correlata alla funzione `BinomDist` nel modo seguente:
`If prob = BinomDist(value, trials, trial_probability), then BinomInv(prob, trials, trial_probability) = value`

ChiDensity

`ChiDensity()` restituisce la probabilità a una coda della distribuzione chi^2. La funzione di densità chi^2 è associata a un test chi^2.

Sintassi:

```
ChiDensity(value, degrees_freedom)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
value	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
degrees_freedom	Numero intero positivo corrispondente al numero di gradi di libertà del numeratore.

ChiDist

`ChiDist()` restituisce la probabilità a una coda della distribuzione chi^2. La distribuzione chi^2 è associata a un test chi^2.

Sintassi:

```
CHIDIST(value, degrees_freedom)
```

Tipo di dati restituiti: numero

Argomenti:

Argomenti

Argomento	Descrizione
value	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
degrees_ freedom	Un numero intero positivo corrispondente al numero di gradi di libertà.

Questa funzione è correlata alla funzione **ChiInv** nel modo seguente:
`If prob = CHIDIST(value,df), then CHIINV(prob, df) = value`

Limiti:

Tutti gli argomenti devono essere numerici, altrimenti viene restituito un valore NULL.

Esempi e risultati:

Esempio	Risultato
CHIDIST(8, 15)	Restituisce 0,9238

ChiInv

`ChiInv()` restituisce l'inverso della probabilità a una coda della distribuzione chi^2.

Sintassi:

```
CHIINV(prob, degrees_freedom)
```

Tipo di dati restituiti: numero

Argomenti:

Argomenti

Argomento	Descrizione
prob	Una probabilità associata alla distribuzione di chi^2. Deve essere un numero compreso tra 0 e 1.
degrees_ freedom	Un numero intero corrispondente al numero di gradi di libertà.

Questa funzione è correlata alla funzione **ChiDist** nel modo seguente:
`If prob = CHIDIST(value,df), then CHIINV(prob, df) = value`

Limiti:

Tutti gli argomenti devono essere numerici, altrimenti viene restituito un valore NULL.

Esempi e risultati:

Esempio	Risultato
CHIINV(0.9237827, 15)	Restituisce 8,0000

FDensity

`FDensity()` restituisce la probabilità della distribuzione `F`.

Sintassi:

```
FDensity(value, degrees_freedom1, degrees_freedom2)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
`value`	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
`degrees_freedom1`	Numero intero positivo corrispondente al numero di gradi di libertà del numeratore.
`degrees_freedom2`	Numero intero positivo corrispondente al numero di gradi di libertà del denominatore.

FDist

`FDist()` restituisce la probabilità cumulativa della distribuzione `F`.

Sintassi:

```
FDist(value, degrees_freedom1, degrees_freedom2)
```

Tipo di dati restituiti: numero

Argomenti:

Argomenti

Argomento	Descrizione
value	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
degrees_freedom1	Numero intero positivo corrispondente al numero di gradi di libertà del numeratore.

Argomento	Descrizione
degrees_ freedom2	Numero intero positivo corrispondente al numero di gradi di libertà del denominatore.

Questa funzione è correlata alla funzione **FInv** nel modo seguente:
`If prob = FDIST(value, df1, df2), then FINV(prob, df1, df2) = value`

Limiti:

Tutti gli argomenti devono essere numerici, altrimenti viene restituito un valore NULL.

Esempi e risultati:

Esempio	Risultato
FDIST(15, 8, 6)	Restituisce 0,0019

FInv

`FInv()` restituisce l'inverso della probabilità cumulativa della distribuzione `F`.

Sintassi:

```
FInv(prob, degrees_freedom1, degrees_freedom2)
```

Tipo di dati restituiti: numero

Argomenti:

Argomenti

Argomento	Descrizione
prob	Una probabilità associata alla distribuzione della probabilità F che deve essere compresa tra 0 e 1.
degrees_ freedom	Un numero intero corrispondente al numero di gradi di libertà.

Questa funzione è correlata alla funzione **FDist** nel modo seguente:
`If prob = FDIST(value, df1, df2), then FINV(prob, df1, df2) = value`

Limiti:

Tutti gli argomenti devono essere numerici, altrimenti viene restituito un valore NULL.

Esempi e risultati:

Esempio	Risultato
FINV(0.0019369, 8, 6)	Restituisce 15,0000

GammaDensity

`GammaDensity()` restituisce la probabilità della distribuzione Gamma.

Sintassi:

```
GammaDensity(value, k, θ)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
`value`	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
`k`	Un numero positivo che definisce il parametro di forma.
`θ`	Un numero positivo che definisce il parametro di scaling.

GammaDist

`GammaDist()` restituisce la probabilità accumulata della distribuzione Gamma.

Sintassi:

```
GammaDist(value, k, θ)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
`value`	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
`k`	Un numero positivo che definisce il parametro di forma.
`θ`	Un numero positivo che definisce il parametro di scaling.

Questa funzione è correlata alla funzione `GammaINV` nel modo seguente:
`If prob = GammaDist(value, k, θ), then GammaInv(prob, k, θ) = value`

GammaInv

`GammaInv()` restituisce l'inverso della probabilità cumulativa della distribuzione Gamma.

Sintassi:

```
GammaInv(prob, k, θ)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
prob	Una probabilità associata alla distribuzione di probabilità Gamma. Deve essere un numero compreso tra 0 e 1.
k	Un numero positivo che definisce il parametro di forma.
θ	Un numero positivo che definisce il parametro di scaling.

Questa funzione è correlata alla funzione `GammaDist` nel modo seguente:
`If prob = GammaDist(value, k, θ), then GammaInv(prob, k, θ) = value`

NormDist

`NormDist()` restituisce la distribuzione cumulativa normale per la media e la deviazione standard specificate. Se mean = 0 e standard_dev = 1, la funzione restituisce la distribuzione normale standard.

Sintassi:

```
NORMDIST(value, [mean], [standard_dev], [cumulative])
```

Tipo di dati restituiti: numero

Argomenti:

Argomenti

Argomento	Descrizione
value	Il valore in corrispondenza del quale si desidera valutare la distribuzione.
mean	Valore opzionale che indica la media aritmetica della distribuzione. Se non si specifica questo argomento, il valore predefinito è 0.
standard_dev	Valore positivo opzionale che indica la deviazione standard della distribuzione. Se non si specifica questo argomento, il valore predefinito è 1.
cumulative	È possibile scegliere di utilizzare una distribuzione normale standard o una distribuzione cumulativa. 0 = distribuzione normale standard 1 = distribuzione cumulativa (valore predefinito)

Questa funzione è correlata alla funzione **NormInv** nel modo seguente:
`If prob = NORMDIST(value, m, sd), then NORMINV(prob, m, sd) = value`

Limiti:

Tutti gli argomenti devono essere numerici, altrimenti viene restituito un valore NULL.

Esempi e risultati:

Esempio	Risultato
NORMDIST(0.5, 0, 1)	Restituisce 0,6915

NormInv

`NormInv()` restituisce il valore contrario della distribuzione cumulativa normale per la media e la deviazione standard specificate.

Sintassi:

```
NORMINV(prob, mean, standard_dev)
```

Tipo di dati restituiti: numero

Argomenti:

Argomenti

Argomento	Descrizione
prob	Una probabilità associata alla distribuzione normale. Deve essere un numero compreso tra 0 e 1.
mean	Un valore che indica la media aritmetica della distribuzione.
standard_dev	Un valore positivo che indica la deviazione standard della distribuzione.

Questa funzione è correlata alla funzione **NormDist** nel modo seguente:
`If prob = NORMDIST(value, m, sd), then NORMINV(prob, m, sd) = value`

Limiti:

Tutti gli argomenti devono essere numerici, altrimenti viene restituito un valore NULL.

Esempi e risultati:

Esempio	Risultato
NORMINV(0.6914625, 0, 1)	Restituisce 0,5000

PoissonDist

`PoissonDist()` restituisce la probabilità cumulativa della distribuzione Poisson.

Sintassi:

```
PoissonDist(value, mean)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
value	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
mean	Un numero positivo che definisce il risultato medio.

Questa funzione è correlata alla funzione PoissonInv nel modo seguente:
`If prob = PoissonDist(value, mean), then PoissonInv(prob, mean) = value`

PoissonFrequency

`PoissonFrequency()` restituisce la distribuzione di probabilità di Poisson.

Sintassi:

```
PoissonFrequency(value, mean)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
value	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
mean	Un numero positivo che definisce il risultato medio.

PoissonInv

`PoissonInv()` restituisce l'inverso della probabilità cumulativa della distribuzione Poisson.

Sintassi:

```
PoissonInv(prob, mean)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
prob	Una probabilità associata alla distribuzione di probabilità Poisson. Deve essere un numero compreso tra 0 e 1.
mean	Un numero positivo che definisce il risultato medio.

Questa funzione è correlata alla funzione PoissonDIST nel modo seguente:
`If prob = PoissonDist(value, mean), then PoissonInv(prob, mean) = value`

TDensity

`TDensity()` restituisce il valore della funzione di densità `t` per lo studente, dove un valore numerico è un valore calcolato di `t` per il quale la probabilità deve essere elaborata.

Sintassi:

```
TDensity(value, degrees_freedom)
```

Tipo di dati restituiti: numero

Argomenti

Argomento	Descrizione
`value`	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
`degrees_ freedom`	Un numero intero positivo corrispondente al numero di gradi di libertà.

TDist

`TDist()` restituisce la probabilità per la distribuzione `t` dello studente in cui un valore numerico è un valore calcolato di `t` per il quale deve essere calcolata la probabilità.

Sintassi:

```
TDist(value, degrees_freedom, tails)
```

Tipo di dati restituiti: numero

Argomenti:

Argomenti

Argomento	Descrizione
value	Il valore in corrispondenza del quale si desidera valutare la distribuzione. Il valore non deve essere negativo.
degrees_ freedom	Un numero intero positivo corrispondente al numero di gradi di libertà.
tails	Deve essere 1 (distribuzione a una coda) o 2 (distribuzione a due code).

Questa funzione è correlata alla funzione **TInv** nel modo seguente:
`If prob = TDIST(value, df ,2), then TINV(prob, df) = value`

Limiti:

Tutti gli argomenti devono essere numerici, altrimenti viene restituito un valore NULL.

Esempi e risultati:

Esempio	Risultato
TDIST(1, 30, 2)	Restituisce 0,3253

TInv

`TInv()` restituisce il valore `t` della distribuzione `t` dello studente in funzione della probabilità e dei gradi di libertà.

Sintassi:

```
TINV(prob, degrees_freedom)
```

Tipo di dati restituiti: numero

Argomenti:

Argomenti

Argomento	Descrizione
prob	Una probabilità a due code associata alla distribuzione t. Deve essere un numero compreso tra 0 e 1.
degrees_ freedom	Un numero intero corrispondente al numero di gradi di libertà.

Limiti:

Tutti gli argomenti devono essere numerici, altrimenti viene restituito un valore NULL.

Questa funzione è correlata alla funzione **TDist** nel modo seguente:
`If prob = TDIST(value, df ,2), then TINV(prob, df) = value.`

Esempi e risultati:

Esempio	Risultato
TINV(0.3253086, 30)	Restituisce 1,0000

5.24 Funzioni di stringa

In questa sezione vengono descritte le funzioni per la gestione e l'elaborazione delle stringhe.

Tutte le funzioni possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche, ad eccezione della funzione **Evaluate** che può essere utilizzata solo nello script di caricamento dei dati.

Panoramica sulle funzioni di stringa

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Capitalize

Capitalize() restituisce la stringa in cui tutte le parole iniziano con la lettera maiuscola.

```
Capitalize (text)
```

Chr

Chr() restituisce il carattere Unicode che corrisponde al numero intero di input.

```
Chr (int)
```

Evaluate

Evaluate() rileva se la stringa di testo di input può essere valutata come espressione di Qlik Sense valida e, in caso affermativo, restituisce il valore dell'espressione come stringa. Se la stringa di input non è un'espressione valida, viene restituito NULL.

```
Evaluate (expression_text)
```

FindOneOf

FindOneOf() ricerca una stringa per individuare la posizione dell'occorrenza di un qualsiasi carattere che fa parte di un set di caratteri forniti. Viene restituita la posizione della prima occorrenza di un qualsiasi carattere dal gruppo di ricerca a meno che non venga fornito un terzo argomento (con un valore superiore a 1). Se non viene trovata una corrispondenza, viene restituito il valore **0**.

```
FindOneOf (text, char_set[, count])
```

Hash128

Hash128() restituisce un hash a 128 bit dei valori di espressione di input combinati. Il risultato è una stringa da 22 caratteri.

```
Hash128 (expr{, expression})
```

Hash160

Hash160() restituisce un hash a 160 bit dei valori di espressione di input combinati. Il risultato è una stringa da 27 caratteri.

```
Hash160 (expr{, expression})
```

Hash256

Hash256() restituisce un hash a 256 bit dei valori di espressione di input combinati. Il risultato è una stringa da 43 caratteri.

```
Hash256 (expr{, expression})
```

Index

Index() ricerca una stringa per individuare la posizione iniziale della n-esima occorrenza di una sottostringa fornita. Un terzo argomento opzionale fornisce il valore di n, che corrisponde a 1 se omesso. Un valore negativo esegue la ricerca dalla fine della stringa. Le posizioni nella stringa sono numerate a partire dal valore **1**.

```
Index  (text, substring[, count])
```

IsJson

IsJson() verifica se una stringa specificata contiene dati JSON (JavaScript Object Notation) validi. È anche possibile convalidare un tipo specifico di dati JSON.

```
IsJson (json [, type])
```

JsonGet

JsonGet() restituisce il percorso di una stringa dati JSON (JavaScript Object Notation). I dati devono essere dati JSON validi ma possono contenere spazi extra o righe nuove.

```
JsonGet (json, path)
```

JsonSet

JsonSet() modifica una stringa contenente dati JSON (JavaScript Object Notation). Può impostare o inserire un valore JSON con la nuova posizione specificata dal percorso. I dati devono essere dati JSON validi ma possono contenere spazi extra o righe nuove.

```
JsonSet(json, path, value)
```

KeepChar

KeepChar() restituisce una stringa costituita dalla prima stringa, 'text', meno uno qualsiasi dei caratteri NON contenuti nella seconda stringa, "keep_chars".

```
KeepChar (text, keep_chars)
```

Left

Left() restituisce una stringa costituita dai primi caratteri (posizionati più a sinistra) della stringa di input, in cui il numero di caratteri viene stabilito dal secondo argomento.

```
Left (text, count)
```

Len

Len() restituisce la lunghezza della stringa di input.

```
Len (text)
```

LevenshteinDist

LevenshteinDist() restituisce la distanza Levenshtein tra due stringhe. Viene definita come il numero minimo di modifiche a un singolo carattere (inserimenti, eliminazioni o sostituzioni) richiesto per cambiare una stringa con un'altra. La funzione è utile per i confronti tra stringhe fuzzy.

```
LevenshteinDist (text1, text2)
```

Lower

Lower() applica il formato minuscolo a tutti i caratteri della stringa di input.

```
Lower (text)
```

LTrim

LTrim() restituisce la stringa di input senza spazi iniziali.

```
LTrim (text)
```

Mid

Mid() restituisce la parte della stringa di input che inizia nella posizione del carattere definito dal secondo argomento, 'start', e che restituisce il numero di caratteri definito dal terzo argomento, 'count'. Se viene omesso 'count', viene restituita la parte rimanente della stringa di input. Il primo carattere nella stringa di input viene contrassegnato con il numero 1.

```
Mid (text, start[, count])
```

Ord

Ord() restituisce il numero del punto del codice Unicode del primo carattere della stringa di input.

```
Ord (text)
```

PurgeChar

PurgeChar() restituisce una stringa costituita dai caratteri contenuti nella stringa di input ('text'), ad eccezione di qualsiasi carattere presente nel secondo argomento ('remove_chars').

```
PurgeChar (text, remove_chars)
```

Repeat

Repeat() crea una stringa costituita dalla stringa di input ripetuta il numero di volte stabilito dal secondo argomento.

```
Repeat (text[, repeat_count])
```

Replace

Replace() restituisce una stringa dopo la sostituzione di tutte le occorrenze di una sottostringa fornita all'interno della stringa di input con un'altra sottostringa. La funzione non è ricorrente e viene applicata da sinistra verso destra.

```
Replace (text, from_str, to_str)
```

Right

Right() restituisce una stringa costituita dagli ultimi caratteri (posizionati più a destra) della stringa di input, in cui il numero di caratteri viene stabilito dal secondo argomento.

```
Right (text, count)
```

RTrim

RTrim() restituisce la stringa di input senza spazi finali.

```
RTrim (text)
```

SubField

SubField() consente di estrarre i componenti della sottostringa da un campo della stringa padre, in cui i campi del record originali sono costituiti da due o più parti separate da un delimitatore.

```
SubField (text, delimiter[, field_no ])
```

SubStringCount

SubStringCount() restituisce il numero di occorrenze della sottostringa specificata nel testo della stringa di input. In mancanza di corrispondenze, viene restituito 0.

```
SubStringCount (text, substring)
```

TextBetween

TextBetween() restituisce il testo nella stringa di input che si trova tra i caratteri specificati come delimitatori.

```
TextBetween (text, delimiter1, delimiter2[, n])
```

Trim

Trim() restituisce la stringa di input senza spazi iniziali e finali.

```
Trim (text)
```

Upper

Upper() applica il carattere maiuscolo a tutti i caratteri della stringa di input per tutti i caratteri di testo nell'espressione. I numeri e i simboli vengono ignorati.

```
Upper (text)
```

Capitalize

Capitalize() restituisce la stringa in cui tutte le parole iniziano con la lettera maiuscola.

