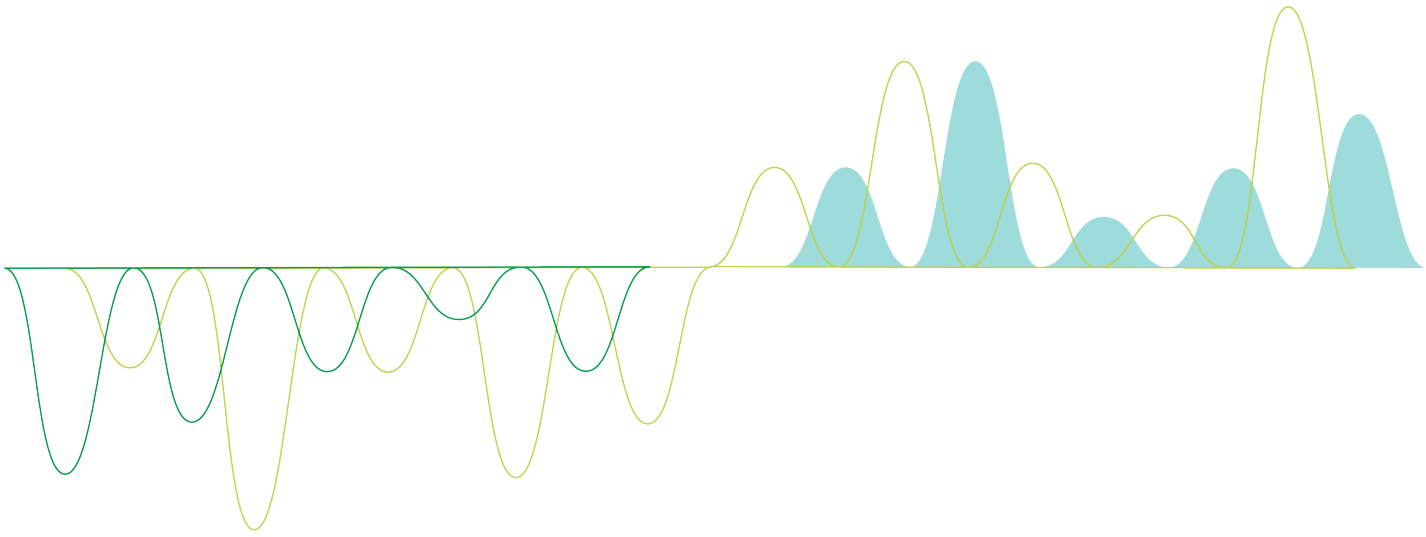
教學課程 - 指令碼撰寫中的後續步驟

Qlik Sense®
February 2024



Contents

1 歡迎使用本教學課程! **5**
1.1 您將會學習的內容 5
1.2 誰應進行本課程 5
1.3 套件內容 5
1.4 本教學課程的授課單元 6
1.5 進一步讀取和資源 6
2 LOAD 與 SELECT 陳述式 **7**
3 轉換資料 **8**
3.1 使用 Crosstable 前置詞 8
Crosstable 前置詞 8
清除記憶體快取 12
3.2 使用 Join 與 Keep 來合併表格 12
Join 13
使用 Join 13
Keep 15
Inner 16
Left 17
Right 18
3.3 使用記錄間函數:Peek、Previous 和 Exists 20
Peek() 20
Previous() 20
Exists() 20
使用 Peek() 和 Previous() 20
使用 Exists() 23
3.4 配對間隔和反覆載入 26
使用 IntervalMatch() 前置詞 26
使用 While 迴圈和反覆載入 IterNo() 28
開放和封閉的間隔 30
4 資料清理 **31**
4.1 對應表格 31
規則: 31
4.2 Mapping 函數和陳述式 31
4.3 對應前置詞 31
4.4 ApplyMap() 函數 32
4.5 MapSubstring() 函數 34
4.6 Map ... Using 36
5 處理階層式資料 **38**
5.1 Hierarchy 前置詞 38
5.2 HierarchyBelongsTo 前置詞 39
授權 40
6 QVD 檔案 **43**
6.1 建立 QVD 檔案 43
Store 44
6.2 從 QVD 檔案讀取資料 45
Buffer 46

6.3 謝謝您！ 49

1 歡迎使用本教學課程!

歡迎使用本教學課程,本課程將引導您進一步瞭解 Qlik Sense 中的指令碼。

熟悉指令碼撰寫的基本事項後,即可在您將資料載入 Qlik Sense 時,開始對資料執行更複雜的操作。例如,這可能包括使用跨維度資料表轉換資料、清除資料以及從稱為 QVD 檔案的 Qlik 資料檔案建立和載入資料。

1.1 您將會學習的內容

After completing this tutorial, you should be comfortable with loading data using some of the more advanced scripting functions in Qlik Sense.

1.2 誰應進行本課程

您應熟悉 Qlik Sense 的指令碼撰寫基本事項。也就是說,您已使用指令碼載入資料和操縱資料。

若您尚未這麼做,建議完成初學者教學課程的指令碼撰寫。

您需要存取資料載入編輯器並且應能在 Qlik Sense Enterprise on Windows 載入資料。

這些說明通常也適用於 Qlik Sense Cloud 企業版。

1.3 套件內容

您下載的 ZIP 套件包含您為了完成教學課程所需要的以下資料檔案:

- *Cutlery.xlsx*
- *Data.xlsx*
- *Events.txt*
- *Employees.xlsx*
- *Intervals.txt*
- *Product.xlsx*
- *Salesman.xlsx*
- *Transactions.csv*
- *Winedistricts.txt*

套件也包含*進階指令碼教學課程*應用程式的副本。應用程式中的其他指令碼區段包含您在本教學課程中建立之其他應用程式的指令碼。您可以將應用程式上傳至中心。

我們建議您按照教學課程中的說明,自行建立應用程式,以最大化您的學習效率。此外,您必須按照教學課程中的說明,上傳並連線至資料檔案,讓資料載入能夠運作。

不過,若您遇到問題,應用程式可協助您排解。我們已指出哪些指令碼區段與每一個課程有關聯。

1.4 本教學課程的授課單元

根據您使用 Qlik Sense 的經驗，本教學課程應花費 3-4 小時來完成。主題的設計應按順序完成。不過，您可以隨時離開並返回。讓人慶幸的是沒有測驗。

轉換資料
使用 Crosstable 前置詞
使用 Join 與 Keep 來合併表格
使用記錄間函數：Peek、Previous 和 Exists
配對間隔和反覆載入
資料清理
處理階層式資料
QVD 檔案

1.5 進一步讀取和資源

- 當您想要進一步瞭解時，Qlik 會提供各種資源。
- 提供 Qlik 線上說明。
- 在 Qlik Continuous Classroom 提供培訓，包括免費線上課程。
- 可在 Qlik Community 找到討論論壇、部落格等。

2 LOAD 與 SELECT 陳述式

您可以使用 LOAD 與 SELECT 陳述式，將資料載入至 Qlik Sense。每個陳述式都會產生一個內部表格。LOAD 用來從檔案載入資料，而 SELECT 用來從資料庫載入資料。

在本教學課程中，您將會使用檔案中的資料，所以您會使用 LOAD 陳述式。

您也可以使用前置 LOAD，以便能夠操縱載入的資料內容。例如，重新命名欄位必須在 LOAD 陳述式進行，但 SELECT 陳述式不允許對欄位名稱做出任何變更。

Qlik Sense 載入資料時適用下列規則：

- Qlik Sense 不會區別 LOAD 或 SELECT 陳述式所產生的表格。這表示如果載入數個表格，完全不會去管表格是由 LOAD 或 SELECT 陳述式載入，還是由兩者混合的方式載入。
- 陳述式中的欄位順序，或資料庫內原始表格中的欄位順序，對 Qlik Sense 邏輯而言都是不重要的。
- 欄位名稱區分大小寫，用來建立資料表格之間的關聯。因此，有時有必要在載入指令碼中重新命名欄位，以達成所需的資料模型。

3 轉換資料

在應用程式中使用資料之前，您可以在資料載入編輯器中轉換並操縱資料。

資料處理的其中一個好處是您可以選擇只從檔案中載入資料的子集 (例如從表格中載入幾個選定資料行)，從而更有效地處理資料。您還可以多次載入資料，以將原始資料分割成幾個新的邏輯表格。還可從多個來源載入資料，然後在 Qlik Sense 中合併成一個表格。

下列活動將會向您顯示如何使用 Crosstable 前置詞載入資料。您還學習到如何使用 Peek 和 Previous 等記錄間函數聯結表格，並使用 While Load 數次載入相同的列。

3.1 使用 Crosstable 前置詞

跨維度資料表是一種常見的表格類型，在兩個標頭資料的正交清單之間具有值矩陣。每當擁有資料的跨維度資料表時，您都可以使用 Crosstable 前置詞轉換資料，並建立所需的欄位。

Crosstable 前置詞

在下面的 *Product* 表格中，每個月有一個資料行，每個產品有一列。

產品表格

產品	Jan 2014	Feb 2014	Mar 2014	Apr 2014	May 2014	Jun 2014
A	100	98	100	83	103	82
B	284	279	297	305	294	292
C	50	53	50	54	49	51

載入表格時，在輸出的表格中，一個欄位用於 *Product*，然後每個月份各一個欄位。

Product 表格含有 Product 欄位，每個月份各一個欄位

如果您想要分析此資料，最好將所有數字放到一個欄位，所有月份放到另一個欄位。在此情況下，在擁有三個資料行的表格中，每個類別 (*Product, Month, Sales*) 都有一個欄位。

Product 表格含有 Product、Month 和 Sales 欄位

Crosstable 前置詞將資料轉換為表格，一個資料行表示 *Month*，另一個表示 *Sales*。另一種表達方式為它採用了欄位名稱並將其轉換為欄位值。

請執行下列動作：

1. 建立新的應用程式並稱之為*進階指令碼教學課程*。
2. 在**資料載入編輯器**中新增新的指令碼區段。
3. 命名區段 *Product*。
4. 在右側功能表的 **AttachedFiles** 之下，按一下**選取資料**。
5. 上傳然後選取 *Product.xlsx*。
6. 在**從以下項目選取資料**視窗中選取 *Product* 表格。