Sintassi:

```
Capitalize(text)
```

Tipo di dati restituiti: stringa

Esempio: Espressioni del grafico

Esempio	Risultato
`Capitalize ( 'star trek' )`	Restituisce 'Star Trek'
`Capitalize ( 'AA bb cC Dd')`	Restituisce 'Aa Bb Cc Dd'

Esempio: Script di caricamento

```
Load  String, Capitalize(String) Inline [String rHode iSland washingTon d.C. new york];
```

Risultato

Stringa	Capitalize(String)
rHode iSland	Rhode Island
washingTon d.C.	Washington D.C.
new york	New York

Chr

Chr() restituisce il carattere Unicode che corrisponde al numero intero di input.

Sintassi:

```
Chr(int)
```

Tipo di dati restituiti: stringa

Esempi e risultati:

Esempio	Risultato
Chr(65)	Restituisce la stringa 'A'
Chr(163)	Restituisce la stringa '£'
Chr(35)	Restituisce la stringa '#'

Evaluate

Evaluate() rileva se la stringa di testo di input può essere valutata come espressione di Qlik Sense valida e, in caso affermativo, restituisce il valore dell'espressione come stringa. Se la stringa di input non è un'espressione valida, viene restituito NULL.

Sintassi:

```
Evaluate(expression_text)
```

Tipo di dati restituiti: duale

La funzione stringa non può essere utilizzata nelle espressioni grafiche.

Esempi e risultati:

Esempio di funzione	Risultato
`Evaluate ( 5 * 8 )`	Restituisce '40'

Esempio di script di caricamento

```
Load  Evaluate(String) as Evaluated, String Inline [String 4 5+3 0123456789012345678 Today()
];
```

Risultato

Stringa	Valutata
4	4
5+3	8
0123456789012345678	0123456789012345678
Today()	2022-02-02

FindOneOf

FindOneOf() ricerca una stringa per individuare la posizione dell'occorrenza di un qualsiasi carattere che fa parte di un set di caratteri forniti. Viene restituita la posizione della prima occorrenza di un qualsiasi carattere dal gruppo di ricerca a meno che non venga fornito un terzo argomento (con un valore superiore a 1). Se non viene trovata una corrispondenza, viene restituito il valore **0**.

Sintassi:

```
FindOneOf(text, char_set[, count])
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale.
char_set	Un set di caratteri da ricercare in text.
count	Definisce l'occorrenza di uno qualsiasi dei caratteri da ricercare. Ad esempio, un valore di 2 ricerche per la seconda occorrenza.

Esempio: Espressioni del grafico

Esempio	Risultato
`FindOneOf( 'my example text string', 'et%s')`	Restituisce '4' perché ‘e’ corrisponde al quarto carattere nella stringa di esempio.

Esempio	Risultato
`FindOneOf( 'my example text string', 'et%s', 3)`	Restituisce '12' perché la ricerca viene eseguita per uno qualsiasi dei caratteri e, t, % o s e "t" corrisponde alla terza occorrenza nella posizione 12 della stringa di esempio.
`FindOneOf( 'my example text string', '¤%&')`	Restituisce '0' perché nessun dei caratteri ¤, % o & esiste nella stringa di esempio.

Esempio: Script di caricamento

```
Load * Inline [SearchFor, Occurrence et%s,1 et%s,3 ¤%&,1]
```

Risultato

SearchFor	Occorrenza	FindOneOf('my example text string', SearchFor, Occurrence)
et%s	1	4
et%s	3	12
¤%&	1	0

Hash128

Hash128() restituisce un hash a 128 bit dei valori di espressione di input combinati. Il risultato è una stringa da 22 caratteri.

Sintassi:

```
Hash128(expr{, expression})
```

Tipo di dati restituiti: stringa

Esempio: Espressioni del grafico

Esempio	Risultato
Hash128 ('abc', 'xyz', '123')	Restituisce 'MA&5]6+3=:>:>G%S<U*S2+'.
Hash128 (Region, Year, Month) Note: Region, Year, and Month are table fields.	Restituisce 'G7*=6GKPJ(Z+)^KM?<$'A+'.

Esempio: Script di caricamento

```
Hash_128: Load *,  Hash128(Region, Year, Month) as Hash128; Load * inline [ Region, Year, Month abc, xyz, 123 EU, 2022, 01 UK, 2022, 02 US, 2022, 02 ];
```

Risultato

Area geografica	Anno	Mese	Hash128
abc	xyz	123	MA&5]6+3=:>;>G%S<U*S2+
EU	2022	01	B40^K&[T@!;VB'XR]<5=/$
UK	2022	02	O5T;+1?[B&"F&1//MA[MN!
US	2022	02	C6@#]4#_G-(]J7EQY#KRW0

Hash160

Hash160() restituisce un hash a 160 bit dei valori di espressione di input combinati. Il risultato è una stringa da 27 caratteri.

Sintassi:

```
Hash160(expr{, expression})
```

Tipo di dati restituiti: stringa

Esempio: Espressioni del grafico

Esempio	Risultato
Hash160 ('abc', 'xyz', '123')	Restituisce 'MA&5]6+3=:>;>G%S<U*S2I:`=X*'.
Hash160 (Region, Year, Month) Note: Region, Year, and Month are table fields.	Restituisce 'G7*=6GKPJ (Z+)^KM?<$'AI.)?U$'.

Esempio: Script di caricamento

```
Hash_160: Load *,  Hash160(Region, Year, Month) as Hash160; Load * inline [ Region, Year, Month abc, xyz, 123 EU, 2022, 01 UK, 2022, 02 US, 2022, 02 ];
```

Risultato

Area geografica	Anno	Mese	Hash160
abc	xyz	123	MA&5]6+3=:>;>G%S<U*S2I:`=X*
EU	2022	01	B40^K&[T@!;VB'XR]<5=//_F853
UK	2022	02	O5T;+1?[B&"F&1//MA[MN!T"FWZ
US	2022	02	C6@#]4#_G-(]J7EQY#KRW`@KF+W

Hash256

Hash256() restituisce un hash a 256 bit dei valori di espressione di input combinati. Il risultato è una stringa da 43 caratteri.

Sintassi:

```
Hash256(expr{, expression})
```

Tipo di dati restituiti: stringa

Esempio: Espressioni del grafico

Esempio	Risultato
Hash256 ('abc', 'xyz', '123')	Restituisce 'MA&5]6+3=:>;>G%S<U*S2I:`=X*A.IO*8N\%Y7Q;YEJ'.
Hash256 (Region, Year, Month) Note: Region, Year, and Month are table fields.	Restituisce 'G7*=6GKPJ(Z+)^KM?<$'AI.)?U$#X2RB [:0ZP=+Z`F:'.

Esempio: Script di caricamento

```
Hash_256: Load *,  Hash256(Region, Year, Month) as Hash256; Load * inline [ Region, Year, Month abc, xyz, 123 EU, 2022, 01 UK, 2022, 02 US, 2022, 02 ];
```

Risultato

Area geografica	Anno	Mese	Hash256
abc	xyz	123	MA&5]6+3=:>;>G%S<U*S2I:`=X*A.IO*8N\%Y7Q;YEJ
EU	2022	01	B40^K&[T@!;VB'XR]<5=//_F853?BE6'G&,YH*T'MF)
UK	2022	02	O5T;+1?[B&"F&1//MA[MN!T"FWZT=4\#V`M%6_\0C>4
US	2022	02	C6@#]4#_G-(]J7EQY#KRW`@KF+W-0]`[Z8R+#'")=+0

Index

Index() ricerca una stringa per individuare la posizione iniziale della n-esima occorrenza di una sottostringa fornita. Un terzo argomento opzionale fornisce il valore di n, che corrisponde a 1 se omesso. Un valore negativo esegue la ricerca dalla fine della stringa. Le posizioni nella stringa sono numerate a partire dal valore **1**.

Sintassi:

```
Index(text, substring[, count])
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale.
substring	Una stringa di caratteri da ricercare in text.
count	Definisce l'occorrenza di **substring** da ricercare. Ad esempio, un valore di 2 ricerche per la seconda occorrenza.

Esempi e risultati:

Esempio	Risultato
Index('abcdefg', 'cd')	Restituisce 3
Index('abcdabcd', 'b', 2)	Restituisce 6 (la seconda occorrenza di 'b')
Index('abcdabcd', 'b',-2)	Restituisce 2 (la seconda occorrenza di 'b' iniziando dalla fine)
`Left( Date, Index( Date,'-') -1 )` where **Date** = 1997-07-14	Restituisce 1997
`Mid( Date, Index( Date, '-', 2 ) -2, 2 )` where **Date** = 1997-07-14	Restituisce 07

Esempio: Script

```
T1: Load  *,  index(String, 'cd') as Index_CD,              // returns 3  in Index_CD index
(String, 'b') as Index_B,               // returns 2 in Index_B index(String, 'b', -1) as
Index_B2;           // returns 2 or 6 in Index_B2 Load * inline [ String abcdefg abcdabcd ];
```

IsJson

IsJson() verifica se una stringa specificata contiene dati JSON (JavaScript Object Notation) validi. È anche possibile convalidare un tipo specifico di dati JSON.

Sintassi:

```
value IsJson(json [, type])
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
json	Stringa da testare. Può contenere spazi extra o nuove righe.
type	Argomento opzionale che specifica il tipo di dati JSON da testare. • 'value' (predefinito) • 'object' • 'array' • 'stringa' • 'number' • 'Boolean' • 'null'

Esempio: JSON valido e tipo

Esempio	Risultato
`IsJson('null')`	Restituisce -1 (true)
`IsJson('"abc"', 'value')`	Restituisce -1 (true)
`IsJson('"abc"', 'string')`	Restituisce -1 (true)
`IsJson(123, 'number')`	Restituisce -1 (true)

Esempio: JSON o tipo non valido

Esempio	Risultato	Descrizione
`IsJson('text')`	Restituisce 0 (false)	'text' non è un valore JSON valido
`IsJson('"text"', 'number')`	Restituisce 0 (false)	'"text"' non è un numero JSON valido
`IsJson('"text"', 'text')`	Restituisce 0 (false)	'text' non è un tipo JSON valido

JsonGet

JsonGet() restituisce il percorso di una stringa dati JSON (JavaScript Object Notation). I dati devono essere dati JSON validi ma possono contenere spazi extra o righe nuove.

Sintassi:

```
value JsonGet(json, path)
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
json	Stringa contenente dati JSON.
path	Il percorso deve essere specificato in base a ≤ RFC 6901. Ciò consentirà la ricerca di proprietà all'interno dei dati JSON senza utilizzare funzioni complesse di sottostringa o indice.

Esempio: JSON e percorso valido

Esempio	Risultato
`JsonGet('{"a":{"foo":"bar"},"b":[123,"abc","ABC"]}', '')`	Restituisce '{"a":{"foo":"bar"},"b":[123,"abc","ABC"]}'
`JsonGet('{"a":{"foo":"bar"},"b":[123,"abc","ABC"]}', '/a')`	Restituisce '{"foo":"bar"}'
`JsonGet('{"a":{"foo":"bar"},"b":[123,"abc","ABC"]}', '/a/foo')`	Restituisce '"bar"'
`JsonGet('{"a":{"foo":"bar"},"b":[123,"abc","ABC"]}', '/b')`	Restituisce '[123,"abc","ABC"]'
`JsonGet('{"a":{"foo":"bar"},"b":[123,"abc","ABC"]}', '/b/0')`	Restituisce '123'
`JsonGet('{"a":{"foo":"bar"},"b":[123,"abc","ABC"]}', '/b/1')`	Restituisce '"abc"'
`JsonGet('{"a":{"foo":"bar"},"b":[123,"abc","ABC"]}', '/b/2')`	Restituisce '"ABC"'

Esempio: JSON o percorso non valido

Esempio	Risultato	Descrizione
`JsonGet('{"a":"b"}','/b')`	Restituisce null	Il percorso non punta a una parte valida dei dati JSON.
`JsonGet('{"a"}','/a')`	Restituisce null	I dati JSON non sono JSON validi (il membro "a" non ha un valore)

JsonSet

JsonSet() modifica una stringa contenente dati JSON (JavaScript Object Notation). Può impostare o inserire un valore JSON con la nuova posizione specificata dal percorso. I dati devono essere dati JSON validi ma possono contenere spazi extra o righe nuove.

Sintassi:

```
value JsonSet(json, path, value)
```

Tipo di dati restituiti: duale

Argomenti

Argomento	Descrizione
json	Stringa contenente dati JSON.
path	Il percorso deve essere specificato in base a ≤ RFC 6901. Ciò consentirà la creazione di proprietà all'interno dei dati JSON senza l'utilizzo di funzione complesse sottostringa o indice e concatenazione.
value	Il nuovo valore della stringa in formato JSON.

Esempio: JSON, percorso e valore valido

Esempio	Risultato
`JsonSet('{}','/a','"b"')`	Restituisce '{"a":"b"}'
`JsonSet('[]','/0','"x"')`	Restituisce '["x"]'
`JsonSet('"abc"','','123')`	Restituisce 123

Esempio: JSON, percorso o valore non valido

Esempio	Risultato	Descrizione
`JsonSet ('"abc"','/x','123')`	Restituisce null	Il percorso non punta a una parte valida dei dati JSON.
`JsonSet('{"a": {"b":"c"}}','a/b','"x"')`	Restituisce null	Il percorso non è valido.
`JsonSet(' {"a":"b"}','/a','abc')`	Restituisce null	Il valore non è un JSON valido. Una stringa deve essere racchiusa tra virgolette.

KeepChar

KeepChar() restituisce una stringa costituita dalla prima stringa, 'text', meno uno qualsiasi dei caratteri NON contenuti nella seconda stringa, "keep_chars".

Sintassi:

```
KeepChar(text, keep_chars)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale.
keep_chars	Una stringa contenente i caratteri in text da conservare.

Esempio: Espressioni del grafico

Esempio	Risultato
`KeepChar ( 'a1b2c3','123' )`	Restituisce '123'.
`KeepChar ( 'a1b2c3','1234' )`	Restituisce '123'.
`KeepChar ( 'a1b22c3','1234' )`	Restituisce '1223'.
`KeepChar ( 'a1b2c3','312' )`	Restituisce '123'.

Esempio: Script di caricamento

```
T1: Load  *,  keepchar(String1, String2) as KeepChar;   Load * inline [ String1, String2
'a1b2c3', '123' ];
```

Risultati

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione *KeepChar* nello script di caricamento.

String1	String2	KeepChar
a1b2c3	123	123

Vedere anche:

p *PurgeChar (page 1453)*

Left

Left() restituisce una stringa costituita dai primi caratteri (posizionati più a sinistra) della stringa di input, in cui il numero di caratteri viene stabilito dal secondo argomento.

Sintassi:

```
Left(text, count)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomento	Descrizione
text	La stringa originale.
count	Definisce il numero di caratteri da includere partendo dal lato sinistro della stringa **text**.

Esempio: Espressione del grafico

Esempio	Risultato
`Left('abcdef', 3)`	Restituisce 'abc'

Esempio: Script di caricamento

```
T1: Load  *,  left(Text,Start) as Left;        Load * inline [ Text, Start 'abcdef', 3 '2021-07-14', 4 '2021-07-14', 2 ];
```

Risultato

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione *Left* nello script di caricamento.