   *在**欄位名稱**下，確保已選取**內嵌欄位名稱**，以便載入資料時包含表格欄位的名稱。*
7. 按一下**插入指令碼**。
   您的指令碼應如下所示：

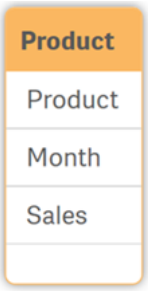
```
LOAD
    Product,
    "Jan 2014",
    "Feb 2014",
    "Mar 2014",
    "Apr 2014",
    "May 2014",
    "Jun 2014"
FROM [lib://AttachedFiles/Product.xlsx]
(ooxml, embedded labels, table is Product);
```

8. 按一下**載入資料**。
9. 開啟**資料模型檢視器**。資料模型應如下所示：

Product 表格含有 Product 欄位，每個月份各一個欄位

10. 在**資料載入編輯器**中，按一下 *Product* 索引標籤。
11. 在 LOAD 陳述式之上輸入下列內容：
    ```
    CrossTable(Month, Sales)
    ```
12. 按一下**載入資料**。
13. 開啟**資料模型檢視器**。資料模型應如下所示：

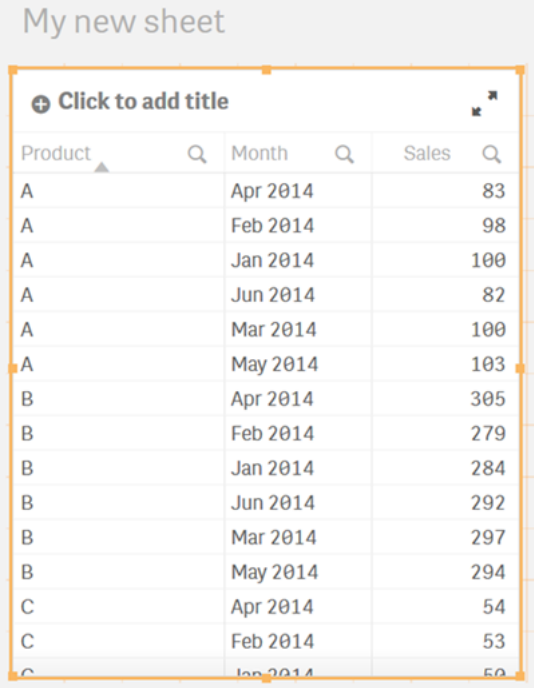
    Product 表格含有 Product、Month 和 Sales 欄位



請注意，輸入資料通常只有一個作為限定詞欄位的資料行；作為內部索引鍵 (在上方範例中為 *Product*)。但是您可以擁有多個。如果是這種情況，所有符合條件的欄位必須在 LOAD 陳述式屬性欄位前面列出，並且 Crosstable 前置詞的第三個參數必須用於定義符合條件欄位的數量。您不能在 Crosstable 關鍵字前面放置前置 LOAD 或前置詞。不過，您可以使用自動串連。

在 Qlik Sense 的表格中，資料呈現如下：

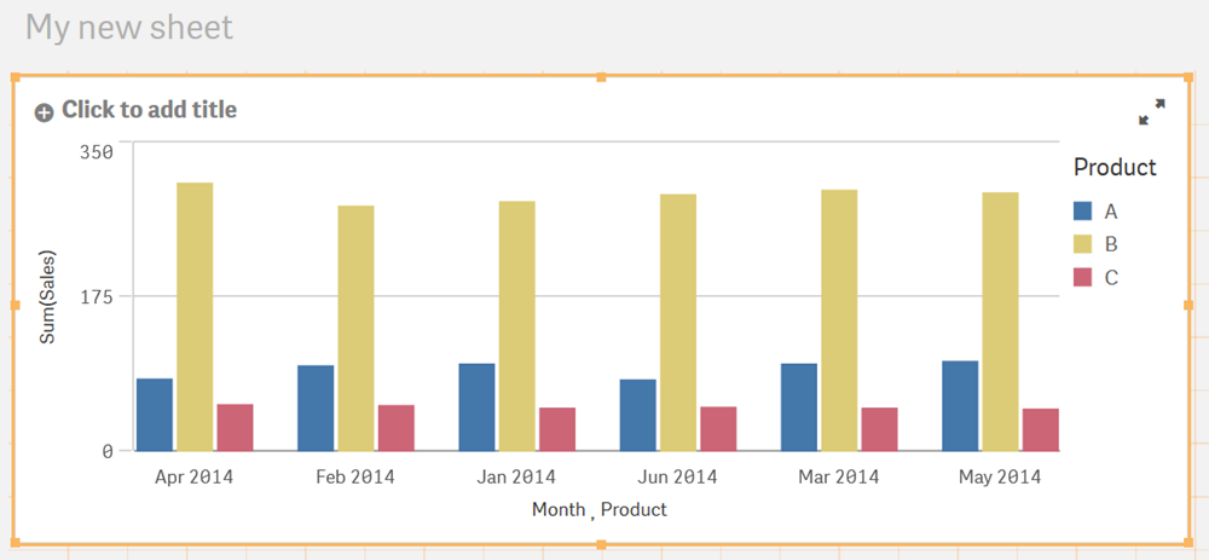
表格中顯示使用 Crosstable 前置詞載入的資料



Product	Month	Sales
A	Apr 2014	83
A	Feb 2014	98
A	Jan 2014	100
A	Jun 2014	82
A	Mar 2014	100
A	May 2014	103
B	Apr 2014	305
B	Feb 2014	279
B	Jan 2014	284
B	Jun 2014	292
B	Mar 2014	297
B	May 2014	294
C	Apr 2014	54
C	Feb 2014	53

例如，您現在可以使用資料建立條形圖：

條形圖中顯示使用 Crosstable 前置詞載入的資料



若要進一步瞭解 Crosstable，請在 Qlik Community 中參閱此篇部落格貼文：Crosstable 載入。行為會在 QlikView 的上下文中討論。不過，邏輯同樣適用於 Qlik Sense。

數值解譯無法用於屬性欄位。也就是說，如果您的資料行標頭是月份，系統不會自動解譯。解決方法是使用 Crosstable 前置詞建立一個臨時表格，並執行第二次通過來解譯，如以下範例所示。

請注意，這只是範例。在 Qlik Sense 中沒有要完成的附帶活動。

```
tmpData:
Crosstable (MonthText, Sales)
LOAD Product, [Jan 2014], [Feb 2014], [Mar 2014], [Apr 2014], [May 2014], [Jun 2014]
FROM ...

Final:
LOAD Product,
Date(Date#(MonthText,'MMM YYYY'),'MMM YYYY') as Month,
Sales
Resident tmpData;
Drop Table tmpData;
```

清除記憶體快取

您可以刪除您建立的表格，以清除記憶體快取。當您如之前章節所述載入臨時項目時，您應在不再需要該表格時將其捨棄。例如：

```
DROP TABLE Table1, Table2, Table3, Table4;
DROP TABLES Table1, Table2, Table3, Table4;
```

您也可以拖放欄位。例如：

```
DROP FIELD Field1, Field2, Field3, Field4;
DROP FIELDS Field1, Field2, Field3, Field4;
DROP FIELD Field1 from Table1;
DROP FIELDS Field1 from Table1;
```

如您所見，關鍵字 TABLE 和 FIELD 可以是單數或複數。

3.2 使用 Join 與 Keep 來合併表格

聯結是將兩個表格結合成一個表格的操作。產生的表格記錄是原始表格記錄的組合，通常以這種方式在所產生表格中形成任何指定組合的兩個記錄，會具有一或數個通用欄位的通用值，也稱為「自然聯結」。在 Qlik Sense 中，可在指令碼中建立聯結，進而產生邏輯表。

可聯結已在指令碼中的表格。之後，Qlik Sense 邏輯不會看到個別的表格，而是聯結的結果，亦即單一的內部表格。在某些情況下這是必要的，但會有一些缺點：

- 載入的表格通常會變得較大，使得 Qlik Sense 運作較慢。
- 某些資訊可能會遺失：原始表格內的頻率 (記錄筆數) 可能不再可用。

Keep 功能可將兩個表格的其中一個或兩個都先減少到表格資料的交集，然後才儲存到 Qlik Sense 中，這是專門設計用來減少需要使用明確聯結的情況。

本手冊中的「聯結」一詞通常指的是內部表格建立之前所進行的聯結。不過，在內部表格建立之後所進行的關聯基本上也算是聯結。

Join

進行聯結的最簡單方式，是在指令碼中使用 Join 前置詞，如此會將內部表格聯結到另一個具名的表格，或是聯結到上一個建立的表格。這樣的聯結算是外部聯結，會從兩個表格建立所有可能的值組合。

範例：

```
LOAD  a, b, c from table1.csv;
join LOAD a, d from table2.csv;
```

產生的內部表格會有欄位 a、b、c 及 d。而記錄筆數會視這兩個表格的欄位值而定。

要聯結之欄位的名稱必須完全相同。要聯結的欄位數目可任意。通常表格應會有一個或數個通用欄位。若無通用欄位，則會呈現表格的笛卡兒座標的乘積。也可能所有欄位都是通用的，但這通常毫無意義。除非 Join 陳述式中指定先前已載入表格的表格名稱，否則 Join 前置詞會使用上一個建立的表格。因此，兩個陳述式的順序不可任意排序。

使用 Join

Qlik Sense 指令碼語言中明確的 Join 前置詞會執行兩個表格的完整聯結。結果會是一個表格。這樣的聯結通常會產生非常大的表格。

請執行下列動作：

1. 開啟*進階指令碼教學課程*應用程式。
2. 在**資料載入編輯器**中新增新的指令碼區段。
3. 叫用區段 *Transactions*。
4. 在右側功能表的 **AttachedFiles** 之下，按一下**選取資料**。
5. 上傳然後選取 *Transactions.csv*。

   *在**欄位名稱**下，確保已選取**內嵌欄位名稱**，以便載入資料時包含表格欄位的名稱。*
6. 在**從以下項目選取資料**視窗中，按一下**插入指令碼**。
7. 上傳然後選取 *Salesman.xlsx*。
8. 在**從以下項目選取資料**視窗中，按一下**插入指令碼**。

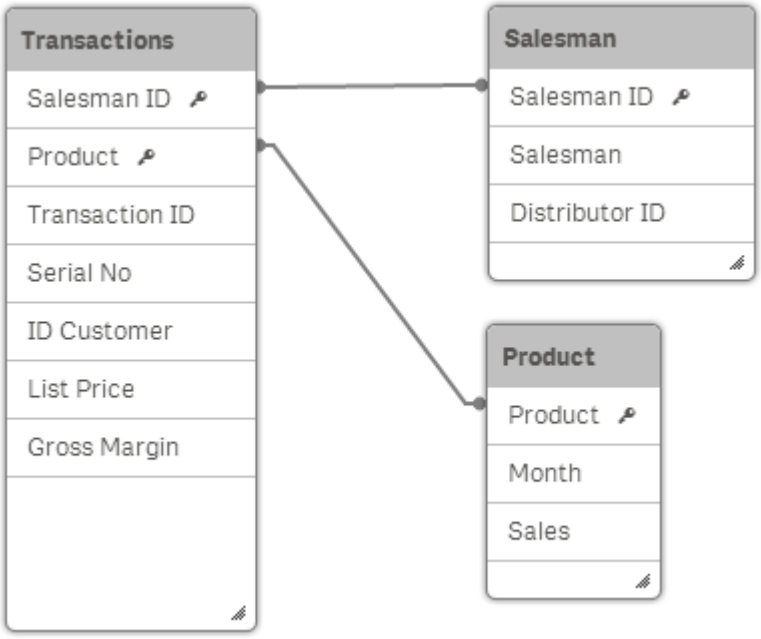
   您的指令碼應如下所示：

   ```
   LOAD
       "Transaction ID",
       "Salesman ID",
       Product,
       "Serial No",
       "ID Customer",
       "List Price",
       "Gross Margin"
   ```

```
FROM [lib://AttachedFiles/Transactions.csv]
(txt, codepage is 28591, embedded labels, delimiter is ',', msq);

LOAD
    "Salesman ID",
    Salesman,
    "Distributor ID"
FROM [lib://AttachedFiles/Salesman.xlsx]
(ooxml, embedded labels, table is Salesman);
```

9. 按一下**載入資料**。
10. 開啟**資料模型檢視器**。資料模型應如下所示：

    資料模型：Transactions、Salesman 和 Product 表格



不過，讓 *Transactions* 和 *Salesman* 表格分離可能不是所需結果。這可能比聯結兩個表格更好。

請執行下列動作：

1. 若要設定聯結表格的名稱，在第一個 LOAD 陳述式之上新增下列行：

   ```
   Transactions:
   ```

2. 若要聯結 *Transactions* 和 *Salesman* 表格，在第二個 LOAD 陳述式之上新增下列行：

   ```
   Join(Transactions)
   ```

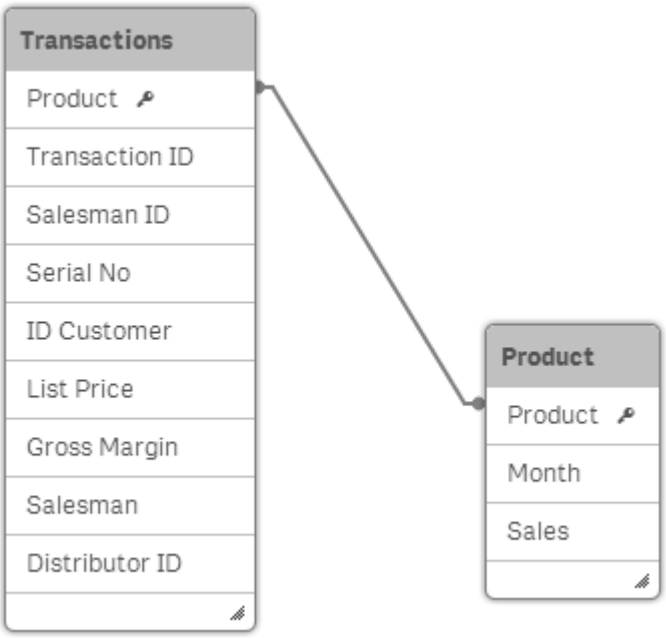
   您的指令碼應如下所示：

   ```
   Transactions:
   LOAD
       "Transaction ID",
       "Salesman ID",
       Product,
   ```

```
    "Serial No",
    "ID Customer",
    "List Price",
    "Gross Margin"
FROM [lib://AttachedFiles/Transactions.csv]
(txt, codepage is 28591, embedded labels, delimiter is ',', msq);

Join(Transactions)
LOAD
    "Salesman ID",
    Salesman,
    "Distributor ID"
FROM [lib://AttachedFiles/Salesman.xlsx]
(ooxml, embedded labels, table is Salesman);
```

3. 按一下**載入資料**。
4. 開啟**資料模型檢視器**。資料模型應如下所示：

   資料模型：Transactions 和 Product 表格



Transactions 和 *Salesman* 表格中的所有欄位現已合併成為單一的 *Transactions* 表格。

*若要進一步瞭解使用 Join 的時機，請在 Qlik Community 中參閱這些部落格貼文：*To Join or not to Join (是否要聯結)、Mapping as an Alternative to Joining (作為替代選項對應聯結)。*行為會在 QlikView 的上下文中討論。不過，邏輯同樣適用於 Qlik Sense。*

Keep

Qlik Sense 的其中一個主要特色是可在表格之間建立關聯而不是聯結，這可降低所佔的記憶體空間、提高速度，並帶來極大的靈活性。Keep 功能是專門用來減少需要使用明確聯結的情況。

兩個 LOAD 或 SELECT 陳述式之間的 Keep 前置詞，可將這兩個表格的其中一個或兩個都先減少到表格資料的交集，然後才儲存到 Qlik Sense 中。Keep 前置詞前面一律必須加上 Inner、Left 或 Right 前置詞的其中一個。從表格選取記錄的方式和對應聯結所使用的方式相同。不過，兩個表格並未聯結，而是以兩個個別的具名表格儲存在 Qlik Sense 中。

Inner

在資料載入指令碼語言中，Join 和 Keep 前置詞的前面可以加上 Inner 前置詞。

若加在 Join 之前，是指定兩個表格之間的聯結必須為內部聯結。產生的表格僅會包含兩個表格之間，具備兩邊完整資料集的組合。

若加在 Keep 之前，是指定兩個表格應先減少到其共同交集，然後才儲存到 Qlik Sense 中。

範例：

在此範例中，我們使用來源表格 *Table1* 與 *Table2*。

請注意，這些只是範例。在 Qlik Sense 中沒有要完成的附帶活動。

Table 1

A	B
1	aa
2	cc
3	ee

Table2

A	C
1	xx
4	yy

Inner Join

首先，我們會在表格上執行 Inner Join，產生 *VTable*，僅包含一個列，存在於兩個表格中的唯一記錄，擁有從兩個表格合併的資料。

```
VTable:
SELECT * from Table1;
inner join SELECT * from Table2;
```

VTable

A	B	C
1	aa	xx

Inner Keep

如果我們改而執行 Inner Keep, 仍然會有兩個表格。兩個表格是透過通用欄位 *A* 建立關聯。

```
VTab1:
SELECT * from Table1;
VTab2:
inner keep SELECT * from Table2;
```

VTab1

A	**B**
1	aa

VTab2

A	**C**
1	xx

Left

在資料載入指令碼語言中, Join 和 Keep 前置詞的前面可以加上 left 前置詞。

若加在 Join 之前, 是指定兩個表格之間的聯結必須為左方聯結。產生的表格僅會包含兩個表格之間, 具備第一個表格完整資料集的組合。

若加在 Keep 之前, 是指定第二個表格應先減少到與第一個表格的共同交集, 然後才儲存到 Qlik Sense 中。

範例:

在此範例中, 我們使用來源表格 *Table1* 與 *Table2*。

Table1

A	**B**
1	aa
2	cc
3	ee

Table2

A	**C**
1	xx
4	yy

首先，我們會在表格上執行 Left Join，產生 *VTable*，包含來自 *Table1* 的所有列，合併 *Table2* 中符合列的欄位。

```
VTable:
SELECT * from Table1;
left join SELECT * from Table2;
```

VTable

A	B	C
1	aa	xx
2	cc	-
3	ee	-

如果我們改而執行 Left Keep，仍然會有兩個表格。兩個表格是透過通用欄位 *A* 建立關聯。

```
VTab1:
SELECT * from Table1;
VTab2:
left keep SELECT * from Table2;
```

VTab1

A	B
1	aa
2	cc
3	ee

VTab2

A	C
1	xx

Right

在 Qlik Sense 指令碼語言中，Join 和 Keep 前置詞的前面可以加上 right 前置詞。

若加在 Join 之前，是指定兩個表格之間的聯結必須為右方聯結。產生的表格僅會包含兩個表格之間，具備第二個表格完整資料集的組合。

若加在 Keep 之前，是指定第一個表格應先減少到與第二個表格的共同交集，然後才儲存到 Qlik Sense 中。

範例：

在此範例中，我們使用來源表格 *Table1* 與 *Table2*。

Table1

A	B
1	aa
2	cc
3	ee

Table2

A	C
1	xx
4	yy

首先，我們會在表格上執行 Right Join，產生 *VTable*，包含來自 *Table2* 的所有列，合併 *Table1* 中符合列的欄位。

```
VTable:
SELECT * from Table1;
right join SELECT * from Table2;
```

VTable

A	B	C
1	aa	xx
4	-	yy

如果我們改而執行 Right Keep，仍然會有兩個表格。兩個表格是透過通用欄位 *A* 建立關聯。

```
VTab1:
SELECT * from Table1;
VTab2:
right keep SELECT * from Table2;
```

VTab1

A	B
1	aa

VTab2

A	C
1	xx
4	yy

3.3 使用記錄間函數：Peek、Previous 和 Exists

當評估目前記錄需要先前載入的資料記錄值時，便會使用這些函數。

在本教學課程部分，我們將檢查 Peek()、Previous() 和 Exists() 函數。

Peek()

Peek() 會傳回表格中已載入的列的欄位值。可以指定列號，也可以指定表格。若未指定列數，將會使用上一個載入的記錄。

語法：

```
Peek(fieldname [ , row [ , tablename ] ] )
```

列必須是整數。0 代表第一筆記錄，1 代表第二筆記錄，依此類推。負數表示從表格結尾算起的順序。-1 代表上次記錄讀取。

如未指定列，則會採用 -1。

Tablename 是沒有結束分號的表格標籤。如未指定 *tablename*，則會採用目前表格。若用於 **LOAD** 陳述式之外，或參考其他表格，則必須包括 *tablename*。

Previous()

Previous() 會使用由於 **where** 子句而尚未捨棄的先前輸入記錄中的資料，來尋找 **expr** 運算式的值。在內部表格的第一筆記錄中，此函數會傳回 NULL。

語法：

```
Previous(expression)
```

Previous() 函數可能為巢狀函數，以存取更早之前的記錄。資料是直接擷取自輸入來源，因此也可參考尚未載入 Qlik Sense 的欄位，即使它們尚未儲存在關聯資料庫中亦然。

Exists()