Testo	Inizio	Left
abcdef	3	abc
2021-07-14	4	2021
2021-07-14	2	20

p Vedere anche *Index (page 1442)*, che consente un'analisi della stringa più complessa.

Len

Len() restituisce la lunghezza della stringa di input.

Sintassi:

```
Len(text)
```

Tipo di dati restituiti: numero intero

Esempio: Espressione del grafico

Esempio	Risultato
`Len('Peter')`	Restituisce '5'

Esempio: Script di caricamento

```
T1: Load String, First&Second as NewString; Load *, mid(String,len(First)+1) as Second; Load *, upper(left(String,1)) as First; Load  * inline [ String this is a sample text string capitalize first letter only ];
```

Risultato

Stringa	NewString
this is a sample text string	This is a sample text string
capitalize first letter only	Capitalize first letter only

LevenshteinDist

LevenshteinDist() restituisce la distanza Levenshtein tra due stringhe. Viene definita come il numero minimo di modifiche a un singolo carattere (inserimenti, eliminazioni o sostituzioni) richiesto per cambiare una stringa con un'altra. La funzione è utile per i confronti tra stringhe fuzzy.

Sintassi:

```
LevenshteinDist(text1, text2)
```

Tipo di dati restituiti: numero intero

Esempio: Espressione del grafico

Esempio	Risultato
`LevenshteinDist('Kitten','Sitting')`	Restituisce '3'

Esempio: Script di caricamento

Script di caricamento

```
T1: Load *, recno() as ID; Load 'Silver' as String_1,* inline [ String_2 Sliver SSiver SSiveer ];  T1: Load *, recno()+3 as ID; Load 'Gold' as String_1,* inline [ String_2 Bold Bool Bond ];  T1: Load *, recno()+6 as ID; Load 'Ove' as String_1,* inline [ String_2 Ove Uve Üve ];  T1: Load *, recno()+9 as ID; Load 'ABC' as String_1,* inline [ String_2 DEFG abc ビビビ ];  set nullinterpret = '<NULL>'; T1: Load *, recno()+12 as ID; Load 'X' as String_1,* inline [ String_2 '' <NULL> 1 ];  R1: Load  ID,  String_1,  String_2,  LevenshteinDist(String_1, String_2) as LevenshteinDistance  resident T1;  Drop table T1;
```

Risultato

ID	String_1	String_2	LevenshteinDistance
1	Silver	Sliver	2
2	Silver	SSiver	2
3	Silver	SSiveer	3
4	Gold	Grassetto	1
5	Gold	Bool	3
6	Gold	Bond	2
7	Ove	Ove	0
8	Ove	Uve	1
9	Ove	Üve	1
10	ABC	DEFG	4
11	ABC	abc	3
12	ABC	ビビビ	3
13	X		1
14	X	-	1
15	X	1	1

Lower

Lower() applica il formato minuscolo a tutti i caratteri della stringa di input.

Sintassi:

```
Lower(text)
```

Tipo di dati restituiti: stringa

Esempio: Espressione del grafico

Esempio	Risultato
`Lower('abcD')`	Restituisce 'abcd'

Esempio: Script di caricamento

```
Load  String, Lower(String) Inline [String rHode iSland washingTon d.C. new york];
```

Risultato

Stringa	Lower(String)
rHode iSland	rhode island
washingTon d.C.	washington d.c.
new york	new york

LTrim

LTrim() restituisce la stringa di input senza spazi iniziali.

Sintassi:

```
LTrim(text)
```

Tipo di dati restituiti: stringa

Esempio: Espressioni del grafico

Esempio	Risultato
`LTrim( ' abc' )`	Restituisce 'abc'
`LTrim( 'abc ' )`	Restituisce 'abc '

Esempio: Script di caricamento

```
Set verbatim=1;  T1:  Load *, len(LtrimString) as LtrimStringLength;       Load *, ltrim
(String) as LtrimString; Load *, len(String) as StringLength;                Load * Inline [
String '   abc    ' ' def  '];
```

L'istruzione "Set verbatim=1" è inclusa nell'esempio per assicurare che gli spazi non vengano automaticamente tagliati prima della dimostrazione della funzione ltrim. Vedere Verbatim (page 205) per ulteriori informazioni.

Risultato

Stringa	StringLength	LtrimStringLength
def	6	5
abc	10	7

Vedere anche:

p *RTrim (page 1457)*

Mid

Mid() restituisce la parte della stringa di input che inizia nella posizione del carattere definito dal secondo argomento, 'start', e che restituisce il numero di caratteri definito dal terzo argomento, 'count'. Se viene omesso 'count', viene restituita la parte rimanente della stringa di input. Il primo carattere nella stringa di input viene contrassegnato con il numero 1.

Sintassi:

```
Mid(text, start[, count])
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale.
start	Numero intero che definisce la posizione del primo carattere di text da includere.
count	Definisce la lunghezza della stringa di output. Se omesso, verranno inclusi tutti i caratteri a partire dalla posizione definita da **start**.

Esempio: Espressioni del grafico

Esempio	Risultato
`Mid('abcdef',3 )`	Restituisce 'cdef'
`Mid('abcdef',3, 2 )`	Restituisce 'cd'

Esempio: Script di caricamento

```
T1: Load *,  mid(Text,Start) as Mid1,            mid(Text,Start,Count) as Mid2;     Load *
inline [ Text, Start, Count 'abcdef', 3, 2 'abcdef', 2, 3 '210714', 3, 2 '210714', 2, 3 ];
```

Risultato

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione *Mid* nello script di caricamento.

Testo	Inizio	Mid1	Conteggio	Mid2
abcdef	2	bcdef	3	bcd
abcdef	3	cdef	2	cd
210714	2	10714	3	107
210714	3	0714	2	07

Vedere anche:

p *Index (page 1442)*

Ord

Ord() restituisce il numero del punto del codice Unicode del primo carattere della stringa di input.

Sintassi:

```
Ord(text)
```

Tipo di dati restituiti: numero intero

Esempi e risultati:

Esempio: Espressione del grafico

Esempio	Risultato
`Ord('A')`	Restituisce il numero intero 65.
`Ord('Ab')`	Restituisce il numero intero 65.

Esempio: Script di caricamento

```
//Guqin (Chinese: 古琴) - 7-stringed zithers T2: Load *,  ord(Chinese) as OrdUnicode,
        ord(Western) as OrdASCII;                    Load * inline [ Chinese, Western 古琴,
Guqin ];
```

Risultato:

Cinese	Occidentale	OrdASCII	OrdUnicode
古琴	Guqin	71	21476

PurgeChar

PurgeChar() restituisce una stringa costituita dai caratteri contenuti nella stringa di input ('text'), ad eccezione di qualsiasi carattere presente nel secondo argomento ('remove_ chars').

Sintassi:

```
PurgeChar(text, remove_chars)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale.
remove_chars	Una stringa contenente i caratteri in text da rimuovere.

Tipo di dati restituiti: stringa

Esempio: Espressioni del grafico

Esempio	Risultato
`PurgeChar ( 'a1b2c3','123' )`	Restituisce 'abc'.
`PurgeChar ( 'a1b2c3','312' )`	Restituisce 'abc'.

Esempio: Script di caricamento

```
T1: Load  *,  purgechar(String1, String2) as PurgeChar;   Load * inline [ String1, String2
'a1b2c3', '123' ];
```

Risultati

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione *PurgeChar* nello script di caricamento.

String1	String2	PurgeChar
a1b2c3	123	abc

Vedere anche:

p *KeepChar (page 1446)*

Repeat

Repeat() crea una stringa costituita dalla stringa di input ripetuta il numero di volte stabilito dal secondo argomento.

Sintassi:

```
Repeat(text[, repeat_count])
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale.
repeat_count	Definisce il numero di volte in cui i caratteri della stringa **text** devono essere ripetuti nella stringa di output.

Esempio: Espressione del grafico

Esempio	Risultato
`Repeat( ' * ', rating )` when **rating** = 4	Restituisce '****'

Esempio: Script di caricamento

```
T1: Load *, repeat(String,2) as Repeat; Load * inline [ String hello world! hOw aRe you? ];
```

Risultato

Stringa	Ripeti
hello world!	hello world!hello world!
hOw aRe you?	hOw aRe you?hOw aRe you?

Replace

Replace() restituisce una stringa dopo la sostituzione di tutte le occorrenze di una sottostringa fornita all'interno della stringa di input con un'altra sottostringa. La funzione non è ricorrente e viene applicata da sinistra verso destra.

Sintassi:

```
Replace(text, from_str, to_str)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale.
from_str	Una stringa che può ricorrere una o più volte all'interno della stringa di input **text**.
to_str	La stringa che sostituirà tutte le occorrenze di **from_str** all'interno della stringa **text**.

Esempi e risultati:

Esempio	Risultato
`Replace('abccde','cc','xyz')`	Restituisce 'abxyzde'

Vedere anche:

Right

Right() restituisce una stringa costituita dagli ultimi caratteri (posizionati più a destra) della stringa di input, in cui il numero di caratteri viene stabilito dal secondo argomento.

Sintassi:

```
Right(text, count)
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale.
count	Definisce il numero di caratteri da includere partendo dal lato destro della stringa **text**.

Esempio: Espressione del grafico

Esempio	Risultato
`Right('abcdef', 3)`	Restituisce 'def'

Esempio: Script di caricamento

```
T1: Load  *,  right(Text,Start) as Right;        Load * inline [ Text, Start 'abcdef', 3
'2021-07-14', 4 '2021-07-14', 2 ];
```

Risultato

Tabella Qlik Sense che mostra l'output derivante dall'utilizzo della funzione *Right* nello script di caricamento.

Testo	Inizio	Right
abcdef	3	def
2021-07-14	4	7-14
2021-07-14	2	14

RTrim

RTrim() restituisce la stringa di input senza spazi finali.

Sintassi:

```
RTrim(text)
```

Tipo di dati restituiti: stringa

Esempio: Espressioni del grafico

Esempio	Risultato
`RTrim( ' abc' )`	Restituisce ' abc'
`RTrim( 'abc ' )`	Restituisce 'abc'

Esempio: Script di caricamento

```
Set verbatim=1;    T1:  Load *, len(RtrimString) as RtrimStringLength;         Load *, rtrim
(String) as RtrimString; Load *, len(String) as StringLength;              Load * Inline [
String '   abc    ' ' def  '];
```

L'istruzione "Set verbatim=1" è inclusa nell'esempio per assicurare che gli spazi non vengano automaticamente tagliati prima della dimostrazione della funzione rtrim. Vedere Verbatim (page 205) per ulteriori informazioni.

Risultato

Stringa	StringLength	RtrimStringLength
def	6	4
abc	10	6

Vedere anche:

p *LTrim (page 1451)*

SubField

SubField() consente di estrarre i componenti della sottostringa da un campo della stringa padre, in cui i campi del record originali sono costituiti da due o più parti separate da un delimitatore.

La funzione **Subfield()**, ad esempio, consente di estrarre il nome di battesimo e il cognome da un elenco di record composto da nomi completi, le parti del componente di un nome del percorso o di estrarre dati da tabelle separate da virgole.

Se si utilizza la funzione **Subfield()** in un'istruzione **LOAD** escludendo il parametro field_no opzionale, per ogni sottostringa verrà generato un record completo. Se diversi campi vengono caricati utilizzando **Subfield()**, vengono creati i prodotti cartesiani di tutte le combinazioni.

Sintassi:

```
SubField(text, delimiter[, field_no ])
```

Tipo di dati restituiti: stringa

Argomenti:

Argomenti

Argomento	Descrizione
text	La stringa originale. Può trattarsi di un testo codificato in forma rigida, una variabile, un'espansione del segno del dollaro o un'altra espressione.
delimiter	Un carattere all'interno del **text** di input che divide la stringa in parti del componente.
field_no	Il terzo argomento opzionale è un numero intero che specifica la sottostringa della stringa principale **text** che verrà restituita. Utilizzare il valore 1 per restituire la prima sottostringa, 2 per restituire la seconda sottostringa e così via. • Se **field_no** è un valore positivo, le sottostringhe vengono estratte da sinistra a destra. • Se **field_no** è un valore negativo, le sottostringhe vengono estratte da destra a sinistra.

È possibile utilizzare SubField() invece di utilizzare combinazioni di funzioni complesse, come ad esempio Len(), Right(), Left(), Mid() e altre funzioni delle stringhe.

Esempi: Script ed espressioni del grafico usando SubField

Esempi - script ed espressioni del grafico

Esempi base

Esempio	Risultato
`SubField(S, ';' ,2)`	Restituisce 'cde' se **S** è 'abc;cde;efg'.
`SubField(S, ';' ,1)`	Restituisce una stringa vuota se **S** è una stringa vuota.
`SubField(S, ';' ,1)`	Restituisce una stringa vuota se **S** è ';'.

Esempio	Risultato
Si supponga di disporre di una variabile contenente il nome di percorso vMyPath, `Set vMyPath=\Users\ext_jrb\Documents\Qlik\Sense\Apps;.`	In un grafico testo e immagine, è possibile aggiungere una misura come: `SubField(vMyPath, '\',-3)`, che restituisce 'Qlik', poiché si tratta della terza sottostringa a partire dall'estremità destra della variabile vMyPath.

Esempio script 1

Script di caricamento

Caricare le espressioni di script e i dati seguenti nell'editor caricamento dati.

```
FullName:                                    LOAD * inline [ Name 'Dave Owen' 'Joe Tem' ];  SepNames:  Lo
(Name, ' ',1) as FirstName, SubField(Name, ' ',-1) as Surname Resident FullName; Drop Table
FullName;
```

Creazione di una visualizzazione

Creare una visualizzazione tabella in un foglio Qlik Sense con **Name**, **FirstName** e **SurName** come dimensioni.

Risultato

Name	FirstName	SurName
Dave Owen	Dave	Owen
Joe Tem	Joe	Tem

Spiegazione

La funzione **SubField()** estrae la prima sottostringa **Name** impostando l'argomento **field_no** su 1. Dato che il valore di **field_no** è positivo, viene seguito un ordine da sinistra a destra per l'estrazione della sottostringa. Una seconda chiamata funzione estrae la seconda sottostringa impostando l'argomento **field_no** a -1, che estrae la sottostringa seguendo un ordine da desta a sinistra.

Esempio script 2

Script di caricamento

Caricare le espressioni di script e i dati seguenti nell'editor caricamento dati.