Exists() 會判定特定欄位值是否已載入資料載入指令碼的欄位中。該函數會傳回 TRUE 或 FALSE，因此可以在 **LOAD** 陳述式或 **IF** 陳述式的 **where** 子句中使用。

語法：

```
Exists(field [, expression ] )
```

欄位必須存在於指令碼目前載入的資料中。*Expression* 是一個評估為欄位值以在指定欄位中搜尋的運算式。如果省略，會採用指定欄位中目前記錄的值。

使用 Peek() 和 Previous()

最簡形式的 Peek() 和 Previous() 用於識別表格內的特定值。以下是 *Employees* 表格中的資料樣本，您將會在此活動中載入。

來自員工表格的資料樣本

日期	已聘用	已解雇
1/1/2011	6	0
2/1/2011	4	2
3/1/2011	6	1
4/1/2011	5	2

目前，這只能收集月份、在職和離職人員的人數資料，因此我們將使用 Peek() 和 Previous() 函數，為 *Employee Count* 和 *Employee Var* 新增欄位，以觀察員工總數的每月差異。

請執行下列動作：

1. 開啟*進階指令碼教學課程*應用程式。
2. 在**資料載入編輯器**中新增新的指令碼區段。
3. 叫用區段 *Employees*。
4. 在右側功能表的 **AttachedFiles** 之下，按一下**選取資料**。
5. 上傳然後選取 *Employees.xlsx*。

   > *在 Field names 下，確保已選取 Embedded field names，以便載入資料時包含表格欄位的名稱。*

6. 在**從以下項目選取資料**視窗中，按一下**插入指令碼**。
   您的指令碼應如下所示：

```
LOAD
    "Date",
    Hired,
    Terminated
FROM [lib://AttachedFiles/Employees.xlsx]
(ooxml, embedded labels, table is Sheet1);
```

7. 修改指令碼，使其如下所示：

```
[Employees Init]:
LOAD
    rowno() as Row,
    Date(Date) as Date,
    Hired,
    Terminated,
    If(rowno()=1, Hired-Terminated,  peek([Employee Count], -1)+(Hired-Terminated)) as
[Employee Count]
FROM [lib://AttachedFiles/Employees.xlsx]
(ooxml, embedded labels, table is Sheet1);
```

   Excel 工作表的 *Date* 欄位中的日期格式為 MM/DD/YYYY。若要確保使用來自系統變數的格式正確解譯日期，則 Date 函數應套用至 *Date* 欄位。

使用 Peek() 函數，您可以為定義的欄位指定任何載入值。在該運算式中，我們要先查看 rowno() 是否等於 1。如果它等於 1，將不存在 *Employee Count*，所以我們會用 *Hired* 減去 *Terminated* 的差值來填充此欄位。

如果 rowno() 大於 1，我們會看看上個月的 *Employee Count*，並將這個數值加到該月的 *Hired* 減去 *Terminated* 員工的差值中。

同時請注意，在 Peek() 函數中我們使用了 (-1)。這是告訴 Qlik Sense 尋找位於目前記錄之上的記錄。如果未指定 (-1)，Qlik Sense 將假定您想要查看之前的記錄。

8. 將下列內容新增至指令碼結尾：

```
[Employee Count]:
LOAD
       Row,
       Date,
       Hired,
       Terminated,
       [Employee Count],
       If(rowno()=1,0,[Employee Count]-Previous([Employee Count])) as [Employee Var]
Resident [Employees Init] Order By Row asc;
Drop Table [Employees Init];
```

使用 Previous() 函數，您可以為定義的欄位指定載入的最後值。在該運算式中，我們要先查看 rowno() 是否等於 1。如果等於 1，我們知道將不存在 Employee Var，因為上個月的 *Employee Count* 沒有記錄。所以我們只是為該值輸入 0。

如果 rowno() 大於 1，我們知道有一個 *Employee Var* 值，因此我們可以看看上個月的 *Employee Count* 並從本月的 *Employee Count* 減去該數值，以建立 *Employee Var* 欄位中的值。

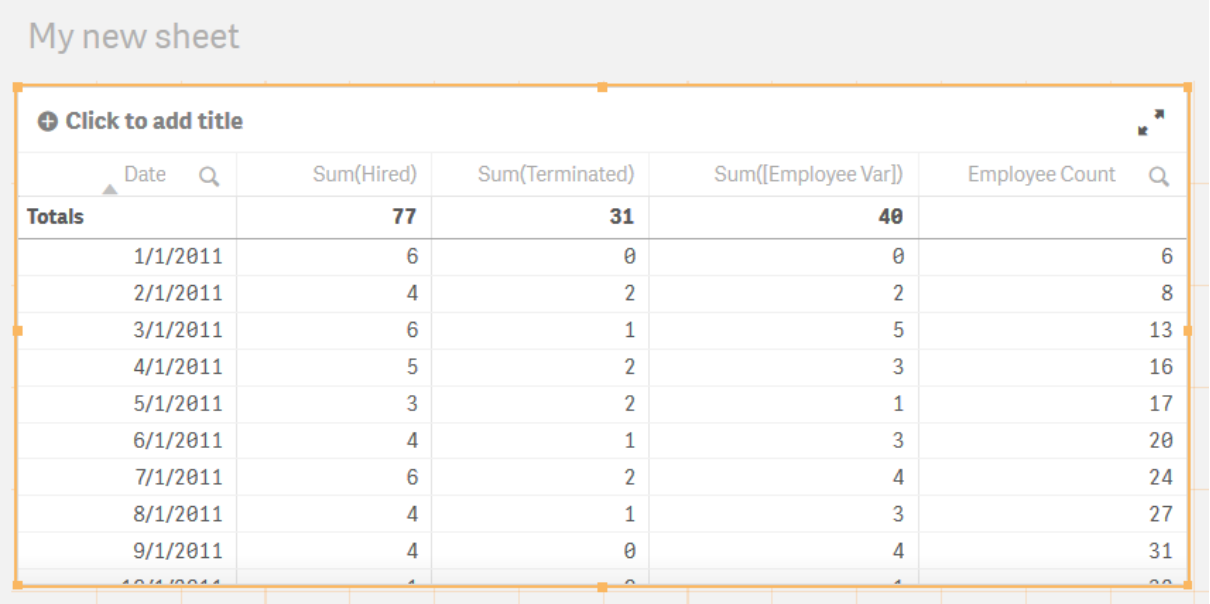
您的指令碼應如下所示：

```
[Employees Init]:
LOAD
    rowno() as Row,
    Date(Date) as Date,
    Hired,
    Terminated,
    If(rowno()=1, Hired-Terminated,  peek([Employee Count], -1)+(Hired-Terminated)) as
[Employee Count]
FROM [lib://AttachedFiles/Employees.xlsx]
(ooxml, embedded labels, table is Sheet1);

[Employee Count]:
LOAD
    Row,
    Date,
    Hired,
    Terminated,
    [Employee Count],
    If(rowno()=1,0,[Employee Count]-Previous([Employee Count])) as [Employee Var]
Resident [Employees Init] Order By Row asc;
Drop Table [Employees Init];
```

9. 按一下**載入資料**。

在應用程式概述的新工作表中，使用 *Date, Hired, Terminated, Employee Count* 和 *Employee Var* 作為表格的資料行來建立表格。產生的表格應如下所示：

使用指令碼中的 Peek 和 Previous 後的表格



Date	Sum(Hired)	Sum(Terminated)	Sum([Employee Var])	Employee Count
Totals	**77**	**31**	**40**	
1/1/2011	6	0	0	6
2/1/2011	4	2	2	8
3/1/2011	6	1	5	13
4/1/2011	5	2	3	16
5/1/2011	3	2	1	17
6/1/2011	4	1	3	20
7/1/2011	6	2	4	24
8/1/2011	4	1	3	27
9/1/2011	4	0	4	31

Peek() 和 Previous() 允許您以表格內的已定義列為目標。兩個函數之間最大的差異在於，Peek() 函數允許使用者查看先前未載入指令碼中的欄位，而 Previous() 函數只允許查看先前已載入的欄位。Previous() 對 LOAD 陳述式的輸入進行運算，而 Peek() 對 LOAD 陳述式的輸出進行運算。(與 RecNo() 和 RowNo() 之間的差異相同。) 這表示，如果您有 Where-子句，兩個函數的表現將有所不同。

因此，如果您需要顯示目前的值比對先前的值，更適合使用 Previous() 函數。在此範例中，我們按月計算員工的變化情況。

如果您的目標是先前未載入表格中的欄位，或如果您的目標是需要特定列，更適合使用 Peek() 函數。此範例中展示了我們透過查看到上個月的 *Employee Count* 並加上當月在職和離職員工人數的差值，計算出 *Employee Count*。請記住，*Employee Count* 不是原始檔案中的欄位

*若要進一步瞭解使用 Peek() 和 Previous() 的時機，請在 Qlik Community 中參閱此篇部落格貼文：*Peek() vs Previous() – When to Use Each*。行為會在 QlikView 的上下文中討論。不過，邏輯同樣適用於 Qlik Sense。*

使用 Exists()

在指令碼中，Exists() 函數通常與 Where 子句配合使用，以在相關資料已載入資料模型的情況下載入資料。

在以下範例中，我們還使用了 Dual() 函數來指派數值給字串。

請執行下列動作：

1. 建立新的應用程式並命名。
2. 在**資料載入編輯器**中新增新的指令碼區段。
3. 叫用區段 *People*。
4. 輸入以下指令碼：

```
//Add dummy people data
PeopleTemp:
LOAD * INLINE [
PersonID, Person
1, Jane
2, Joe
3, Shawn
4, Sue
5, Frank
6, Mike
7, Gloria
8, Mary
9, Steven,
10, Bill
];

//Add dummy age data
AgeTemp:
LOAD * INLINE [
PersonID, Age
1, 23
2, 45
3, 43
4, 30
5, 40
6, 32
7, 45
8, 54
9,
10, 61
11, 21
12, 39
];

//LOAD new table with people
People:
NoConcatenate LOAD
    PersonID,
    Person
Resident PeopleTemp;

Drop Table PeopleTemp;

//Add age and age bucket fields to the People table
Left Join (People)
```

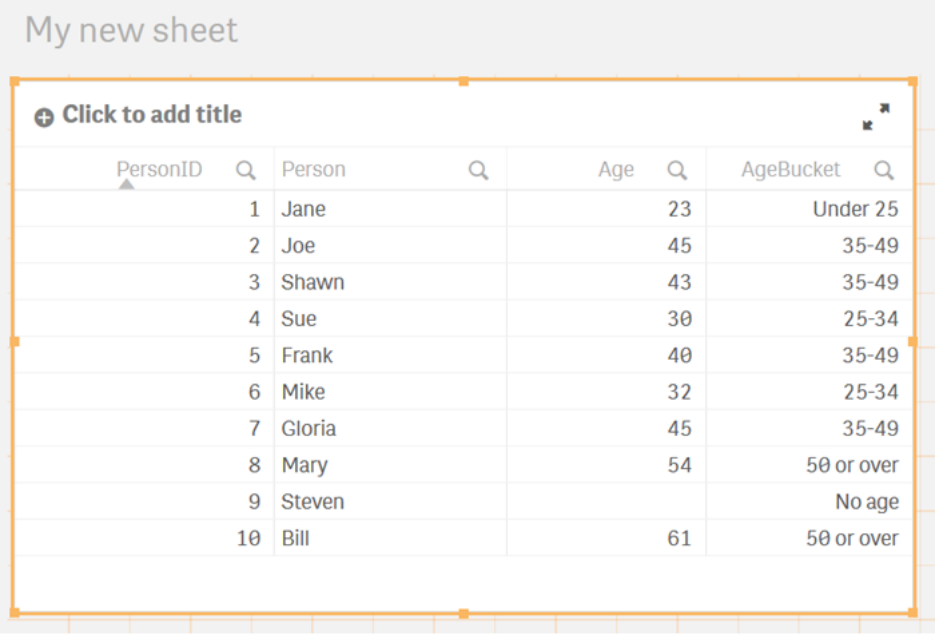
```
LOAD
    PersonID,
    Age,
       If(IsNull(Age) or Age='', Dual('No age', 5),
        If(Age<25, Dual('Under 25', 1),
         If(Age>=25 and Age <35, Dual('25-34', 2),
          If(Age>=35 and Age<50, Dual('35-49' , 3),
           If(Age>=50, Dual('50 or over', 4)
            ))))) as AgeBucket
Resident AgeTemp
Where Exists(PersonID);

DROP Table AgeTemp;
```

5. 按一下**載入資料**。
   在指令碼中，僅在 *PersonID* 已載入資料模型中的情況下才會載入 *Age* 和 *AgeBucket* 欄位。

   請注意，在 *AgeTemp* 表格中列出了 *PersonID* 為 11 和 12 的年齡，但由於這些 ID 未載入資料模型中 (在 *People* 表格中)，它們會被 Where Exists(PersonID) 子句排除。也可如下編寫此子句：Where Exists(PersonID, PersonID)。

   指令碼的輸出呈現如下：

   使用指令碼中的 Exists 後的表格

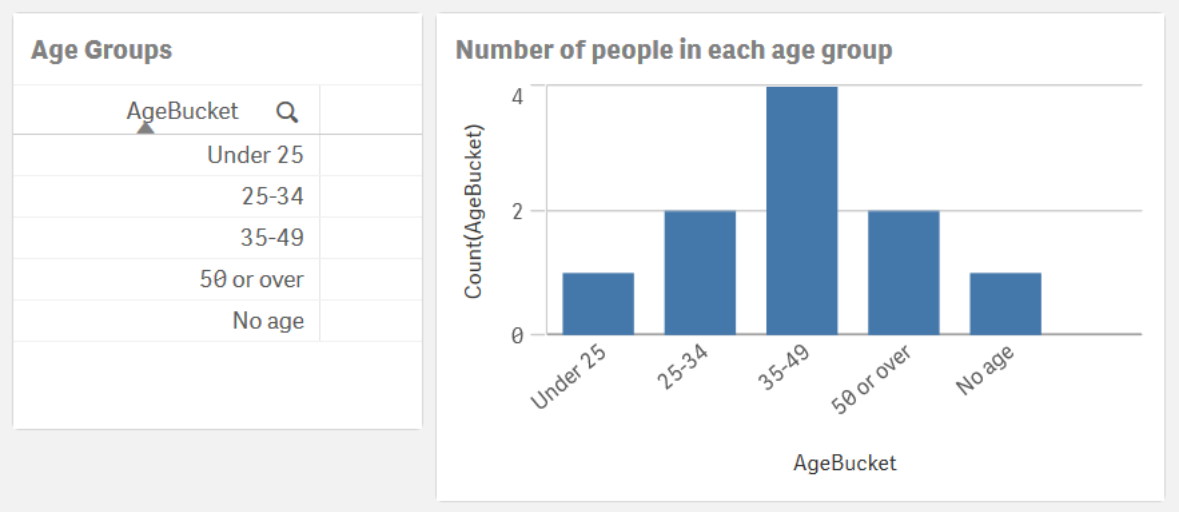


PersonID	Person	Age	AgeBucket
1	Jane	23	Under 25
2	Joe	45	35-49
3	Shawn	43	35-49
4	Sue	30	25-34
5	Frank	40	35-49
6	Mike	32	25-34
7	Gloria	45	35-49
8	Mary	54	50 or over
9	Steven		No age
10	Bill	61	50 or over

   如果資料模型中尚未載入 *AgeTemp* 表格中的任何 *PersonID*，那麼 *Age* 和 *AgeBucket* 欄位將不會聯結成為 *People* 表格。使用 Exists() 函數有助於防止資料模型中出現孤立記錄/資料，即不包含任何關聯人員的 *Age* 和 *AgeBucket* 欄位。

6. 建立新的工作表並命名。
7. 開啟新工作表並按一下**編輯工作表**。
8. 透過維度 *AgeBucket* 將標準表格新增至工作表，並將視覺化命名為*年齡群組*。

9. 將條形圖新增至含有維度 *AgeBucket* 以及量值 *Count([AgeBucket])* 的工作表。命名視覺化 *Number of people in each age group*。
10. 根據您的喜好調整表格和條形圖的屬性，然後按一下**完成**。
    您的工作表現應如下所示：
    依年齡分組的工作表



當需要指定數值給字串時，Dual() 函數在指令碼或圖表運算式中非常有用。

在上述指令碼中，您有一個載入年齡的應用程式，您決定把這些年齡放進儲存值區中，以便基於年齡儲存值區對比實際年齡建立視覺化。對於 25 歲以下、介於 25 和 35 歲之間的人有一個儲存值區，以此類推。透過使用 Dual() 函數，可以為年齡儲存值區指派一個稍後將用於在列表框或圖表中排列年齡儲存值區的數值。因此，和在應用程式工作表中一樣，排序會將 "No age" 排在清單末尾。

*若要進一步瞭解 Exists() 和 Dual()，請在 Qlik Community 中參閱此篇部落格貼文：*雙重 & 存在 – 實用函數

3.4 配對間隔和反覆載入

LOAD 或 SELECT 陳述式的 Intervalmatch 前置詞是用於將離散數值連結到一或多個數值間隔。這是非常強大的功能，例如可用於生產等環境中。

使用 IntervalMatch() 前置詞

最基本的間隔配對是當您在一個表格中擁有一系列數字或日期 (事件)，在另一個表格中擁有一系列間隔。您的目標是連結兩個表格。一般而言，這是多對多關係，即一個間隔包含位於其中的多個日期，並且一個日期也同屬於多個間隔。為了解決這個問題，您需要在兩個原始表格之間建立橋接表格。有多種方法可執行此操作。

在 Qlik Sense 中解決此問題的最簡單方法是在 LOAD 或 SELECT 陳述式前面加上 IntervalMatch() 前置詞。LOAD/SELECT 陳述式只需包含兩個欄位，即定義間隔的 From 和 To 欄位。IntervalMatch() 前置詞將產生已載入間隔與之前載入的數字欄位之間的所有組合，指定為該前置詞的參數。

請執行下列動作:

1. 建立新的應用程式並命名。
2. 在**資料載入編輯器**中新增新的指令碼區段。
3. 叫用區段 *Events*。
4. 在右側功能表的 **AttachedFiles** 之下, 按一下**選取資料**。
5. 上傳然後選取 *Events.txt*。
6. 在**從以下項目選取資料**視窗中, 按一下**插入指令碼**。
7. 上傳然後選取 *Intervals.txt*。
8. 在**從以下項目選取資料**視窗中, 按一下**插入指令碼**。
9. 在指令碼中, 將第一個表格命名為*事件*, 然後將第二個表格命名為 *Intervals*。
10. 在指令碼的結尾處新增 IntervalMatch 以建立第三個表格, 該表格橋接最初的兩個表格:

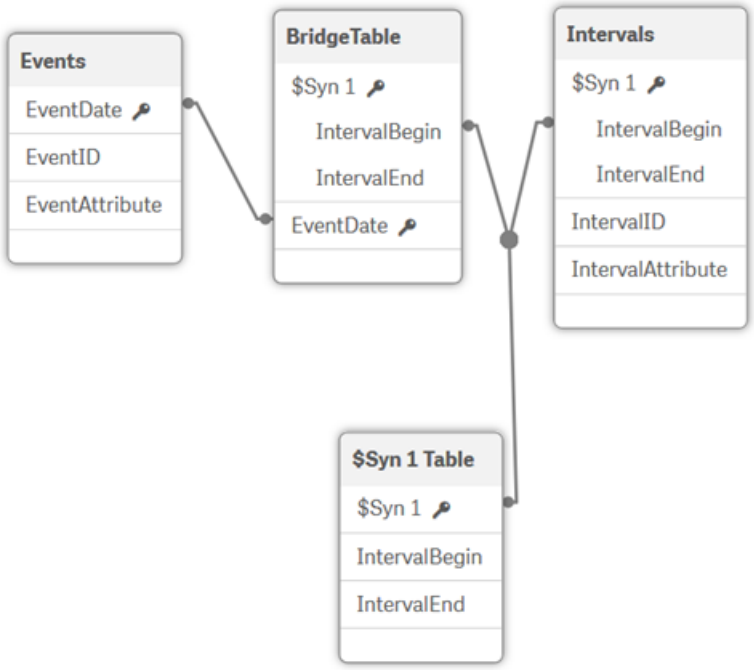
```
BridgeTable:
IntervalMatch (EventDate)
LOAD distinct IntervalBegin, IntervalEnd
Resident Intervals;
```

11. 您的指令碼應如下所示:

```
Events:
LOAD
    EventID,
    EventDate,
    EventAttribute
FROM [lib://AttachedFiles/Events.txt]
(txt, utf8, embedded labels, delimiter is '\t', msq);

Intervals:
LOAD
    IntervalID,
    IntervalAttribute,
    IntervalBegin,
    IntervalEnd
FROM [lib://AttachedFiles/Intervals.txt]
(txt, utf8, embedded labels, delimiter is '\t', msq);

BridgeTable:
IntervalMatch (EventDate)
LOAD distinct IntervalBegin, IntervalEnd
Resident Intervals;
```