```
LOAD DISTINCT Instrument, SubField(Player,',') as Player, SubField(Project,',') as Project;
Load * inline [ Instrument|Player|Project Guitar|Neil,Mike|Music,Video Guitar|Neil|Music,OST
Synth|Neil,Jen|Music,Video,OST Synth|Jo|Music Guitar|Neil,Mike|Music,OST ] (delimiter is '|');
```

Creazione di una visualizzazione

Creare una visualizzazione tabella in un foglio Qlik Sense con **Instrument**, **Player** e **Project** come dimensioni.

Risultato

Instrument	Player	Project
Guitar	Mike	Music
Guitar	Mike	Video
Guitar	Mike	OST
Guitar	Neil	Music
Guitar	Neil	Video
Guitar	Neil	OST
Synth	Jen	Music
Synth	Jen	Video
Synth	Jen	OST
Synth	Jo	Music
Synth	Neil	Music
Synth	Neil	Video
Synth	Neil	OST

Spiegazione

Questo esempio mostra come l'utilizzo di più istanze della funzione **Subfield()**, ciascuna delle quali non include il parametro field_no, dall'interno della stessa istruzione **LOAD** crei prodotti cartesiani di tutte le combinazioni. L'opzione **DISTINCT** viene utilizzata per evitare la creazione di record duplicati.

SubStringCount

SubStringCount() restituisce il numero di occorrenze della sottostringa specificata nel testo della stringa di input. In mancanza di corrispondenze, viene restituito 0.

Sintassi:

```
SubStringCount(text, sub_string)
```

Tipo di dati restituiti: numero intero

Argomenti:

Argomento	Descrizione
text	La stringa originale.
sub_string	Una stringa che può ricorrere una o più volte all'interno della stringa di input **text**.

Esempio: Espressioni del grafico

Esempio	Risultato
`SubStringCount ( 'abcdefgcdxyz', 'cd' )`	Restituisce '2'
`SubStringCount ( 'abcdefgcdxyz', 'dc' )`	Restituisce '0'

Esempio: Script di caricamento

```
T1: Load *,  substringcount(upper(Strings),'AB') as SubStringCount_AB; Load * inline [ Strings
ABC:DEF:GHI:AB:CD:EF:GH aB/cd/ef/gh/Abc/abandoned ];
```

Risultato

Stringhe	SubStringCount_AB
aB/cd/ef/gh/Abc/abandoned	3
ABC:DEF:GHI:AB:CD:EF:GH	2

TextBetween

TextBetween() restituisce il testo nella stringa di input che si trova tra i caratteri specificati come delimitatori.

Sintassi:

```
TextBetween(text, delimiter1, delimiter2[, n])
```

Tipo di dati restituiti: stringa

Argomenti:

Argomento	Descrizione
text	La stringa originale.
delimiter1	Specifica il primo carattere di delimitazione (o stringa) da ricercare in **text**.
delimiter2	Specifica il secondo carattere di delimitazione (o stringa) da ricercare in **text**.
n	Definisce l'occorrenza della coppia di delimitazione in cui eseguire la ricerca. Ad esempio, un valore di 2 restituisce i caratteri tra la seconda occorrenza di delimiter1 e la seconda occorrenza di delimiter2.

Esempio: Espressioni del grafico

Esempio	Risultato
`TextBetween('<abc>', '<', '>')`	Restituisce 'abc'
`TextBetween('<abc><de>', '<', '>',2)`	Restituisce 'de'

Esempio	Risultato
`TextBetween('abc', '<', '>')` `TextBetween('<a<b', '<', '>')`	Entrambi gli esempi restituiscono NULL. Se una qualsiasi parte del delimitatore non viene trovata nella stringa, viene restituito NULL.
`TextBetween('<>', '<', '>')`	Restituisce una stringa con lunghezza zero.
`TextBetween('<abc>', '<', '>', 2)`	Restituisce NULL, in quanto n è superiore al numero di occorrenze dei delimitatori.

Esempio: Script di caricamento

```
Load *, textbetween(Text,'<','>') as TextBetween, textbetween(Text,'<','>',2) as
SecondTextBetween; Load * inline [ Text <abc><de> <def><ghi><jkl> ];
```

Risultato

Testo	TextBetween	SecondTextBetween
<abc><de>	abc	de
<def><ghi><jkl>	def	ghi

Trim

Trim() restituisce la stringa di input senza spazi iniziali e finali.

Sintassi:

```
Trim(text)
```

Tipo di dati restituiti: stringa

Esempi e risultati:

Esempio: Espressione del grafico

Esempio	Risultato
`Trim( ' abc' )`	Restituisce 'abc'
`Trim( 'abc ' )`	Restituisce 'abc'
`Trim( ' abc ' )`	Restituisce 'abc'

Esempio: Script di caricamento

```
Set verbatim=1;                                          T1: Load *, len(TrimString) as TrimStringLength;
(String) as TrimString; Load *, len(String) as StringLength;                Load * inline [
String '   abc    ' ' def  '](delimiter is '\t');
```

L'istruzione "Set verbatim=1" è inclusa nell'esempio per assicurare che gli spazi non vengano automaticamente tagliati prima della dimostrazione della funzione trim. Vedere Verbatim (page 205) per ulteriori informazioni.

Risultato:

Stringa	StringLength	TrimStringLength
def	6	3
abc	10	3

Upper

Upper() applica il carattere maiuscolo a tutti i caratteri della stringa di input per tutti i caratteri di testo nell'espressione. I numeri e i simboli vengono ignorati.

Sintassi:

```
Upper(text)
```

Tipo di dati restituiti: stringa

Esempio: Espressione del grafico

Esempio	Risultato
`Upper(' abcD')`	Restituisce 'ABCD'

Esempio: Script di caricamento

```
Load  String,Upper(String) Inline [String rHode iSland washingTon d.C. new york];
```

Risultato

Stringa	Upper(String)
rHode iSland	RHODE ISLAND
washingTon d.C.	WASHINGTON D.C.
new york	NEW YORK

5.25 Funzioni di sistema

Le funzioni di sistema forniscono funzioni per accedere alle proprietà del sistema, del dispositivo e delle app Qlik Sense.

Prospetto delle funzioni di sistema

Alcune funzioni vengono ulteriormente descritte dopo la panoramica. Per tali funzioni, è inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Author()

Questa funzione restituisce una stringa contenente la proprietà Author dell'app attuale. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

Non è possibile impostare la proprietà Author nella versione attuale di Qlik Sense. Se si esegue la migrazione di un documento QlikView, la proprietà Author verrà conservata.

ClientPlatform()

Questa funzione restituisce la stringa dell'agente utente del browser client. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

Esempio:

```
Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/35.0.1916.114 Safari/537.36
```

ComputerName

Questa funzione restituisce una stringa contenente il nome del computer come restituito dal sistema operativo. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

Se il nome del computer ha più di 15 caratteri, la stringa conterrà solo i primi 15 caratteri.

```
ComputerName( )
```

DocumentName

Questa funzione di script restituisce una stringa contenente il nome dell'app Qlik Sense attuale, senza percorso ma con l'estensione. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

```
DocumentName( )
```

DocumentPath

Questa funzione restituisce una stringa contenente il percorso completo dell'app Qlik Sense attuale. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

```
DocumentPath( )
```

Questa funzione non è supportata in modalità standard. .

DocumentTitle

Questa funzione restituisce una stringa contenente il titolo dell'app Qlik Sense attuale. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

```
DocumentTitle( )
```

EngineVersion

Questa funzione restituisce la versione completa dell'engine di Qlik Sense sotto forma di stringa.

```
EngineVersion ()
```

GetCollationLocale

Questa funzione di script restituisce il nome della lingua delle impostazioni locali delle regole di confronto utilizzato. Se la variabile CollationLocale non è stata impostata, vengono restituite le impostazioni locali effettive del computer dell'utente.

```
GetCollationLocale( )
```

GetObjectField

GetObjectField() restituisce il nome della dimensione. **Index** è un numero intero opzionale che indica la dimensione da restituire.

```
GetObjectField - funzione per grafici([index])
```

GetRegistryString

Questa funzione restituisce il valore di una chiave nel registro di Windows. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

```
GetRegistryString(path, key)
```

Questa funzione non è supportata in modalità standard. .

IsPartialReload

Questa funzione restituisce- 1 (True) se l'operazione di ricaricamento attuale è parziale, altrimenti restituisce 0 (False).

```
IsPartialReload ()
```

InObject

La funzione grafico **InObject()** valuta se l'oggetto corrente è contenuto o meno all'interno di un altro oggetto con l'ID specificato nell'argomento funzione. L'oggetto può essere un foglio o una visualizzazione.

```
InObject - funzione per grafici(id_str)
```

ObjectId

La funzione del grafico **ObjectId()** restituisce l'ID dell'oggetto nel quale è valutata l'espressione. La funzione accetta un argomento opzionale che specifica il tipo di oggetto a cui si riferisce la funzione. L'oggetto può essere un foglio o una visualizzazione. La funzione è disponibile solo nelle espressioni del grafico.

```
ObjectId - funzione per grafici([object_type_str])
```

OSUser

Questa funzione restituisce una stringa contenente il nome dell'utente attualmente connesso. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

```
OSUser( )
```

In Qlik Sense Desktop e Qlik Sense Mobile Client Managed questa funzione restituisce sempre 'Personal\Me'.

ProductVersion

Questa funzione restituisce la versione e il numero di build completi di Qlik Sense sotto forma di stringa.

Questa funzione è deprecata e sostituita da **EngineVersion()**.

```
ProductVersion ()
```

ReloadTime

Questa funzione restituisce un'indicazione di data/ora relativa al completamento dell'ultimo caricamento di dati. Può essere utilizzato sia nello script di caricamento dei dati che in un'espressione grafica.

```
ReloadTime( )
```

StateName

StateName() restituisce il nome dello stato alternato della visualizzazione in cui è utilizzata. StateName può essere utilizzata, ad esempio, per creare visualizzazioni con colori e testo dinamici che mostrano i cambiamenti di stato di una visualizzazione. Questa funzione può essere utilizzata nelle espressioni dei grafici, ma non può essere utilizzata per determinare lo stato a cui si riferisce l'espressione.

```
StateName - funzione per grafici()
```

EngineVersion

Questa funzione restituisce la versione completa dell'engine di Qlik Sense sotto forma di stringa.

Sintassi:

```
EngineVersion()
```

InObject - funzione per grafici

La funzione grafico **InObject()** valuta se l'oggetto corrente è contenuto o meno all'interno di un altro oggetto con l'ID specificato nell'argomento funzione. L'oggetto può essere un foglio o una visualizzazione.

Questa funzione può essere utilizzata per mostrare la gerarchia degli oggetti in un foglio, dall'oggetto di livello superiore del foglio alle visualizzazioni nidificate all'interno di altre visualizzazioni. Questa funzione può essere utilizzata insieme alle funzioni **if** e **ObjectId** per creare una navigazione personalizzata nelle app.

Sintassi:

```
InObject(id_str)
```

Tipo di dati restituiti: Booleano

In Qlik Sense, il valore booleano vero è rappresentato da -1 e il valore falso è rappresentato da 0.

Argomenti

Argomento	Descrizione
id_str	Un valore stringa che rappresenta l'ID dell'oggetto da valutare.

L'ID del foglio può essere ottenuto dall'URL dell'app. Per le visualizzazioni, utilizzare le opzioni **Sviluppatore** per identificare l'ID dell'oggetto e la stringa di testo del tipo di oggetto.

Procedere come indicato di seguito:

1. In modalità analisi, aggiungere il seguente testo all'URL:
   /options/developer
2. Fare clic con il pulsante destro del mouse su una visualizzazione e fare clic su **Sviluppatore**.
3. Sotto **Proprietà**, ottenere l'ID oggetto dall'intestazione finestra di dialogo, oltre al tipo di oggetto dalla proprietà **"qType"**.

Limiti:

Questa funzione può dare risultati imprevisti se richiamata in un oggetto (ad esempio, un pulsante) all'interno di un contenitore che rappresenta una voce principale. Questa limitazione si applica anche alle voci principali della casella di filtro, che rappresentano contenitori di diverse caselle di elenco. Ciò si verifica a causa del modo in cui le voci principali utilizzano la gerarchia oggetti.

InObject() è spesso utilizzato in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
if (page 557)	Le funzioni **if** e **ObjectId** possono essere utilizzate insieme per creare espressioni condizionali. Ad esempio, le visualizzazioni possono ottenere una colorazione condizionale attraverso espressioni che utilizzano queste funzioni.
ObjectId - funzione per grafici (page 1471)	In modo simile a **if**, **ObjectId** è utilizzato anche con **InObject** per creare espressioni condizionali.

Esempio 1 - Funzionalità di base

Espressione del grafico e risultati

Il seguente esempio di base mostra come determinare se un oggetto è contenuto in un altro oggetto. In questo caso, controlleremo se un oggetto **Testo e immagine** risiede in un oggetto foglio, utilizzando l'ID del foglio come argomento.

Procedere come indicato di seguito:

1. Aprire un nuovo foglio e trascinare un grafico **Testo e immagine** sul foglio.
2. Nel pannello proprietà, fare clic su **Aggiungi misura**.
3. Fare clic su fx per visualizzare l'editor delle espressioni.
4. Incollare la seguente espressione nella finestra di dialogo:
   ```
   =InObject()
   ```
5. Modificare l'espressione per includere l'ID del foglio come stringa tra le parentesi.

Ad esempio, per un foglio con ID 1234-5678, si utilizza quanto segue:

6. `=InObject('1234-5678')`
7. Fare clic su **Applica**.

Il valore `-1` viene visualizzato nel grafico, a indicare che l'espressione è stata valutata come vera.

Esempio 2 - Oggetti con colori condizionati

Espressione del grafico e risultati

Panoramica

L'esempio seguente mostra come creare pulsanti di navigazione personalizzati con colorazioni diverse per indicare il foglio attualmente aperto.

Si inizia creando una nuova app e aprendo l'editor caricamento dati. Incollare il seguente script di caricamento in una nuova scheda. Si noti che i dati stessi sono un segnaposto e non saranno utilizzati nel contenuto dell'esempio.

Script di caricamento

```
Transactions:

Load
*
Inline
[
id,date,amount
8188,'1/19/2022',37.23
8189,'1/7/2022',17.17
8190,'2/28/2022',88.27
8191,'2/5/2022',57.42
8192,'3/16/2022',53.80
8193,'4/1/2022',82.06
8194,'4/7/2022',40.39
8195,'5/16/2022',87.21
8196,'6/15/2022',95.93
8197,'7/26/2022',45.89
8198,'8/9/2022',36.23
8199,'9/22/2022',25.66
8200,'11/23/2022',82.77
8201,'12/27/2022',69.98
8202,'1/1/2023',76.11
8203,'2/8/2022',25.12
8204,'3/19/2022',46.23
8205,'6/26/2022',84.21
8206,'9/14/2022',96.24
8207,'11/29/2022',67.67
];
```

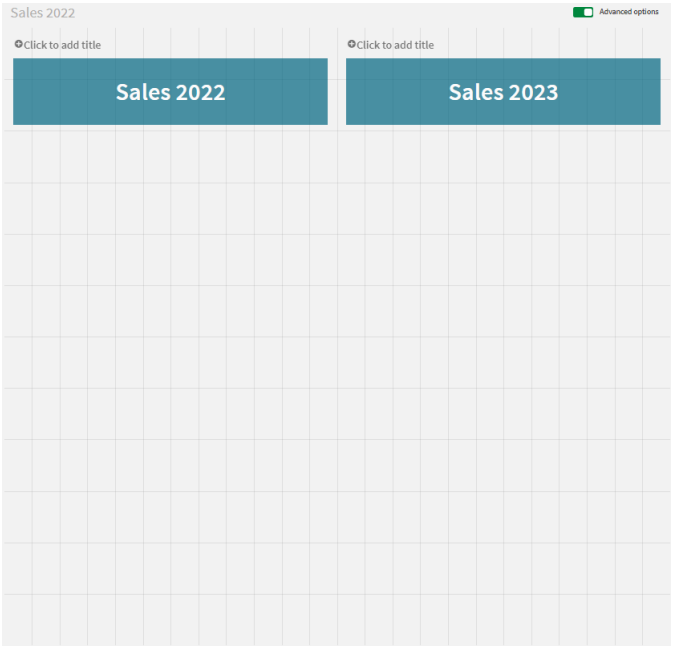
Creazione delle visualizzazioni

Caricare i dati e creare due nuovi fogli. Intitolarli rispettivamente *Vendite 2022* e *Vendite 2023*.

Successivamente, costruire due oggetti pulsante che verranno utilizzati per navigare tra i due fogli.

Procedere come indicato di seguito:

1. Aggiungere due oggetti **Pulsante** al foglio.
2. Sotto **Aspetto** > **Generale**, impostare l'**Etichetta** di ciascun pulsante rispettivamente su *Vendite 2022* e *Vendite 2023*.
3. Disporre i pulsanti in modo che corrispondano all'immagine seguente.
   Disposizione del foglio Vendite 2022 con due pulsanti di navigazione



4. Selezionare il pulsante *Vendite 2022* ed espandere **Azioni e navigazione** nel pannello delle proprietà.
5. Fare clic su **Aggiungi azione** e sotto **Navigazione**, selezionare **Vai in un foglio**.
6. Sotto **Foglio**, selezionare *Vendite 2022*.
7. Ripetere la configurazione dell'azione di questo pulsante per collegare il pulsante **Vendite 2023** al foglio *Vendite 2023*.
8. Per convertire i pulsanti in voci principali, fare clic con il pulsante destro del mouse e selezionare **Aggiungi a voci principali**.

Ora è possibile copiare ogni pulsante e incollarlo nel foglio *Vendite 2023*, utilizzando le stesse dimensioni e la stessa disposizione sul foglio.

Creazione di colori condizionali

Quindi, configurare i pulsanti in modo che siano blu se sono collegati al foglio attualmente aperto e grigio chiaro se sono collegati al foglio non aperto.

Procedere come indicato di seguito:

1. Aprire il foglio *Vendite 2022* e ottenere l'ID foglio dall'URL. Mantenere aperto il foglio *Vendite 2022*.
2. Fare clic sulla voce principale del pulsante **Vendite 2022** e selezionare **Modifica** nel pannello proprietà.
3. Sotto **Aspetto** > **Sfondo**, selezionare per colorare il pulsante **Per espressione**.
4. In **Espressione**, incollare il testo seguente:
   =if(InObject(''), Blue(), LightGray())
5. Tra le parentesi dell'espressione precedente, incollare l'ID del foglio *Vendite 2022*.

Il pulsante è ora configurato per diventare blu quando il foglio *Vendite 2022* è aperto e grigio chiaro quando non è aperto.

Ripetere le istruzioni di cui sopra per il foglio *Vendite 2023*, collegando la voce principale del pulsante **Vendite 2023** all'ID foglio *Vendite 2023*.

Ogni foglio dovrebbe ora avere due pulsanti che indicano il foglio attualmente aperto con il colore blu.

Foglio Vendite 2022 con colorazione blu per indicare che Vendite 2022 risulta attualmente visualizzato

IsPartialReload

Questa funzione restituisce- 1 (True) se l'operazione di ricaricamento attuale è parziale, altrimenti restituisce 0 (False).

Sintassi:

```
IsPartialReload()
```

ObjectId - funzione per grafici

La funzione del grafico **ObjectId()** restituisce l'ID dell'oggetto nel quale è valutata l'espressione. La funzione accetta un argomento opzionale che specifica il tipo di oggetto a cui si riferisce la funzione. L'oggetto può essere un foglio o una visualizzazione. La funzione è disponibile solo nelle espressioni del grafico.

Sintassi:

```
ObjectId([object_type_str])
```

Tipo di dati restituiti: stringa

L'unico argomento della funzione, **object_type_str**, è opzionale e si riferisce a un valore stringa che rappresenta il tipo dell'oggetto.

Argomenti

Argomento	Descrizione
object_type_str	Un valore stringa che rappresenta il tipo dell'oggetto da valutare.

Se nell'espressione della funzione non è specificato alcun argomento, **ObjectId()** restituisce l'ID dell'oggetto in cui viene utilizzata l'espressione. Per restituire l'ID dell'oggetto foglio all'interno del quale appare la visualizzazione, usare *ObjectId('sheet')*.

Nel caso di oggetti di visualizzazione nidificati all'interno di altri oggetti di visualizzazione, specificare il tipo di oggetto desiderato nell'argomento della funzione per ottenere risultati diversi. Ad esempio, per un grafico **Testo e immagine** con un contenitore, utilizzare *'text-image'* per restituire l'oggetto **Testo e immagine** e *'contenitore'* per restituire l'ID del contenitore.

Procedere come indicato di seguito:

1. In modalità analisi, aggiungere il seguente testo all'URL:
   /options/developer
2. Fare clic con il pulsante destro del mouse su una visualizzazione e fare clic su **Sviluppatore**.
3. Sotto **Proprietà**, ottenere l'ID oggetto dall'intestazione finestra di dialogo, oltre al tipo di oggetto dalla proprietà **"qType"**.

Limiti:

Questa funzione può dare risultati imprevisti se richiamata in un oggetto (ad esempio, un pulsante) all'interno di un contenitore che rappresenta una voce principale. Questa limitazione si applica anche alle voci principali della casella di filtro, che rappresentano contenitori di diverse caselle di elenco. Ciò si verifica a causa del modo in cui le voci principali utilizzano la gerarchia oggetti.

L'espressione grafico *ObjectId('sheet')* restituirà una stringa vuota in tali casi, mentre *ObjectId ('masterobject')* mostrerà l'identificatore della voce principale posseduta.

ObjectId() viene spesso utilizzato in combinazione con le seguenti funzioni:

Funzioni correlate

Funzione	Interazione
if (page 557)	Le funzioni **if** e **ObjectId** possono essere utilizzate insieme per creare espressioni condizionali. Ad esempio, le visualizzazioni possono ottenere una colorazione condizionale attraverso espressioni che utilizzano queste funzioni.
InObject - funzione per grafici (page 1466)	In modo simile a **if**, **InObject** viene utilizzato anche con **ObjectId** per creare espressioni condizionali.

Esempio 1 – Restituzione ID oggetto grafico

Espressione del grafico e risultati

Il seguente esempio di base dimostra come restituire l'ID di una visualizzazione.

Procedere come indicato di seguito:

1. Aprire un nuovo foglio e trascinare un grafico **Testo e immagine** sul foglio.
2. Nel pannello proprietà, fare clic su **Aggiungi misura**.
3. Fare clic su fx per visualizzare l'editor delle espressioni.
4. Incollare la seguente espressione nella finestra di dialogo:
   ```
   =ObjectId()
   ```
5. Fare clic su **Applica**.

L'ID dell'oggetto **Testo e immagine** viene mostrato nella visualizzazione.

Lo stesso risultato può essere ottenuto con la seguente espressione:

```
=ObjectId('text-image')
```

Esempio 2 - ID del foglio di ritorno

Espressione del grafico e risultati

Il seguente esempio di base dimostra come restituire l'ID del foglio in cui appare una visualizzazione.

Procedere come indicato di seguito:

1. Aprire un nuovo foglio e trascinare un grafico **Testo e immagine** sul foglio.
2. Nel pannello proprietà, fare clic su **Aggiungi misura**.
3. Fare clic su fx per visualizzare l'editor delle espressioni.
4. Incollare la seguente espressione nella finestra di dialogo:
   `=ObjectId('sheet')`
5. Fare clic su **Applica**.

L'ID del foglio viene visualizzato nella visualizzazione.

Esempio 3 - Espressione nidificata

Espressione del grafico e risultati

L'esempio seguente mostra come la funzione **ObjectId()** possa essere nidificata all'interno di altre espressioni.

Procedere come indicato di seguito:

1. Aprire un nuovo foglio e trascinare un grafico **Testo e immagine** sul foglio.
2. Nel pannello proprietà, fare clic su **Aggiungi misura**.
3. Fare clic su fx per visualizzare l'editor delle espressioni.
4. Incollare la seguente espressione nella finestra di dialogo:
   =if(InObject(ObjectId('text-image')), 'In Text & image', 'Not in Text & image')
5. Fare clic su **Applica**.

Il testo *In Testo e immagine* appare nel grafico, indicando che l'oggetto a cui si fa riferimento nell'espressione è un grafico **Testo e immagine**.

Per un esempio più dettagliato di utilizzo della colorazione condizionale, vedere l'esempio su *InObject - funzione per grafici (page 1466)*

ProductVersion

Questa funzione restituisce la versione e il numero di build completi di Qlik Sense sotto forma di stringa. Questa funzione è deprecata e sostituita da **EngineVersion()**.

Sintassi:

```
ProductVersion()
```

StateName - funzione per grafici

StateName() restituisce il nome dello stato alternato della visualizzazione in cui è utilizzata. StateName può essere utilizzata, ad esempio, per creare visualizzazioni con colori e testo dinamici che mostrano i cambiamenti di stato di una visualizzazione. Questa funzione può essere utilizzata nelle espressioni dei grafici, ma non può essere utilizzata per determinare lo stato a cui si riferisce l'espressione.

Sintassi:

```
StateName ()
```

Example 1:

Testo dinamico

```
='Region - ' & if(StateName() = '$', 'Default', StateName())
```

Example 2:

Colori dinamici

```
if(StateName() = 'Group 1', rgb(152, 171, 206),
      if(StateName() = 'Group 2', rgb(187, 200, 179),
            rgb(210, 210, 210)
      )
)
```

5.26 Funzioni di tabella

Le funzioni di tabella restituiscono informazioni relative alla tabella dati in fase di caricamento. Se non viene specificato alcun nome di tabella e la funzione viene utilizzata all'interno di un'istruzione **LOAD**, viene utilizzata la tabella attuale.

Nello script di caricamento dei dati è possibile utilizzare tutte le funzioni, mentre in un'espressione grafica è possibile utilizzare solo **NoOfRows**.

Panoramica sulle funzioni di tabella

Alcune funzioni vengono ulteriormente descritte dopo la panoramica. Per tali funzioni, è inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

FieldName

La funzione di script **FieldName** restituisce il nome del campo con il numero specificato all'interno di una tabella caricata in precedenza. Se la funzione viene utilizzata all'interno di un'istruzione **LOAD**, non deve fare riferimento alla tabella in corso di caricamento.

```
FieldName (field_number ,table_name)
```

FieldNumber

La funzione di script **FieldNumber** restituisce il numero di un campo specifico all'interno di una tabella caricata in precedenza. Se la funzione viene utilizzata all'interno di un'istruzione **LOAD**, non deve fare riferimento alla tabella in corso di caricamento.

```
FieldNumber (field_name ,table_name)
```

NoOfFields

La funzione di script **NoOfFields** restituisce il numero di campi all'interno di una tabella caricata in precedenza. Se la funzione viene utilizzata all'interno di un'istruzione **LOAD**, non deve fare riferimento alla tabella in corso di caricamento.

```
NoOfFields (table_name)
```

NoOfRows

La funzione **NoOfRows** restituisce il numero di righe (record) all'interno di una tabella caricata in precedenza. Se la funzione viene utilizzata all'interno di un'istruzione **LOAD**, non deve fare riferimento alla tabella in corso di caricamento.

```
NoOfRows (table_name)
```

NoOfTables

Questa funzione di script restituisce il numero di tabelle caricate in precedenza.

```
NoOfTables()
```

TableName

Questa funzione di script restituisce il nome della tabella con il numero specificato.

```
TableName(table_number)
```

TableNumber

Questa funzione di script restituisce il numero della tabella specificata. La tabella ripetizione ha il numero 0.

Se table_name non esiste, viene restituito NULL.

```
TableNumber(table_name)
```

Esempio:

In questo esempio si desidera creare una tabella con informazioni sulle tabelle e sui campi caricati.

Innanzitutto si caricheranno alcuni dati semplici. Questa operazione crea le due tabelle che verranno utilizzate per illustrare le funzioni di tabella descritte in questa sezione.

```
Characters:
Load Chr(RecNo()+Ord('A')-1) as Alpha, RecNo() as Num autogenerate 26;

ASCII:
Load
```

```
 if(RecNo()>=65 and RecNo()<=90,RecNo()-64) as Num,
 Chr(RecNo()) as AsciiAlpha,
 RecNo() as AsciiNum
autogenerate 255
 Where (RecNo()>=32 and RecNo()<=126) or RecNo()>=160 ;
```

In seguito, si ripete l'operazione sulle tabelle caricate utilizzando la funzione **NoOfTables**, quindi sui campi di ogni tabella utilizzando la funzione **NoOfFields** e si caricheranno le informazioni utilizzando le funzioni di tabella.

```
//Iterate through the loaded tables
For t = 0 to NoOfTables() - 1

//Iterate through the fields of table
 For f = 1 to NoOfFields(TableName($(t)))
  Tables:
  Load
   TableName($(t)) as Table,
   TableNumber(TableName($(t))) as TableNo,
   NoOfRows(TableName($(t))) as TableRows,
   FieldName($(f),TableName($(t))) as Field,
   FieldNumber(FieldName($(f),TableName($(t))),TableName($(t))) as FieldNo
   Autogenerate 1;
 Next f
Next t;
```

La tabella Tables risultante avrà l'aspetto seguente:

Load table

Table	TableNo	TableRows	Field	FieldNo
Characters	0	26	Alpha	1
Characters	0	26	Num	2
ASCII	1	191	Num	1
ASCII	1	191	AsciiAlpha	2
ASCII	1	191	AsciiNum	3

FieldName

La funzione di script **FieldName** restituisce il nome del campo con il numero specificato all'interno di una tabella caricata in precedenza. Se la funzione viene utilizzata all'interno di un'istruzione **LOAD**, non deve fare riferimento alla tabella in corso di caricamento.

Sintassi:

```
FieldName(field_number ,table_name)
```

Argomenti:

Argomenti

Argomento	Descrizione
field_number	Il numero del campo a cui si desidera fare riferimento.
table_name	La tabella contenente il campo a cui si desidera fare riferimento.

Esempio:

```
LET a = FieldName(4,'tab1');
```

FieldNumber

La funzione di script **FieldNumber** restituisce il numero di un campo specifico all'interno di una tabella caricata in precedenza. Se la funzione viene utilizzata all'interno di un'istruzione **LOAD**, non deve fare riferimento alla tabella in corso di caricamento.

Sintassi:

```
FieldNumber(field_name ,table_name)
```

Argomenti:

Argomenti

Argomento	Descrizione
field_name	Il nome del campo.
table_name	Il nome della tabella contenente il campo.

Se il campo field_name non esiste in table_name o table_name non esiste, la funzione restituisce 0.

Esempio:

```
LET a = FieldNumber('Customer','tab1');
```

NoOfFields

La funzione di script **NoOfFields** restituisce il numero di campi all'interno di una tabella caricata in precedenza. Se la funzione viene utilizzata all'interno di un'istruzione **LOAD**, non deve fare riferimento alla tabella in corso di caricamento.

Sintassi:

```
NoOfFields(table_name)
```

Argomenti:

Argomenti

Argomento	Descrizione
table_name	Il nome della tabella.

Esempio:

```
LET a = NoOfFields('tab1');
```

NoOfRows

La funzione **NoOfRows** restituisce il numero di righe (record) all'interno di una tabella caricata in precedenza. Se la funzione viene utilizzata all'interno di un'istruzione **LOAD**, non deve fare riferimento alla tabella in corso di caricamento.

Sintassi:

```
NoOfRows(table_name)
```

Argomenti:

Argomenti

Argomento	Descrizione
table_name	Il nome della tabella.

Esempio:

```
LET a = NoOfRows('tab1');
```

5.27 Funzioni trigonometriche e iperboliche

In questa sezione vengono descritte le funzioni per l'esecuzione delle operazioni trigonometriche e iperboliche. In tutte le funzioni gli argomenti sono espressioni che restituiscono angoli misurati in radianti, in cui **x** deve essere interpretato come un numero reale.

Tutti gli angoli sono misurati in radianti.

Tutte le funzioni possono essere utilizzate sia nello script di caricamento dei dati che nelle espressioni grafiche.

cos

Coseno di **x**. Il risultato è un numero compreso tra -1 e 1.

```
cos( x )
```

acos

Coseno inverso di **x**. La funzione è definita solo se -1≤**x**≤1. Il risultato è un numero compreso tra 0 e π.

```
acos( x )
```

sin

Seno di **x**. Il risultato è un numero compreso tra -1 e 1.

```
sin( x )
```

asin

Seno inverso di **x**. La funzione è definita solo se -1≤**x**≤1. Il risultato è un numero compreso tra - π/2 e π/2.

```
asin( x )
```

tan

Tangente di **x**. Il risultato è un numero reale.

```
tan( x )
```

atan

Tangente inversa di **x**. Il risultato è un numero compreso tra - π/2 e π/2.

```
atan( x )
```

atan2

Generalizzazione bidimensionale della funzione della tangente inversa. Restituisce l'angolo compreso tra l'origine e il punto rappresentato dalle coordinate **x** e **y**. Il risultato è un numero compreso tra - π e + π.

```
atan2( y,x )
```

cosh

Coseno iperbolico di **x**. Il risultato è un numero reale positivo.

```
cosh( x )
```

sinh

Seno iperbolico di **x**. Il risultato è un numero reale.

```
sinh( x )
```

tanh

Tangente iperbolica di **x**. Il risultato è un numero reale.

```
tanh( x )
```

acosh

Coseno iperbolico inverso di **x**. Il risultato è un numero reale positivo.

```
acosh( x )
```

asinh

Seno iperbolico inverso di **x**. Il risultato è un numero reale.

```
asinh( x )
```

atanh

Tangente iperbolica inversa di **x**. Il risultato è un numero reale.

```
atanh( x )
```

Esempi:

Il codice di script seguente carica un tabella campione, quindi carica una tabella contenente le operazioni trigonometriche e iperboliche calcolate sui valori.