12. 按一下**載入資料**。
13. 開啟**資料模型檢視器**。資料模型應如下所示:

資料模型：Events、BridgeTable、Intervals 和 $Syn1 表格



資料模型包含一個複合索引鍵 (*IntervalBegin* 和 *IntervalEnd* 欄位)，該索引鍵會將自身顯示為 Qlik Sense 合成鍵。

基本表格包括：

- 每個事件只包含一個記錄的 *Events* 表格。
- 每個間隔只包含一個記錄的 *Intervals* 表格。
- 每個事件和間隔組合只包含一個記錄、並且連結之前兩個表格的橋接表格。

請注意，如果間隔彼此重疊，一個事件可能同時屬於多個間隔。一個間隔顯然同時包括多個事件。

本資料模型是一個標準化的精簡模型，是您的最佳選擇。*Events* 表格和 *Intervals* 表格均保持不變，並包含記錄的原始數值。在這些表格中進行的所有 Qlik Sense 計算，例如 Count (EventID)，將正常工作並正確評估。

*若要進一步瞭解 IntervalMatch()，請在 Qlik Community 中參閱此篇部落格貼文：*使用 IntervalMatch()

使用 While 迴圈和反覆載入 IterNo()

您可使用 While 迴圈和 IterNo() 得到幾乎一樣的橋接表格，該表格在間隔的上下限之間建立可列舉數值。

在 LOAD 陳述式內的迴圈可使用 While 子句建立。例如：

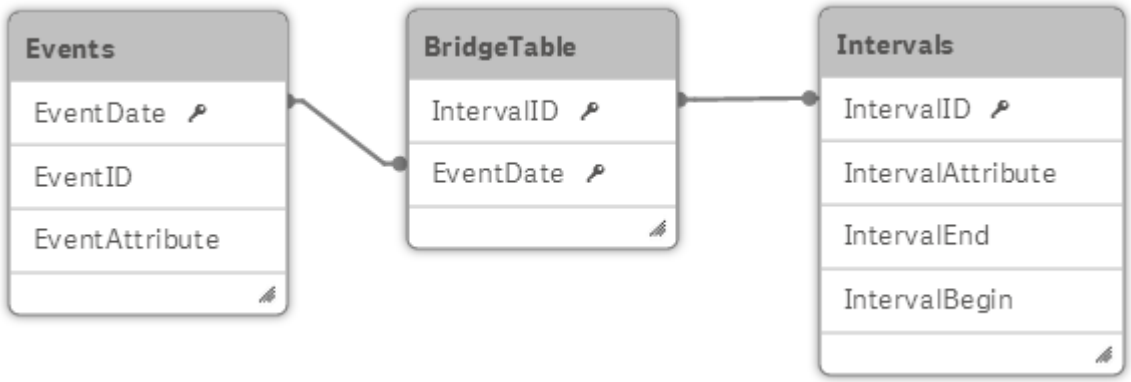
```
LOAD Date, IterNo() as Iteration From … While IterNo() <= 4;
```

只要 While 子句中的運算式為 true，此 LOAD 陳述式將重複每個輸入記錄並反覆載入。IterNo() 函數在首次反覆中返回「1」，在第二次中返回「2」，以此類推。

您有間隔的主要索引鍵 IntervalID，因此在指令碼中唯一的不同就是橋接表格的建立方式：

請執行下列動作：

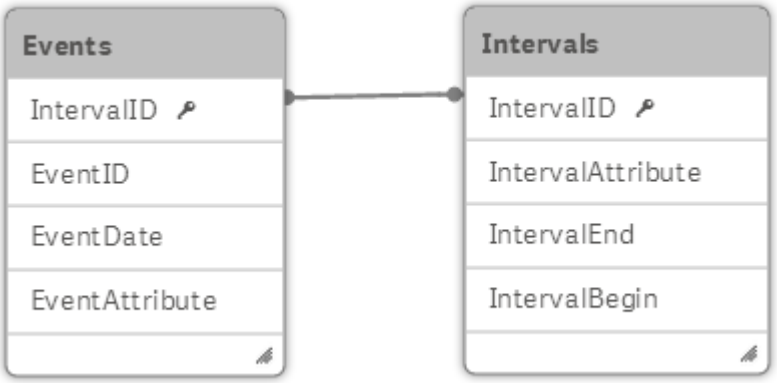
1. 使用下列指令碼取代現有的 Bridgetable 陳述式：
   ```
   BridgeTable:
   LOAD distinct * Where Exists(EventDate);
   LOAD IntervalBegin + IterNo() - 1 as EventDate, IntervalID
       Resident Intervals
       While IntervalBegin + IterNo() - 1 <= IntervalEnd;
   ```
2. 按一下**載入資料**。
3. 開啟**資料模型檢視器**。資料模型應如下所示：
   資料模型：Events、BridgeTable 和 Intervals 表格



   一般而言，採用三個表格的解決方案是最佳方案，因為它允許間隔與事件之間存在多對多關係。但是有一種常見的情況，您知道一個事件只能屬於一個單一間隔。在此情況下，橋接表格並非必要。*IntervalID* 可以直接儲存在事件表格中。要執行此操作，有多種方法，最有用的方法就是將 Bridgetable 聯結到 *Events* 表格中。

4. 將下列指令碼新增至指令碼結尾：
   ```
   Join (Events)
   LOAD EventDate, IntervalID
   Resident BridgeTable;

   Drop Table BridgeTable;
   ```
5. 按一下**載入資料**。
6. 開啟**資料模型檢視器**。資料模型應如下所示：

資料模型：Events 和 Intervals 表格



開放和封閉的間隔

間隔是開放還是封閉由結束點確定，具體要看結束點是否包括在間隔內。

- 如果結束點包括在間隔內，則為封閉間隔：
  [a,b] = {x ∈ ℝ | a ≤ x ≤ b}
- 如果結束點不包括在間隔內，則為開放間隔：
  ]a,b[= {x ∈ ℝ | a < x < b}
- 如果只有一個結束點包括在間隔內，則為半開放間隔：
  [a,b[= {x ∈ ℝ | a ≤ x < b}

如果遇到間隔重疊且一個數字可以同時屬於多個間隔的情況，您通常需要使用封閉間隔。

但在某些情況下，您不想要重疊間隔，您想讓一個數字只屬於一個間隔。因此，如果一個點位於一個間隔的結尾處，同時也位於下一個間隔的開頭，您就會遇到問題。這樣的數值將同屬於兩個間隔。因此，您想要半開放間隔。

此問題的實際解決方法是從所有間隔的結束值中減去少量值，這樣便可以建立封閉但不重疊的間隔。如果您的數值是日期，最簡單的方法就是使用函數 DayEnd()，該函數會返回當日的最後一毫秒：

```
Intervals:
LOAD…, DayEnd(IntervalEnd – 1) as IntervalEnd From Intervals;
```

您也可以手動減去少量值。如果您這麼做了，請確保減去的數量不至於太少，因為運算會四捨五入至 52 個有效二進位數字 (14 個小數位數)。如果您使用的量太小，差別不會很明顯，您將用回原來的數值。

4 資料清理

有時，您載入 Qlik Sense 的來源資料不一定是您想在 Qlik Sense 應用程式中顯示的樣子。Qlik Sense 提供了多種函數和陳述式，以方便您將資料轉換成我們想要的格式。

執行指令碼時，可以在 Qlik Sense 指令碼中使用對應來取代或修改欄位值或名稱，因此對應可用於清理資料，並確保更高的一致性，或者取代部分或全部欄位值。

從不同的表格載入資料時，代表相同事物的欄位值的命名不一定會一致。因為缺乏一致性會阻礙關聯，所以需要解決這個問題。透過建立欄位值比較的對應表格，即可漂亮地解決此問題。

4.1 對應表格

透過 Mapping 載入或 Mapping 選取載入的表格，與其他表格的處理方式有所不同。它們會儲存在記憶體的不同區域，並且僅在指令碼執行時作為對應表格之用。執行指令碼後，這些表格會被自動捨棄。

規則：

- 對應表格必須有兩個資料行：第一個包含比較值，第二個包含所需的對應值。
- 必須對這兩個資料行加以命名，但名稱本身沒有相關性。該資料行名稱和一般內部表格的欄位名稱並無關聯。

4.2 Mapping 函數和陳述式

本教學課程中將說明以下對應函數/陳述式：

•Mapping 前置詞

• ApplyMap()

• MapSubstring()

•Map … Using 陳述式

•Unmap 陳述式

4.3 對應前置詞

Mapping 前置詞用於在指令碼中建立對應表格。對應表格可與 ApplyMap() 函數、MapSubstring() 函數或 Map … Using 陳述式一起使用。

請執行下列動作：

1. 建立新的應用程式並命名。
2. 在**資料載入編輯器**中新增新的指令碼區段。
3. 叫用區段 *Countries*。
4. 輸入以下指令碼：

```
CountryMap:
MAPPING LOAD * INLINE [
Country, NewCountry
U.S.A., US
U.S., US
United States, US
United States of America, US
];
```

CountryMap 表格儲存兩個資料行：*Country* 和 *NewCountry*。*Country* 資料行將輸入的各種國家拼寫方式儲存在 *Country* 欄位中。*NewCountry* 資料行儲存各個值的對應方式。此對應表格將用於在 *Country* 欄位中儲存一致的 *US* 國家值。例如，如果 *U.S.A.* 儲存在 *Country* 欄位中，將其對應為 *US*。

4.4 ApplyMap() 函數

使用 ApplyMap() 根據之前建立的對應表格取代欄位中的資料。在使用 ApplyMap() 函數之前，需要先載入對應表格。您將載入的 *Data.xlsx* 表格中的資料如下所示：

資料表格

ID	名稱	國家/地區	代碼
1	John Black	美國	SDFGBS1DI
2	Steve Johnson	美國	2ABC
3	Mary White	美國	DJY3DFE34
4	Susan McDaniels	u	DEF5556
5	Dean Smith	US	KSD111DKFJ1

請注意國家是以多種方式輸入的。為了確保國家欄位的一致性，已載入對應表格，並使用了 **ApplyMap()** 函數。

請執行下列動作：

1. 在您於上方輸入的指令碼之下，選取並載入 *Data.xlsx*，然後插入指令碼。
2. 在新建立的 LOAD 陳述式之上輸入下列內容：
   ```
   Data:
   ```
   您的指令碼應如下所示：
   ```
   CountryMap:
   MAPPING LOAD * INLINE [
       Country, NewCountry
       U.S.A., US
       U.S., US
       United States, US
       United States of America, US
   ];
   ```

```
Data:
LOAD
    ID,
    Name,
    Country,
    Code
FROM [lib://AttachedFiles/Data.xlsx]
(ooxml, embedded labels, table is Sheet1);
```

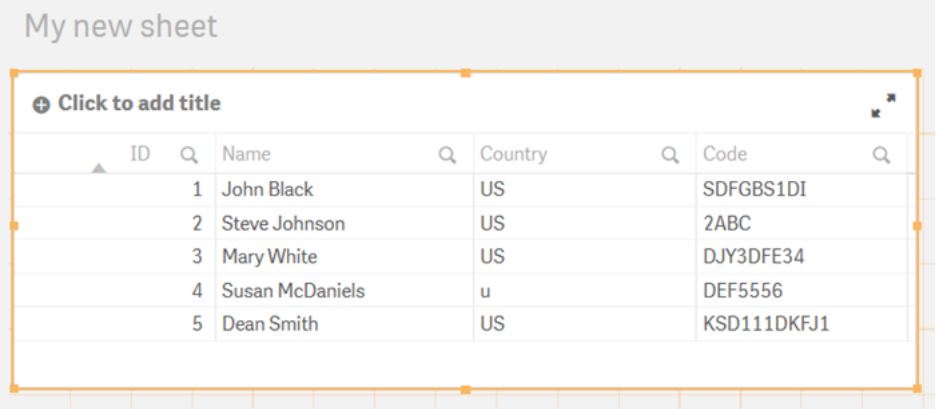
3. 如下所示修改包括 Country, 的行:

```
ApplyMap('CountryMap', Country) as Country,
```

ApplyMap() 函數的首個參數的對應名稱用單引號括住了。第二個參數是其中的資料將被取代的欄位。

4. 按一下**載入資料**。
產生的表格如下所示:

表格中顯示使用 ApplyMap() 函數載入的資料



United States 的各種拼寫已全部變更為 *US*。有一個記錄沒有拼寫正確, 因此 ApplyMap() 函數沒有變更該欄位值。使用 ApplyMap() 函數時, 如果對應表格沒有配對的值, 您可使用第三個參數新增預設運算式。

5. 將 'US' 新增為 ApplyMap() 函數的第三個參數, 以處理國家沒有正確輸入的情況:

```
ApplyMap('CountryMap', Country, 'US') as Country,
```

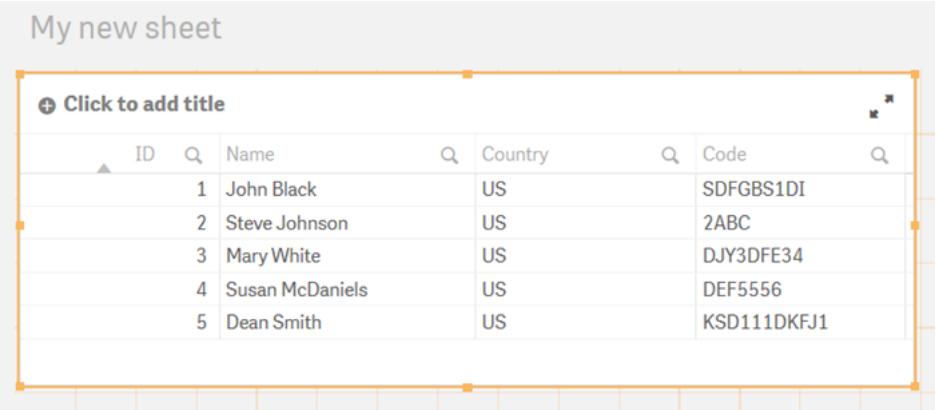
您的指令碼應如下所示:

```
CountryMap:
MAPPING LOAD * INLINE [
    Country, NewCountry
    U.S.A., US
    U.S., US
    United States, US
    United States of America, US
];

Data:
```

```
LOAD
    ID,
    Name,
    ApplyMap('CountryMap', Country, 'US') as Country,
    Code
FROM [lib://AttachedFiles/Data.xlsx]
(ooxml, embedded labels, table is Sheet1);
```

6. 按一下**載入資料**。
   產生的表格如下所示：

   表格中顯示使用 ApplyMap 函數載入的資料



*若要進一步瞭解 ApplyMap()，請在 Qlik Community 中參閱此篇部落格貼文：*請勿聯結 - 改用 Applymap

4.5 MapSubstring() 函數

透過 MapSubstring() 函數，您可以對應部分欄位。

在 ApplyMap() 建立的表格中，我們想要把數字寫成文字格式，所以將使用 MapSubstring() 函數把數字資料替換成文字。

要執行此操作，首先要建立一個對應表格。

請執行下列動作：

1. 在 *CountryMap* 區段之後新增以下指令碼行，但必須加在 *Data* 區段之前。

```
CodeMap:
MAPPING LOAD * INLINE [
F1, F2
1, one
2, two
3, three
4, four
5, five
```

```
11, eleven
];
```

在 *CodeMap* 表格中，數字 1 至 5 以及 11 都有對應。

2. 在指令碼的 *Data* 區段，如下所示修改 Code 陳述式：

```
MapSubString('CodeMap', Code) as Code
```

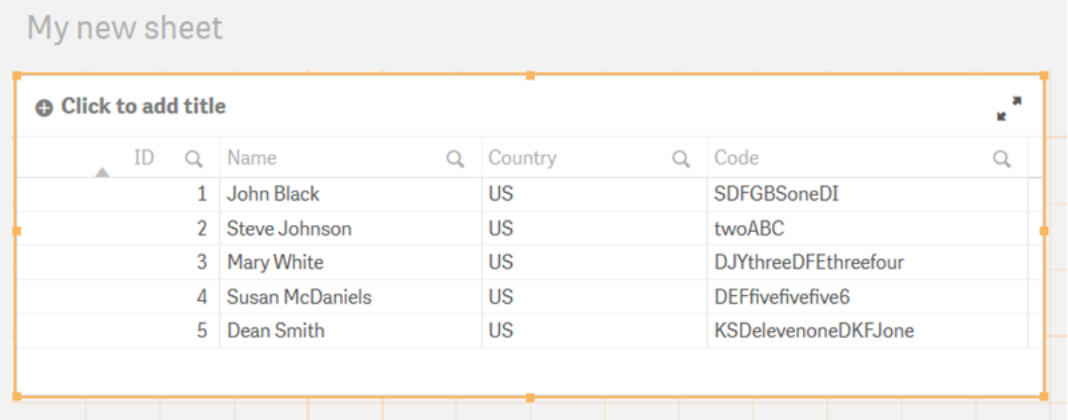
您的指令碼應如下所示：

```
CountryMap:
MAPPING LOAD * INLINE [
    Country, NewCountry
    U.S.A., US
    U.S., US
    United States, US
    United States of America, US
];

CodeMap:
MAPPING LOAD * INLINE [
F1, F2
1, one
2, two
3, three
4, four
5, five
11, eleven
];

Data:
LOAD
    ID,
    Name,
    ApplyMap('CountryMap', Country, 'US') as Country,
    MapSubString('CodeMap', Code) as Code
FROM [lib://AttachedFiles/Data.xlsx]
(ooxml, embedded labels, table is Sheet1);
```

3. 按一下**載入資料**。

產生的表格如下所示：

表格中顯示使用 MapSubString 函數載入的資料



ID	Name	Country	Code
1	John Black	US	SDFGBSoneDI
2	Steve Johnson	US	twoABC
3	Mary White	US	DJYthreeDFEthreefour
4	Susan McDaniels	US	DEFfivefivefive6
5	Dean Smith	US	KSDelevenoneDKFJone

在 *Code* 欄位中，數字字元已被文字取代。如果數字出現多次，如 ID=3 和 ID=4 那樣，則同樣重複文字。ID=4。*Susan McDaniels* 在其程式碼中有 6。因為 6 沒有在 *CodeMap* 表格中對應，因此將保持不變。ID=5、*Dean Smith* 的程式碼中有 111。這已經被對應為 'elevenone'。

*若要進一步瞭解 MapSubstring()，請在 Qlik Community 中參閱此篇部落格貼文：*Mapping … and not the geographical kind (對應 … 不是地理種類)

4.6 Map … Using

也可以使用 Map … Using 陳述式將對應套用至欄位。不過，這與 ApplyMap() 的運作方式有些不同。。ApplyMap() 處理每次欄位名稱遇到的對應問題，而 Map … Using 處理值儲存在內部表格中欄位名稱下遇到的對應問題。

讓我們來看一個例子。假設我們在指令碼中多次載入 *Country* 欄位，並希望每次載入欄位時，都能套用對應。像本課程教學之前說明的那樣，可以使用 ApplyMap() 函數，也可以使用 Map … Using。

如果使用了 Map … Using，當欄位儲存在內部表格中時，對應將套用至欄位。在下面的範例中，對應被套用至 Data1 表格中的 *Country* 欄位，但不會被套用至 *Data2* 表格中的 *Country2* 欄位。這是因為 Map … Using 陳述式只能套用至名為 *Country* 的欄位。當 *Country2* 欄位儲存在內部表格中時，它的名稱不再是 *Country*。如要將對應套用至 *Country2* 表格，您必須使用 ApplyMap() 函數。

如果 *Country* 要在 Unmap 陳述式之後載入，則 Unmap 陳述式將終止 Map … Using 陳述式，系統不會套用 *CountryMap*。