```
SampleData:
LOAD * Inline
[Value
-1
0
1];

Results:
Load *,
cos(Value),
acos(Value),
sin(Value),
asin(Value),
tan(Value),
atan(Value),
atan2(Value, Value),
cosh(Value),
sinh(Value),
tanh(Value)
RESIDENT SampleData;

Drop Table SampleData;
```

6 Restrizione dell'accesso al file system

Per motivi di sicurezza, in modalità standard Qlik Sense non supporta i percorsi nello script di caricamento dei dati o funzioni e variabili che espongono il file system.

Tuttavia, poiché il file system risulta supportato in QlikView, è possibile disabilitare la modalità standard e utilizzare la modalità legacy per poter riutilizzare gli script di caricamento di QlikView.

La disabilitazione della modalità standard può determinare rischi per la sicurezza, in quanto viene esposto il file system.

Disabilitazione della modalità standard (page 1489)

6.1 Aspetti relativi alla sicurezza quando si effettua la connessione alle connessioni dati ODBC e OLE DB basate su file

Le connessioni dati ODBC e OLE DB che utilizzano driver basati su file visualizzeranno il percorso del file di dati connesso nella stringa di connessione. Il percorso può essere visualizzato quando si modifica la connessione, nella finestra di dialogo di selezione dei dati o in alcune query SQL. Questo può avvenire sia nella modalità standard che nella modalità legacy.

Se l'esposizione del percorso al file dati è un problema, si consiglia di connettersi al file dati mediante una connessione dati cartella, se possibile.

6.2 Limitazioni nella modalità standard

Nella modalità standard molte istruzioni, variabili e funzioni non possono essere utilizzate oppure sono sottoposte a limitazioni. L'utilizzo di istruzioni non supportate nello script di caricamento dei dati genera errori quando lo script viene eseguito. I messaggi di errore vengono memorizzati nel file di registro dello script. L'utilizzo di variabili e funzioni non supportate non genera messaggi di errore o l'inserimento di voci nel file di registro, tuttavia, la funzione restituisce NULL.

Durante la modifica dello script di caricamento dei dati, non esiste alcuna indicazione del fatto che una variabile, un'istruzione o una funzione non è supportata.

Variabili di sistema

Variabili di sistema

Variabile	Modalità standard	Modalità legacy	Definizione
Floppy	Non supportato	Supportato	Restituisce la lettera relativa alla prima unità floppy rilevata, in genere *a:*.
CD	Non supportato	Supportato	Restituisce la lettera relativa alla prima unità CD-ROM rilevata. Se non viene rilevata alcuna unità CD-ROM, viene restituito *c:*.
QvPath	Non supportato	Supportato	Restituisce la stringa costituita dal percorso del file eseguibile di Qlik Sense.
QvRoot	Non supportato	Supportato	Restituisce la directory principale del file eseguibile di Qlik Sense.
QvWorkPath	Non supportato	Supportato	Restituisce la stringa costituita dal percorso dell'app Qlik Sense attuale.
QvWorkRoot	Non supportato	Supportato	Restituisce la directory principale dell'app Qlik Sense attuale.
WinPath	Non supportato	Supportato	Restituisce la stringa costituita dal percorso di Windows.
WinRoot	Non supportato	Supportato	Restituisce la directory principale di Windows.

Variabile	Modalità standard	Modalità legacy	Definizione
$(include=...)	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	La variabile **Include/Must_Include** specifica un file contenente del testo che deve essere inserito nello script e valutato come codice di script. Non è utilizzato per aggiungere dati. È possibile memorizzare parti del codice di script in un file di testo separato e riutilizzarlo in diverse app. Questa è una variabile definita dall'utente.

Istruzioni di script regolari

Istruzioni di script regolari

Istruzione	Modalità standard	Modalità legacy	Definizione
Binary	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	L'istruzione **binary** viene utilizzata per caricare i dati da un'altra app.
Connect	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	L'istruzione **CONNECT** consente di definire l'accesso di Qlik Sense a un database generico mediante l'interfaccia OLE DB/ODBC. Per ODBC, occorre innanzitutto specificare la sorgente dati utilizzando l'amministratore ODBC.

Istruzione	Modalità standard	Modalità legacy	Definizione
Directory	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	L'istruzione **Directory** definisce in quale directory ricercare i file dei dati nelle istruzioni **LOAD** successive finché non viene eseguita una nuova istruzione **Directory**.
Execute	Non supportato	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	L'istruzione **Execute** viene utilizzata per eseguire altri programmi, mentre Qlik Sense sta caricando i dati. Ad esempio, per effettuare le connessioni necessarie.
LOAD from ...	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	L'istruzione **LOAD** carica i campi da un file, dai dati definiti nello script, da una tabella caricata in precedenza, da una pagina Web, dal risultato di un'istruzione **SELECT** seguente o dalla generazione automatica di dati.
Store into ...	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	L'istruzione **Store** crea un file QVD, o text.

Istruzioni di controllo dello script

Istruzioni di controllo dello script

Istruzione	Modalità standard	Modalità legacy	Definizione
For each... filelist mask/dirlist mask	Input supportato: Percorso che utilizza la connessione alla libreria Output restituito: connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system Output restituito: Connessione alla libreria o percorso al file system, in base all'input	La sintassi filelist mask restituisce un elenco con valori separati da virgole di tutti i file presenti nella directory attuale che presentano una corrispondenza con **filelist mask**. La sintassi **dirlist mask** restituisce un elenco con valori separati da virgole di tutte le directory incluse nella directory attuale che presentano una corrispondenza con la maschera del nome della directory.

Funzioni di file

Funzioni di file

Funzione	Modalità standard	Modalità legacy	Definizione
Attribute()	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	Restituisce il valore dei metatag di file multimediali differenti come testo.
ConnectString()	Output restituito: nome di connessione della libreria	Il nome di connessione della libreria o la connessione effettiva, a seconda dell'input	Restituisce la stringa di connessione attiva per le connessioni ODBC o OLE DB.
FileDir()	Output restituito: connessione alla libreria	Output restituito: Connessione alla libreria o percorso al file system, in base all'input	La funzione **FileDir** restituisce una stringa contenente il percorso della directory del file tabella in corso di lettura.

Funzione	Modalità standard	Modalità legacy	Definizione
FilePath()	Output restituito: connessione alla libreria	Output restituito: Connessione alla libreria o percorso al file system, in base all'input	La funzione **FilePath** restituisce una stringa contenente il percorso completo del file tabella in corso di lettura.
FileSize()	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	La funzione **FileSize** restituisce un valore intero contenente le dimensioni in byte del file filename oppure, se non viene specificato alcun filename, del file tabella in corso di lettura.
FileTime()	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	La funzione **FileTime** restituisce un indicatore temporale in UTC per la data e l'ora dell'ultima modifica del file filename. Se non viene specificato alcun filename, la funzione farà riferimento al file tabella in corso di lettura.
GetFolderPath()	Non supportato	Output restituito: percorso assoluto	La funzione **GetFolderPath** restituisce il valore della funzione Microsoft Windows *SHGetFolderPath*. Questa funzione utilizza come input il nome di una cartella Microsoft Windows e restituisce il percorso completo della cartella.

Funzione	Modalità standard	Modalità legacy	Definizione
QvdCreateTime()	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	Questa funzione di script restituisce l'intestazione XML relativa alla data e ora da un file QVD, se disponibile, altrimenti restituisce NULL. Nella data e ora, l'ora è fornita in UTC.
QvdFieldName()	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	La funzione script restituisce il nome del numero campo **fieldno** in un file QVD. Se il campo non esiste, viene restituito NULL.
QvdNoOfFields()	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	Questa funzione dello script restituisce il numero di campi all'interno di file QVD.
QvdNoOfRecords()	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	Questa funzione dello script restituisce il numero di record attualmente presente in un file QVD.
QvdTableName()	Input supportato: Percorso che utilizza la connessione alla libreria	Input supportato: Percorso che utilizza la connessione alla libreria o al file system	Questa funzione di script restituisce il nome della tabella memorizzata in un file QVD.

Funzioni di sistema

Funzioni di sistema

Funzione	Modalità standard	Modalità legacy	Definizione
DocumentPath()	Non supportato	Output restituito: percorso assoluto	Questa funzione restituisce una stringa contenente il percorso completo dell'app Qlik Sense attuale.

Funzione	Modalità standard	Modalità legacy	Definizione
GetRegistryString()	Non supportato	Supportato	Restituisce il valore di una chiave di registro denominata con un dato percorso di registro. Questa funzione può essere utilizzata indifferentemente nel grafico e nello script.

6.3 Disabilitazione della modalità standard

È possibile disabilitare la modalità standard o, in altre parole, impostare la modalità legacy, per poter riutilizzare gli script di caricamento di QlikView che fanno riferimento a percorsi di file assoluti o relativi, così come alle connessioni della libreria.

La disabilitazione della modalità standard può determinare rischi per la sicurezza, in quanto viene esposto il file system.

Qlik Sense

Per Qlik Sense, la modalità può essere disabilitata QMC mediante la proprietà **Modalità standard**.

Qlik Sense Desktop

In Qlik Sense Desktop è possibile impostare la modalità standard/legacy nel file *Settings.ini*.

Se si è installato Qlik Sense Desktop utilizzando il percorso di installazione predefinito, *Settings.ini* si trova nel percorso *C:\Users\{user}\Documents\Qlik\Sense\Settings.ini*. Se si è installato Qlik Sense Desktop in una cartella selezionata dall'utente, *Settings.ini* si trova nella cartella *Engine* del percorso di installazione.

Procedere come indicato di seguito:

1. Aprire il file *Settings.ini* in un editor di testo.
2. Modificare *StandardReload=1* in *StandardReload=0*.
3. Salvare il file e avviare Qlik Sense Desktop.

Qlik Sense Desktop sarà ora eseguito in modalità legacy.

Impostazioni

Le impostazioni disponibili per StandardReload sono:

- 1 (modalità standard)
- 0 (modalità legacy)

6 Scripting a livello di grafico

Durante la modifica dei dati sul grafico, l'utente utilizza un sottoinsieme di script Qlik Sense, costituito da una serie di istruzioni. Un'istruzione può essere un'istruzione di script regolare o un'istruzione di controllo dello script. Alcune istruzioni possono essere precedute da prefissi.

Le istruzioni regolari vengono generalmente utilizzate per la manipolazione dei dati. Queste istruzioni possono essere scritte su un qualsiasi numero di righe nello script e devono sempre terminare con un punto e virgola, ";".

In genere, le istruzioni di controllo vengono utilizzate per controllare il flusso di esecuzione dello script. Ogni clausola di un'istruzione di controllo deve essere mantenuta in una singola riga dello script e può terminare con un punto e virgola oppure con un fine riga.

I prefissi possono essere applicati alle istruzioni regolari pertinenti, ma mai a istruzioni di controllo.

Tutte le parole chiave dello script possono essere immesse con qualsiasi combinazione di caratteri maiuscoli e minuscoli. I nomi dei campi e delle variabili utilizzati nelle istruzioni possono essere immessi indipendentemente dal formato del carattere.

In questa sezione, è possibile trovare un elenco in ordine alfabetico di tutte le istruzioni script, istruzioni di controllo e prefissi disponibili nel sottoinsieme dello script utilizzato durante la modifica dei dati grafici.

6.4 Istruzione di controllo

Durante la modifica dei dati sul grafico, l'utente utilizza un sottoinsieme di script Qlik Sense, costituito da una serie di istruzioni. Un'istruzione può essere un'istruzione di script regolare o un'istruzione di controllo dello script.

In genere, le istruzioni di controllo vengono utilizzate per controllare il flusso di esecuzione dello script. Ogni clausola di un'istruzione di controllo deve essere inserita in una singola riga nello script e può terminare con un punto e virgola o con un carattere di fine riga.

I prefissi non vengono mai applicati alle istruzioni di controllo.

Tutte le parole chiave dello script possono essere immesse con qualsiasi combinazione di caratteri maiuscoli e minuscoli.

Panoramica istruzioni di controllo modificatore grafico

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Call

L'istruzione di controllo **call** consente di chiamare una subroutine che deve essere definita da un'istruzione **sub** precedente.

```
Call name ( [ paramlist ])
```

Do..loop

L'istruzione di controllo **do..loop** è un costrutto per la ripetizione di script che esegue una o più istruzioni finché non incontra una condizione logica.

```
Do..loop [ ( while | until ) condition ] [statements]
[exit do [ ( when | unless ) condition ] [statements]
loop [ ( while | until ) condition ]
```

End

La parola chiave dello script **End** viene utilizzata per chiudere le clausole **If**, **Sub** e **Switch**.

Exit

La parola chiave dello script **Exit** fa parte dell'istruzione **Exit Script**, ma può essere utilizzata anche per uscire dalle clausole **Do**, **For** o **Sub**.

Exit script

Questa istruzione di controllo interrompe l'esecuzione dello script. Può essere inserita in un punto qualsiasi dello script.

```
Exit script[ (when | unless) condition ]
```

For..next

L'istruzione di controllo **for..next** è un costrutto per la ripetizione di script con un contatore. Le istruzioni all'interno del ciclo incluso tra **for** e **next** verranno eseguite per ogni valore del contatore in base ai limiti inferiore e superiore specificati.

```
For..next  counter = expr1 to expr2 [ stepexpr3 ]
[statements]
[exit for [ ( when | unless ) condition ]
[statements]
Next [counter]
```

For each ..next

L'istruzione di controllo **for each..next** è un costrutto per la ripetizione di script che esegue una o più istruzioni per ogni valore in un elenco le cui voci sono separate da virgole. Le istruzioni incluse nel ciclo fra **for** e **next** verranno eseguite per ogni valore nell'elenco.

```
For each..next var in list
[statements]
[exit for [ ( when | unless ) condition ]
[statements]
next [var]
```

If..then

L'istruzione di controllo **if..then** è un costrutto per la selezione di script che forza l'esecuzione dello script su percorsi diversi in base a una o più condizioni logiche.

*Poiché **if..then** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, ciascuna delle quattro possibili clausole corrispondenti (**if..then**, **elseif..then**, **else** e **end if**) deve essere contenuta in una sola riga.*

```
If..then..elseif..else..end if condition then
  [ statements ]
{ elseif condition then
  [ statements ] }
[ else
  [ statements ] ]
end if
```

Next

La parola chiave dello script **Next** consente di chiudere i loop **For**.

Sub

L'istruzione di controllo **sub..end sub** definisce una subroutine che può essere richiamata da un'istruzione **call**.

```
Sub..end sub name [ ( paramlist )] statements end sub
```

Switch

L'istruzione di controllo **switch** è un costrutto per la selezione di script che forza l'esecuzione dello script su percorsi diversi, in base al valore di un'espressione.

```
Switch..case..default..end switch expression {case valuelist [ statements ]}
[default statements] end switch
```

To

La parola chiave dello script **To** viene utilizzata in diverse istruzioni dello script.

Call

L'istruzione di controllo **call** consente di chiamare una subroutine che deve essere definita da un'istruzione **sub** precedente.

Sintassi:

```
Call name ( [ paramlist ])
```

Argomenti:

Argomenti

Argomento	Descrizione
name	Il nome della subroutine.

Argomento	Descrizione
paramlist	Un elenco separato da virgole di parametri effettivi da inviare alla subroutine. Ogni voce dell'elenco può essere un nome di campo, una variabile o un'espressione arbitraria.

La subroutine chiamata da un'istruzione **call** deve essere definita da un'istruzione **sub** rilevata precedentemente durante l'esecuzione dello script.

I parametri vengono copiati nella subroutine e, se il parametro nell'istruzione **call** è una variabile e non un'espressione, verranno copiati nuovamente all'uscita dalla subroutine.

Limiti:

- Poiché **call** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, non deve superare un limite di riga.
- Quando si definisce una routine secondaria con `Sub..end sub`all'interno di un'istruzione di controllo, ad esempio `if..then`, è possibile richiamare la routine secondaria solo dall'interno della stessa istruzione di controllo.

Do..loop

L'istruzione di controllo **do..loop** è un costrutto per la ripetizione di script che esegue una o più istruzioni finché non incontra una condizione logica.

Sintassi:

```
Do [ ( while | until ) condition ] [statements]
[exit do [ ( when | unless ) condition ] [statements]
loop[ ( while | until ) condition ]
```

> *Poiché **do..loop** è un'istruzione di controllo che termina con un punto e virgola o con un carattere di fine riga, ciascuna delle tre possibili clausole corrispondenti (**do**, **exit do** e **loop**) deve essere contenuta in una sola riga.*

Argomenti:

Argomenti

Argomento	Descrizione
condition	Un'espressione logica che restituisce un valore True o False.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.
while / until	La clausola condizionale **while** o **until** deve comparire una sola volta in ciascuna istruzione **do..loop**, dopo **do** o dopo **loop**. Ogni espressione condition verrà interpretata solo al primo rilevamento, ma verrà valutata ogni volta che sarà rilevata nel ciclo.

Argomento	Descrizione
exit do	Se all'interno del ciclo è presente una clausola **exit do**, l'esecuzione dello script verrà trasferita alla prima istruzione dopo la clausola **loop** indicando quindi la fine del ciclo. Una clausola **exit do** può essere resa condizionale dall'utilizzo opzionale di un suffisso **when** o **unless**.

End

La parola chiave dello script **End** viene utilizzata per chiudere le clausole **If**, **Sub** e **Switch**.

Exit

La parola chiave dello script **Exit** fa parte dell'istruzione **Exit Script**, ma può essere utilizzata anche per uscire dalle clausole **Do**, **For** o **Sub**.

Exit script

Questa istruzione di controllo interrompe l'esecuzione dello script. Può essere inserita in un punto qualsiasi dello script.

Sintassi:

```
Exit Script [ (when | unless) condition ]
```

Poiché **exit script** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, non deve superare un limite di riga.

Argomenti:

Argomenti

Argomento	Descrizione
condition	Un'espressione logica che restituisce un valore True o False.
when / unless	Un'istruzione **exit script** può essere resa condizionale dall'utilizzo opzionale della clausola **when** o **unless**.

Esempi:

```
//Exit script
Exit Script;

//Exit script when a condition is fulfilled
Exit Script when a=1
```

For..next

L'istruzione di controllo **for..next** è un costrutto per la ripetizione di script con un contatore. Le istruzioni all'interno del ciclo incluso tra **for** e **next** verranno eseguite per ogni valore del contatore in base ai limiti inferiore e superiore specificati.

Sintassi:

```
For counter = expr1 to expr2 [ step expr3 ]
[statements]
[exit for [ ( when | unless ) condition ]
[statements]
Next [counter]
```

Le espressioni *expr1*, *expr2* ed *expr3* vengono valutate solo la prima volta che il ciclo viene eseguito. Il valore della variabile counter può essere modificato dalle istruzioni all'interno del ciclo, tuttavia l'utilizzo di questa procedura di programmazione non è consigliato.

Se all'interno del ciclo è presente una clausola **exit for**, l'esecuzione dello script verrà trasferita alla prima istruzione dopo la clausola **next** indicando quindi la fine del ciclo. Una clausola **exit for** può essere resa condizionale dall'utilizzo opzionale di un suffisso **when** o **unless**.

*Poiché **for..next** è un'istruzione di controllo che termina con un punto e virgola o con un carattere di fine riga, ciascuna delle tre possibili clausole corrispondenti (**for..to..step**, **exit for** e **next**) deve essere contenuta in una sola riga.*

Argomenti:

Argomenti

Argomento	Descrizione
counter	Un nome di variabile. Se *counter* viene specificato dopo **next**, deve avere lo stesso nome di variabile rilevato dopo l'istruzione **for** corrispondente.
expr1	Un'espressione che determina il primo valore della variabile*counter* per cui deve essere eseguito il ciclo.
expr2	Un'espressione che determina l'ultimo valore della variabile *counter* per cui deve essere eseguito il ciclo.
expr3	Un'espressione che determina il valore che indica l'incremento della variabile *counter* ogni volta che il ciclo è stato eseguito.
condition	Un'espressione logica che restituisce un valore True o False.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.

For each..next

L'istruzione di controllo **for each..next** è un costrutto per la ripetizione di script che esegue una o più istruzioni per ogni valore in un elenco le cui voci sono separate da virgole. Le istruzioni incluse nel ciclo fra **for** e **next** verranno eseguite per ogni valore nell'elenco.

Sintassi:
Una sintassi speciale consente di generare elenchi contenenti nomi di file e di directory nella directory attuale.

```
for each var in list
[statements]
[exit for [ ( when | unless ) condition ]
[statements]
next [var]
```

Argomenti:

Argomenti

Argomento	Descrizione
var	Il nome di una variabile di script che acquisisce un nuovo valore dall'elenco a ogni esecuzione del ciclo. Se **var** viene specificato dopo **next**, deve avere lo stesso nome di variabile rilevato dopo l'istruzione **for each** corrispondente.

Il valore della variabile **var** può essere modificato dalle istruzioni all'interno del ciclo, tuttavia l'utilizzo di questa procedura di programmazione non è consigliato.

Se all'interno del ciclo è presente una clausola **exit for**, l'esecuzione dello script verrà trasferita alla prima istruzione dopo la clausola **next** indicando quindi la fine del ciclo. Una clausola **exit for** può essere resa condizionale dall'utilizzo opzionale di un suffisso **when** o **unless**.

*Poiché **for each..next** è un'istruzione di controllo che termina con un punto e virgola o con un carattere di fine riga, ciascuna delle tre possibili clausole corrispondenti (**for each**, **exit for** e **next**) deve essere contenuta in una sola riga.*

Sintassi:

```
list := item { , item }
item := constant | (expression) | filelist mask | dirlist mask |
fieldvaluelist mask
```

Argomenti

Argomento	Descrizione
constant	Qualsiasi numero o stringa. Tenere presente che una stringa inserita direttamente nello script deve essere racchiusa tra virgolette singole. Se la stringa non viene racchiusa tra virgolette singole, verrà interpretata come una variabile, pertanto verrà utilizzato il valore della variabile. Non è necessario che i numeri siano racchiusi tra virgolette singole.
expression	Un'espressione arbitraria.
mask	Una maschera di un nome di file o di cartella che può includere un carattere qualsiasi di nome di file valido, così come i caratteri speciali standard, quali ***** e **?**. È possibile utilizzare percorsi di file assoluti o percorsi lib://.
condition	Un'espressione logica che restituisce un valore True o False.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.
filelist mask	Questa sintassi restituisce un elenco con valori separati da virgole di tutti i file presenti nella directory attuale che presentano una corrispondenza con la maschera del nome di file. *Questo argomento supporta esclusivamente le connessioni alla libreria in modalità standard.*
dirlist mask	Questa sintassi restituisce un elenco con valori separati da virgole di tutte le cartelle incluse nella cartella attuale che presentano una corrispondenza con la maschera del nome di file. *Questo argomento supporta esclusivamente le connessioni alla libreria in modalità standard.*
fieldvaluelist mask	Questa sintassi ripete i valori di un campo già caricato in Qlik Sense.

*Qlik Connettori provider di archiviazione Web e altre connessioni DataFiles non supportano le maschere di filtro che utilizzano caratteri speciali (***** e **?**).*

Example 1: Caricamento di un elenco di file

```
// LOAD the files 1.csv, 3.csv, 7.csv and xyz.csv
for each a in 1,3,7,'xyz'
   LOAD * from file$(a).csv;
next
```

Example 2: Creazione di un elenco di file sul disco

In questo esempio viene caricato un elenco di tutti i file correlati a Qlik Sense in una cartella.

```
sub DoDir (Root)
   for each Ext in 'qvw', 'qva', 'qvo', 'qvs', 'qvc', 'qvf', 'qvd'

      for each File in filelist (Root&'/*.' &Ext)

         LOAD
            '$(File)' as Name,
            FileSize( '$(File)' ) as Size,
            FileTime( '$(File)' ) as FileTime
         autogenerate 1;

      next File

   next Ext
   for each Dir in dirlist (Root&'/*' )

      call DoDir (Dir)

   next Dir

end sub

call DoDir ('lib://DataFiles')
```

Example 3: Ripetizione dei valori di un campo

In questo esempio viene ripetuto l'elenco di valori caricati di FIELD e viene generato un nuovo campo NEWFIELD. Per ciascun valore di FIELD, verranno creati due record NEWFIELD.

```
load * inline [
FIELD
one
two
three
];

FOR Each a in FieldValueList('FIELD')
LOAD '$(a)' &'-'&RecNo() as NEWFIELD AutoGenerate 2;
NEXT a
```

La tabella risultante avrà l'aspetto seguente:

Example table

NEWFIELD
one-1
one-2

NEWFIELD
two-1
two-2
three-1
three-2

If..then..elseif..else..end if

L'istruzione di controllo **if..then** è un costrutto per la selezione di script che forza l'esecuzione dello script su percorsi diversi in base a una o più condizioni logiche.

In genere, le istruzioni di controllo vengono utilizzate per controllare il flusso di esecuzione dello script. In un'espressione del grafico, utilizzare invece la funzione condizionale **if**.

Sintassi:

```
If  condition then
  [ statements ]
{ elseif condition then
  [ statements ] }
[ else
  [ statements ] ]
end if
```

Poiché **if..then** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, ciascuna delle quattro possibili clausole corrispondenti (**if..then**, **elseif..then**, **else** e **end if**) deve essere contenuta in una sola riga.

Argomenti:

Argomenti

Argomento	Descrizione
condition	Un'espressione logica che può restituire un valore True o False.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.

Example 1:

```
if a=1 then

      LOAD * from abc.csv;
      SQL SELECT e, f, g from tab1;

end if
```

Example 2:

```
if a=1 then; drop table xyz; end if;
```

Example 3:

```
if x>0 then
      LOAD * from pos.csv;
elseif x<0 then
      LOAD * from neg.csv;
else
      LOAD * from zero.txt;
end if
```

Next

La parola chiave dello script **Next** consente di chiudere i loop **For**.

Sub..end sub

L'istruzione di controllo **sub..end sub** definisce una subroutine che può essere richiamata da un'istruzione **call**.

Sintassi:

```
Sub name [ ( paramlist )] statements end sub
```

Gli argomenti vengono copiati nella subroutine e, se i relativi parametri reali nell'istruzione **call** corrispondono a un nome di variabile, vengono copiati nuovamente quando si chiude la subroutine.

Se una subroutine presenta più parametri formali di quelli effettivi passati da un'istruzione **call**, i parametri extra vengono inizializzati su NULL e possono essere utilizzati come variabili locali all'interno della subroutine.

Argomenti:

Argomenti

Argomento	Descrizione
name	Il nome della subroutine.
paramlist	Un elenco separato da virgole di nomi di variabili per i parametri formali della subroutine. Può essere utilizzato come qualsiasi variabile all'interno della subroutine.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.

Limiti:

- Poiché **sub** è un'istruzione di controllo che termina con un punto e virgola o con un carattere di fine riga, ciascuna delle due clausole corrispondenti (**sub** e **end sub**) deve essere contenuta in una sola riga.
- Quando si definisce una routine secondaria con `Sub..end sub`all'interno di un'istruzione di controllo, ad esempio `if..then`, è possibile richiamare la routine secondaria solo dall'interno della stessa istruzione di controllo.

Example 1:

```
Sub INCR (I,J)
I = I + 1
Exit Sub when I < 10
J = J + 1
End Sub
Call INCR (X,Y)
```

Example 2: - trasferimento parametri

```
Sub ParTrans (A,B,C)
A=A+1
B=B+1
C=C+1
End Sub
A=1
X=1
C=1
Call ParTrans (A, (X+1)*2)
```

Dall'esempio precedente risulta che localmente, all'interno della subroutine, A verrà inizializzato su 1, B verrà inizializzato su 4 e C verrà inizializzato su NULL.

Quando si chiude la subroutine, la variabile globale A otterrà 2 come valore (ricopiato dalla subroutine). Il secondo parametro reale “(X+1)*2” non verrà ricopiato dato che non si tratta di una variabile. Infine, la variabile globale C non verrà influenzata dalla chiamata della subroutine.

Switch..case..default..end switch

L'istruzione di controllo **switch** è un costrutto per la selezione di script che forza l'esecuzione dello script su percorsi diversi, in base al valore di un'espressione.

Sintassi:

```
Switch expression {case valuelist [ statements ]} [default statements] end switch
```

*Poiché **switch** è un'istruzione di controllo e come tale termina con un punto e virgola o con un carattere di fine riga, ciascuna delle quattro possibili clausole corrispondenti (**switch**, **case**, **default** e **end switch**) deve essere contenuta in una sola riga.*

Argomenti:

Argomenti

Argomento	Descrizione
expression	Un'espressione arbitraria.
valuelist	Un elenco separato da virgole dei valori con i quali viene confrontato il valore dell'espressione. L'esecuzione dello script continua con le istruzioni del primo gruppo in cui il valore di valuelist è pari al valore nell'espressione. Ciascun valore in valuelist può essere un'espressione arbitraria. Se non viene individuata alcuna corrispondenza in nessuna clausola **case**, vengono eseguite le eventuali istruzioni della clausola **default**.
statements	Qualsiasi gruppo di una o più istruzioni dello script di Qlik Sense.

Esempio:

```
Switch I
Case 1
LOAD '$(I): CASE 1' as case autogenerate 1;
Case 2
LOAD '$(I): CASE 2' as case autogenerate 1;
Default
LOAD '$(I): DEFAULT' as case autogenerate 1;
End Switch
```

To

La parola chiave dello script **To** viene utilizzata in diverse istruzioni dello script.

6.5 Prefissi

I prefissi possono essere applicati alle istruzioni regolari pertinenti, ma mai a istruzioni di controllo.

Tutte le parole chiave dello script possono essere immesse con qualsiasi combinazione di caratteri maiuscoli e minuscoli. I nomi dei campi e delle variabili utilizzati nelle istruzioni possono essere immessi indipendentemente dal formato del carattere.

Panoramica prefissi modificatore grafico

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

Add

Il prefisso **Add** può essere aggiunto a qualsiasi istruzione **LOAD** o **SELECT** nello script per specificare che dovrebbe aggiungere record a un'altra tabella. Specifica anche che questa istruzione dovrebbe essere eseguita in un ricaricamento parziale. Il prefisso **Add** può essere usato anche in un'istruzione **Map**.

```
Add [only] [Concatenate[(tablename )]] (loadstatement | selectstatement)
```

```
Add [ Only ] mapstatement
```

Replace

Il prefisso **Replace** può essere aggiunto a qualsiasi istruzione **LOAD** o **SELECT** nello script per specificare che la tabella caricata dovrebbe sostituire un'altra tabella. Specifica anche che questa istruzione dovrebbe essere eseguita in un ricaricamento parziale. Il prefisso **Replace** può essere usato anche in un'istruzione **Map**.

```
Replace [only] [Concatenate[(tablename) ]] (loadstatement | selectstatement)
Replace [only] mapstatement
```

Add

In un contesto di modifica del grafico, il prefisso **Add** è utilizzato con **LOAD** per aggiungere valori alla tabella *HC1*, che rappresenta l'ipercubo calcolato da Qlik associative engine. È possibile specificare uno o più colonne. I valori mancanti vengono automaticamente completati da Qlik associative engine.

Sintassi:

```
Add loadstatement
```

Esempio:

Questo esempio aggiunge due righe alle colonne *Date* e *Vendite* dall'istruzione inline.

```
Add Load
x as Dates,
y as Sales
Inline
[
Dates,Sales
2001/09/1,1000
2001/09/10,-300
]
```

Replace

In un contesto di modifica del grafico, il prefisso **Replace** modifica tutti i valori della tabella *HC1* con un valore calcolato definito dallo script.

Sintassi:

```
Replace loadstatement
```

Esempio:

Questo esempio sovrascrive tutti i valori nella colonna `z` con la somma di `x` e `y`.

```
Replace Load
x+y as z
Resident HC1;
```

6.6 Istruzioni regolari

Le istruzioni regolari vengono generalmente utilizzate per la manipolazione dei dati. Queste istruzioni possono essere scritte su un qualsiasi numero di righe nello script e devono sempre terminare con un punto e virgola, ";".

Tutte le parole chiave dello script possono essere immesse con qualsiasi combinazione di caratteri maiuscoli e minuscoli. I nomi dei campi e delle variabili utilizzati nelle istruzioni possono essere immessi indipendentemente dal formato del carattere.

Panoramica istruzioni regolari modificatore grafico

Ciascuna funzione viene descritta ulteriormente dopo la panoramica. È inoltre possibile fare clic sul nome della funzione nella sintassi per accedere immediatamente ai dettagli per tale funzione specifica.

LOAD

In un contesto di modifica del grafico, l'istruzione **LOAD** carica dati aggiuntivi sull'ipercubo, da dati definiti nello script o da una tabella precedentemente caricata. È anche possibile caricare dati da connessioni di analisi.

*L'istruzione **LOAD** deve includere il prefisso **Replace** o **Add**, altrimenti verrà rifiutata.*

```
Add | Replace Load [ distinct ] fieldlist
(
inline data [ format-spec ] |
resident table-label
) | extension pluginname.functionname([script] tabledescription)]
[ where criterion | while criterion ]
[ group by groupbyfieldlist ]
[order by orderbyfieldlist ]
```

Let

L'istruzione **let** è un complemento all'istruzione **set**, utilizzata per definire le variabili degli script. L'istruzione **let**, a differenza dell'istruzione **set**, valuta l'espressione posta sul lato destro del simbolo '=' al tempo di esecuzione dello script prima dell'assegnazione alla variabile.

```
Let variablename=expression
```

Set

L'istruzione **set** viene utilizzata per definire le variabili di script. Le variabili possono essere utilizzate per sostituire stringhe, percorsi, unità e così via.

```
Set variablename=string
```

Put

L'istruzione **Put** viene utilizzata per impostare alcuni valori numerici nell'ipercubo.

HCValue

L'istruzione **HCValue** è utilizzata per recuperare valori in una riga di una colonna specificata.

Load

In un contesto di modifica del grafico, l'istruzione **LOAD** carica dati aggiuntivi sull'ipercubo, da dati definiti nello script o da una tabella precedentemente caricata. È anche possibile caricare dati da connessioni di analisi.

*L'istruzione **LOAD** deve includere il prefisso **Replace** o **Add**, altrimenti verrà rifiutata.*

Sintassi:

```
Add | Replace LOAD fieldlist
(
inline data [ format-spec ] |
resident table-label
) | extension pluginname.functionname([script] tabledescription)]
[ where criterion | while criterion ]
[ group by groupbyfieldlist ]
[order by orderbyfieldlist ]
```

Argomenti:

Argomenti

Argomento	Descrizione
fieldlist	*fieldlist ::= (* | field {, * | field })* Un elenco di campi da caricare. L'utilizzo del carattere * come elenco dei campi indica tutti i campi della tabella. *field ::= (fieldref | expression) [***as** *aliasname]* La definizione di campo deve sempre contenere un valore letterale, un riferimento a un campo esistente o un'espressione. *fieldref ::= (fieldname |@fieldnumber |@startpos:endpos [**I** | **U** | **R** | **B** | **T**])* *fieldname* è un testo che è identico al nome di un campo nella tabella. Tenere presente che il nome di campo deve essere racchiuso da virgolette doppie o parentesi quadre se, ad esempio, contiene spazi. Talvolta i nomi dei campi non sono disponibili in modo esplicito. Verrà quindi utilizzata una notazione differente: @*fieldnumber* rappresenta il numero di campo di un file tabellare delimitato. Deve essere un intero positivo preceduto da "@". La numerazione viene sempre eseguita partendo da 1 fino al numero di campi presenti. @*startpos*:*endpos* rappresenta le posizioni di inizio e di fine di un campo in un file con record a lunghezza fissa. Le posizioni devono essere entrambe numeri interi positivi. I due valori numerici devono essere preceduti dal simbolo "@" e separati da due punti. La numerazione viene sempre eseguita partendo da 1 fino al numero di posizioni presenti. Nell'ultimo campo **n** viene utilizzato come posizione finale. • Se @*startpos*:*endpos* è seguito immediatamente dal carattere **I** o **U**, i byte letti verranno interpretati come un valore intero binario con segno (**I**) o senza segno (**U**), in base all'ordine dei byte Intel. Il numero di posizioni lette deve essere 1, 2 o 4. • Se @*startpos*:*endpos* è immediatamente seguito dal carattere **R**, la lettura dei byte verrà interpretata come un numero binario reale (a virgola mobile a 32 bit o a 64 bit conforme allo standard IEEE). Il numero di posizioni lette deve essere 4 o 8. • Se @*startpos*:*endpos* è immediatamente seguito dal carattere **B**, la lettura dei byte sarà interpretata come numeri BCD (Binary Coded Decimal), in base allo standard COMP-3. Può essere specificato un numero di byte qualsiasi. *expression* può essere una funzione numerica o una funzione stringa basata su uno o molti altri campi della stessa tabella. Per ulteriori informazioni, vedere la sintassi delle espressioni. **as** viene utilizzato per assegnare un nuovo nome al campo.

Argomento	Descrizione
inline	**inline** viene utilizzato quando i dati devono essere immessi direttamente nello script e non caricati da un file. *data ::= [text]* I dati immessi mediante una clausola **inline** devono essere racchiusi tra virgolette doppie o tra parentesi quadre. Il testo tra parentesi o virgolette viene interpretato allo stesso modo del contenuto di un file. Pertanto, quando si desidera modificare o immettere una nuova riga in un file di testo, è necessario eseguire questa operazione anche nel testo di una clausola **inline**, ad esempio premendo il tasto INVIO mentre si digita lo script. Il numero di colonne è definito nella prima riga. *format-spec ::= (fspec-item {, fspec-item })* La specifica del formato è costituita da un elenco di più voci di specifica del formato, racchiuse tra parentesi.
resident	**resident** viene usato se i dati devono essere caricati da una tabella caricata in precedenza. *table label* è un'etichetta che precede l'istruzione **LOAD** che ha creato la tabella originale. L'etichetta dovrà essere indicata con i due punti finali.

Argomento	Descrizione
extension	È possibile caricare dati da connessioni di analisi. È necessario utilizzare la clausola **extension** per chiamare una funzione definita nel plug-in SSE (Server-Side Extension) o per valutare uno script. È possibile inviare una singola tabella al plug-in SSE, che restituisce una singola tabella di dati. Se il plug-in non specifica i nomi dei campi restituiti, i campi saranno denominati Field1, Field2 e così via. `Extension pluginname.functionname( tabledescription );` • Caricamento di dati mediante una funzione in un plug-in SSE *tabledescription ::= (table { ,tablefield})* Se i campi della tabella non vengono specificati, verranno utilizzati in ordine di caricamento. • Caricamento di dati mediante valutazione di uno script in un plug-in SSE *tabledescription ::= (script, table { ,tablefield})* **Gestione dei tipi di dati nella definizione dei campi della tabella** I tipi di dati sono riconosciuti automaticamente nelle connessioni di analisi. Se i dati non hanno valori numerici e hanno almeno una stringa di testo non NULL, il campo è considerato di tipo testo. In tutti gli altri casi è considerato di tipo numerico. È possibile forzare il tipo di dati racchiudendo il nome del campo in **String()** o **Mixed()**. • **String()** forza il tipo del campo su testo. Se il campo è numerico, viene estratta la parte di testo del valore duale, senza eseguire alcuna conversione. • **Mixed()** forza il tipo del campo su duale. **String()** o **Mixed()** non possono essere utilizzati esternamente alle definizioni dei campi della tabella **extension** e non è possibile utilizzare altre funzioni di Qlik Sense nella definizione di un campo della tabella.
where	**where** è una clausola utilizzata per dichiarare se un record deve essere incluso o meno nella selezione. La selezione viene inclusa se *criterion* è True. *criterion* è un'espressione logica.
while	**while** è una clausola che specifica se un record deve essere letto ripetutamente. Viene letto lo stesso record finché *criterion* è True. Per risultare utile, una clausola **while** deve generalmente includere la funzione **IterNo( )**. *criterion* è un'espressione logica.

Argomento	Descrizione
group by	**group by** è una clausola usata per definire su quali campi devono essere aggregati (raggruppati) i dati. I campi di aggregazione devono essere inclusi in qualche modo nelle espressioni caricate. Nessun altro campo tranne i campi di aggregazione può essere usato al di fuori delle funzioni di aggregazione nelle espressioni caricate. *groupbyfieldlist ::= (fieldname { ,fieldname })*
order by	**order by** è una clausola utilizzata per ordinare i record di una tabella residente prima che vengano elaborati da un'istruzione **load**. La tabella residente può essere ordinata su uno o più campi, in modo crescente o decrescente. L'ordinamento viene eseguito innanzitutto in base ai valori numerici e secondariamente in base all'ordine di confronto nazionale. Questa clausola può essere utilizzata solamente quando la sorgente dati è una tabella residente. I campi di ordinamento specificano il campo in base al quale viene ordinata la tabella residente. Il campo può essere specificato da un nome o dal suo numero nella tabella residente (il primo campo è il numero 1). orderbyfieldlist ::= fieldname [sortorder] { , fieldname [sortorder] } *sortorder* è *asc* per crescente o *desc* per decrescente. Se *sortorder* non è specificato, viene utilizzato *asc*. *fieldname*, *path*, *filename* e *aliasname* sono stringhe di testo che rappresentano ciò che implica il loro rispettivo nome. Qualsiasi campo presente nella tabella sorgente può essere utilizzato come *fieldname*. Tuttavia, i campi creati mediante la clausola as (*aliasname*) non appartengono all'ambito e non possono essere utilizzati all'interno della stessa istruzione **load**.

Let

L'istruzione **let** è un complemento all'istruzione **set**, utilizzata per definire le variabili degli script. L'istruzione **let**, a differenza dell'istruzione **set**, valuta l'espressione posta sul lato destro del simbolo '=' al tempo di esecuzione dello script prima dell'assegnazione alla variabile.

Sintassi:

```
Let variablename=expression
```

Esempi e risultati:

Esempio	Risultato

`Set x=3+4;`
`Let y=3+4;`
`z=$(y)+1;`
	`$(x)` verrà valutato come ' 3+4 '
	
	`$(y)` verrà valutato come ' 7 '
	
	`$(z)` verrà valutato come ' 8 '
	
	Notare la differenza tra le istruzioni **Set** e **Let**. L'istruzione **Set** assegna la stringa '3+4' alla variabile, mentre l'istruzione **Let** valuta la stringa e assegna 7 alla variabile.

Let T=now( );	`$(T)` verrà restituito il valore dell'ora attuale.

Set

L'istruzione **set** viene utilizzata per definire le variabili di script. Le variabili possono essere utilizzate per sostituire stringhe, percorsi, unità e così via.

Sintassi:

```
Set variablename=string
```

Example 1:

```
Set FileToUse=Data1.csv;
```

Example 2:

```
Set Constant="My string";
```

Example 3:

```
Set BudgetYear=2012;
```

Put

L'istruzione **put** viene utilizzata per impostare alcuni valori numerici nell'ipercubo.

È possibile accedere alle colonne tramite etichette. È possibile inoltre accedere a colonne e righe tramite ordine di dichiarazione. Per ulteriori informazioni, vedere gli esempi di seguito.

Sintassi:

```
put column(position)=value
```

Example 1:

È possibile accedere alle colonne tramite etichette.

Questo esempio imposterà un valore di `1` nella prima posizione della colonna con etichetta *Vendite*.

```
Put Sales(1) = 1;
```

Example 2:

È possibile accedere alle colonne di misura tramite ordine di dichiarazione utilizzando il formato per misure `#hc1.measure`.

Questo esempio imposterà il valore 1000 nella decima posizione dell'ipercubo ordinato finale.

```
Put #hc1.measure.2(10) = 1000;
```

Example 3:

È possibile accedere alle righe di dimensione tranite ordine di dichiarazione utilizzando il formato per dimensioni `#hc1.dimension`.

Questo esempio inserisce il valore della costante Pi nella quinta riga della terza dimensione dichiarata.

```
Put #hc1.dimension.3(5) = Pi();
```

Se non sono presenti dimensioni o espressioni di questo tipo, in valore o etichette, viene restituito un errore che indica che la colonna non è stata trovata. Se l'indice della colonna non è compreso nell'intervallo, non viene visualizzato alcun errore.

HCValue

La funzione **HCValue** è utilizzata per recuperare valori in una riga di una colonna specificata.

Sintassi:

```
HCValue(column,position)
```

Example 1:

Questo esempio restituisce il valore alla prima posizione della colonna con etichetta "Vendite".

```
HCValue(Sales,1)
```

Example 2:

Questo esempio restituisce il valore alla decima posizione dell'ipercubo ordinato.

```
HCValue(#hc1.measure.2,10)
```

Example 3:

Questo esempio restituisce il valore alla quinta riga della terza dimensione.

```
HCValue(#hc1.dimension.3,5)
```

Se non sono presenti dimensioni o espressioni di questo tipo, in valore o etichette, viene restituito un errore che indica che la colonna non è stata trovata. Se l'indice della colonna non è compreso nell'intervallo, viene restituito NULL.

7 Funzioni e istruzioni di QlikView non supportate in Qlik Sense

La maggior parte delle funzioni e delle istruzioni utilizzabili negli script di caricamento e nelle espressioni grafiche di QlikView è supportata anche in Qlik Sense, tuttavia vi sono alcune eccezioni che sono descritte di seguito.

7.1 Istruzioni di script non supportate in Qlik Sense

Istruzioni di script QlikView non supportate in Qlik Sense

Istruzione	Commenti
Command	Utilizzare invece **SQL**.
InputField	

7.2 Funzioni non supportate in Qlik Sense

In questo elenco sono descritte le funzioni grafiche e di script di QlikView non supportate in Qlik Sense.

- **GetCurrentField**
- **GetExtendedProperty**
- **Input**
- **InputAvg**
- **InputSum**
- **MsgBox**
- **NoOfReports**
- **ReportComment**
- **ReportId**
- **ReportName**
- **ReportNumber**

7.3 Prefissi non supportati in Qlik Sense

Questo elenco presenta i prefissi di QlikView non supportati da Qlik Sense.

- **Bundle**
- **Image_Size**
- **Info**

8 Funzioni e istruzioni non consigliate in Qlik Sense

La maggior parte delle funzioni e delle istruzioni che è possibile utilizzare negli script di caricamento e nelle espressioni grafiche di QlikView è supportata anche in Qlik Sense, tuttavia l'utilizzo di alcune funzioni non è consigliato in Qlik Sense. Vi sono inoltre funzioni e istruzioni disponibili in versioni precedenti di Qlik Sense che sono stati deprecati.

Per motivi di compatibilità, tali funzioni continueranno a funzionare come previsto, tuttavia è consigliabile aggiornare il codice come illustrato in questa sezione, in quanto esse potrebbero essere rimosse dalle versioni successive.

8.1 Istruzioni di script non consigliate in Qlik Sense

In questa tabella sono riportate le istruzioni di script di cui è sconsigliato l'utilizzo in Qlik Sense.

Istruzioni di script non consigliate

Istruzione	Consiglio
Command	Utilizzare invece **SQL**.
CustomConnect	Utilizzare invece **Custom Connect**.

8.2 Parametri dell'istruzione di script non consigliati in Qlik Sense

In questa tabella sono riportati i parametri dell'istruzione di script di cui è sconsigliato l'utilizzo in Qlik Sense.

Parametri dell'istruzione di script non consigliati

Istruzione	Parametri
Buffer	Utilizzare **Incremental** invece di: • **Inc** (non consigliato) • **Incr** (non consigliato)

Istruzione	Parametri
LOAD	Le seguenti parole chiave di parametri vengono generate dai wizard di trasformazione dei file di QlikView. La funzionalità viene mantenuta quando i dai vengono ricaricati, ma Qlik Sense non fornisce supporto guidato/wizard per la generazione dell'istruzione con questi parametri: • Bottom • Cellvalue • Col • Colmatch • Colsplit • Colxtr • Compound • Contain • Equal • Every • Expand • Filters • Intarray • Interpret • Length • Longer • Numerical • Pos • Remove • Rotate • Row • Rowcnd • Shorter • Start • Strcnd • Top • Transpose • Unwrap • XML: XMLSAX and Pattern is Path

8.3 Funzioni non consigliate in Qlik Sense

In questa tabella sono riportate le funzioni grafiche e di script di cui è sconsigliato l'utilizzo in Qlik Sense.

Funzioni non consigliate

Funzione	Consiglio
NumAvg **NumCount** **NumMax** **NumMin** **NumSum**	Utilizzare invece le funzioni di scala. *Funzioni di scala (page 1321)*
Color() **QliktechBlue** **QliktechGray**	Utilizzare invece altre funzioni colore. **QliktechBlue()** può essere sostituito da **RGB(8, 18, 90)** e **QliktechGray** può essere sostituito da **RGB(158, 148, 137)** per ottenere gli stessi colori. *Funzioni colore (page 547)*
QlikViewVersion	Utilizzare invece **EngineVersion**. *EngineVersion (page 1466)*
ProductVersion	Utilizzare invece **EngineVersion**. *EngineVersion (page 1466)*
QVUser	
Year2Date	Utilizzare invece **YearToDate**.
Vrank	Utilizzare invece **Rank**.
WildMatch5	Utilizzare invece **WildMatch**.

Qualificatore ALL

In QlikView, il qualificatore **ALL** può essere posizionato prima di un'espressione. Ciò equivale a utilizzare **{1} TOTAL**. In tal caso, il calcolo verrà effettuato in base a tutti i valori del campo nel documento, ignorando le dimensioni del grafico e le selezioni correnti. Viene restituito sempre lo stesso valore, indipendentemente dallo stato logico nel documento. Se si utilizza il qualificatore **ALL**, non è possibile utilizzare un'espressione di gruppo, perché tale qualificatore **ALL** definisce di per sé un gruppo. Per motivi di legacy, il qualificatore **ALL** è ancora valido anche in questa versione di Qlik Sense, ma potrebbe essere rimosso nelle prossime versioni.