請執行下列動作：

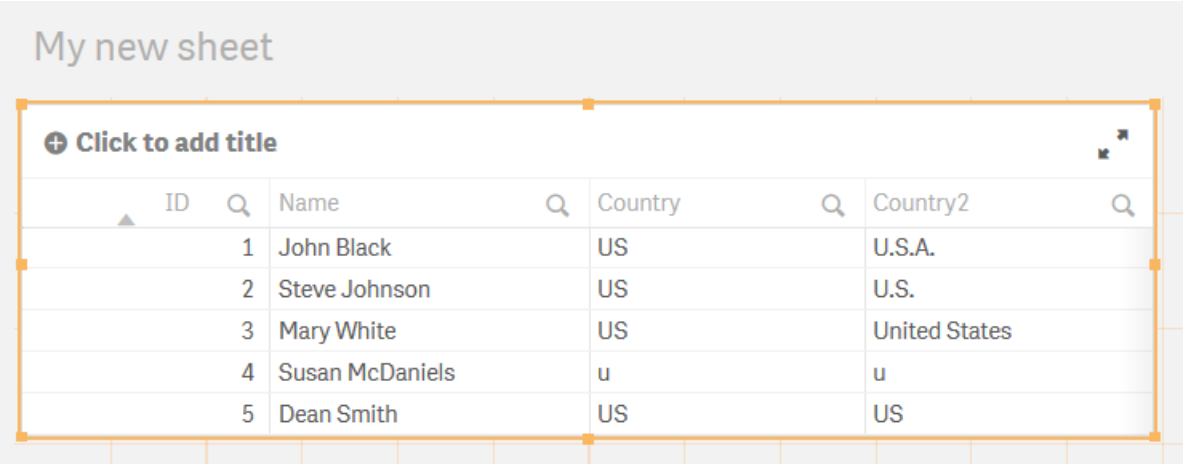
1. 使用下列內容取代 *Data* 表格的指令碼：

```
Map Country Using CountryMap;
Data1:
    LOAD
    ID,
    Name,
```

```
    Country
FROM [lib://AttachedFiles/Data.xlsx]
(ooxml, embedded labels, table is Sheet1);

Data2:
LOAD
    ID,
    Country as Country2
FROM [lib://AttachedFiles/Data.xlsx]
(ooxml, embedded labels, table is Sheet1);
UNMAP;
```

2. 按一下**載入資料**。
   產生的表格如下所示：

   表格中顯示使用 Map ... Using 函數載入的資料



5 處理階層式資料

階層是所有商業智慧解決方案的重要組成部分，用於描述固有包含各種精細層級的維度。一些階層簡單直觀，而另一些則十分複雜，需要大量思考才能正確建模。

從階層的上層到下層，分支越來越具體。例如，在擁有市場、國家、州和城市層級的維度中，分支美洲顯示在階層的頂層，分支美國顯示在第二層，分支加州顯示在第三層，而舊金山位於底層。加州比美國更具體，舊金山又比加州更具體。

在關係模型中儲存階層是多個解決方案所面臨的共同難題。有幾種方法：

- 水平階層
- 鄰接清單模型
- 路徑列舉法
- 巢狀集合模型
- 上階清單

為了實現本教學課程的目的，我們將建立一個上階清單，因為該清單顯示階層的形式可直接在查詢中使用。如需瞭解其他方法的更多資訊，請瀏覽 Qlik Community。

5.1 Hierarchy 前置詞

Hierarchy 前置詞是一個放在 LOAD 或 SELECT 陳述式前面的指令碼命令，可載入相鄰節點表格。LOAD 陳述式至少要有三個欄位：ID (節點的唯一索引鍵)、父節點的參考及名稱。

前置詞會將載入的表格轉換為展開的節點表格；該表格有很多其他的資料行，分別用於階層的各個層級。

請執行下列動作：

1. 建立新的應用程式並命名。
2. 在**資料載入編輯器**中新增新的指令碼區段。
3. 叫用區段 *Wine*。
4. 在右側功能表的 **AttachedFiles** 之下，按一下**選取資料**。
5. 上傳然後選取 *Winedistricts.txt*。
6. 在**從以下項目選取資料**視窗中，取消核取 *Lbound* 和 *RBound* 欄位，就不會載入這些內容。
7. 按一下**插入指令碼**。
8. 在 LOAD 陳述式之上輸入下列內容：

   ```
   Hierarchy (NodeID, ParentID, NodeName)
   ```

   您的指令碼應如下所示：

   ```
   Hierarchy (NodeID, ParentID, NodeName)
   LOAD
       NodeID,
       ParentID,
   ```

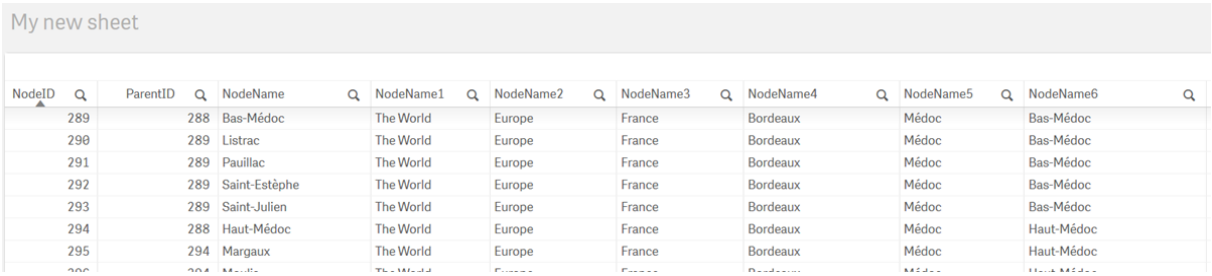
```
    NodeName
FROM [lib://AttachedFiles/Winedistricts.txt]
(txt, utf8, embedded labels, delimiter is '\t', msq);
```

9. 按一下**載入資料**。
10. 使用**資料模型檢視器**的**預覽**區段檢視產生的表格。
    產生的展開節點表格擁有與來源表格一樣的記錄數:每個節點一個。展開的節點表格非常實用,因為它符合在關係模型中分析階層的一些要求:

    - 所有節點名稱存在於一個相同的資料行中,因此可用於搜尋。
    - 此外,不同的節點層級已分別擴展至各個欄位,即那些可用於向下探查群組或作為樞紐分析表中維度的欄位中。
    - 此外,不同的節點層級已分別擴展至各個欄位,即那些可用於向下探查群組的欄位中。
    - 可以將其設為包含節點的唯一路徑,以正確的順序列出所有上階。
    - 也可以將其設為包含節點深度,例如到根的距離。

    產生的表格如下所示:

    表格中顯示使用 Hierarchy 前置詞載入的資料樣本



NodeID	ParentID	NodeName	NodeName1	NodeName2	NodeName3	NodeName4	NodeName5	NodeName6
289	288	Bas-Médoc	The World	Europe	France	Bordeaux	Médoc	Bas-Médoc
290	289	Listrac	The World	Europe	France	Bordeaux	Médoc	Bas-Médoc
291	289	Pauillac	The World	Europe	France	Bordeaux	Médoc	Bas-Médoc
292	289	Saint-Estèphe	The World	Europe	France	Bordeaux	Médoc	Bas-Médoc
293	289	Saint-Julien	The World	Europe	France	Bordeaux	Médoc	Bas-Médoc
294	288	Haut-Médoc	The World	Europe	France	Bordeaux	Médoc	Haut-Médoc
295	294	Margaux	The World	Europe	France	Bordeaux	Médoc	Haut-Médoc

5.2 HierarchyBelongsTo 前置詞

像 Hierarchy 前置詞一樣,HierarchyBelongsTo 前置詞是一個放在 LOAD 或 SELECT 陳述式前面的指令碼命令,可載入相鄰節點表格:

對此,LOAD 陳述式也至少要有三個欄位:ID (節點的唯一索引鍵)、父節點的參考及名稱。前置詞會將載入的表格轉換為上階表格,該表格將所有上階和下階的組合列為單獨的記錄。因此,很容易就能找到特定節點的所有上階或下階。

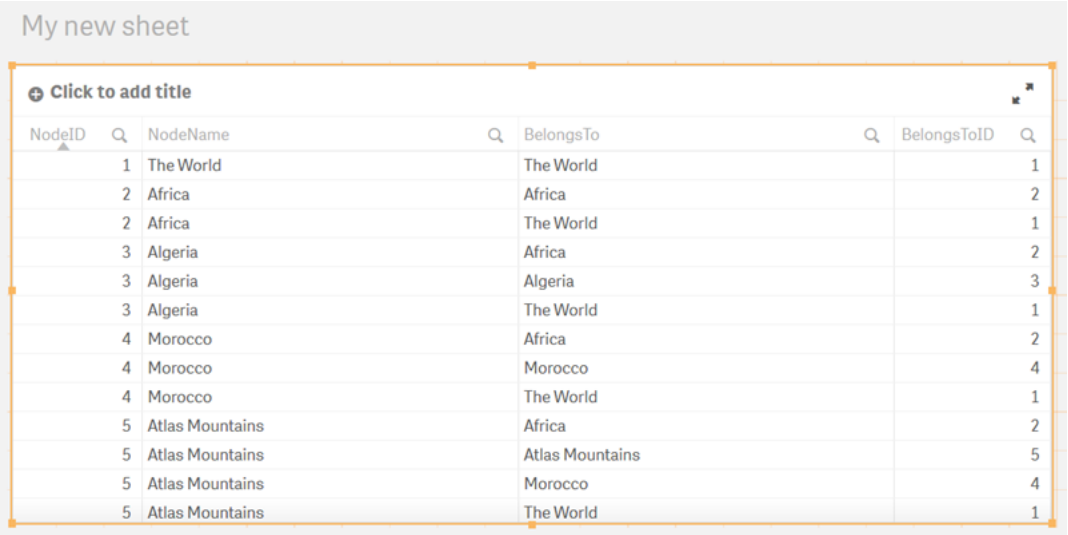
請執行下列動作:

1. 修改**資料載入編輯器**中的 Hierarchy 陳述式,使其成為以下格式:
   `HierarchyBelongsTo (NodeID, ParentID, NodeName, BelongsToID, BelongsTo)`
2. 按一下**載入資料**。
3. 使用**資料模型檢視器**的**預覽**區段檢視產生的表格。
   上階表格符合在關係模型中分析階層的一些要求:

- 如果節點 ID 代表單個節點，上階 ID 則代表階層的整個樹狀和子樹狀結構。
- 所有節點名稱在角色中同時以節點和樹狀結構的形式存在，並且均可用於搜尋。
- 也可以將其設為包含節點深度和上階深度之間的深度差異，即與子樹狀圖的根的距離。

產生的表格如下所示：

表格中顯示使用 HierarchyBelongsTo 前置詞載入的資料



NodeID	NodeName	BelongsTo	BelongsToID
1	The World	The World	1
2	Africa	Africa	2
2	Africa	The World	1
3	Algeria	Africa	2
3	Algeria	Algeria	3
3	Algeria	The World	1
4	Morocco	Africa	2
4	Morocco	Morocco	4
4	Morocco	The World	1
5	Atlas Mountains	Africa	2
5	Atlas Mountains	Atlas Mountains	5
5	Atlas Mountains	Morocco	4
5	Atlas Mountains	The World	1

授權

階層經常用於授權。組織階層就是一個範例。每個管理員有權查看其所在部門的所有內容，包括所有子部門。但他們不一定有權查看其他部門的內容。

組織階層範例

這表示，系統會允許不同的人查看不同的組織子樹狀結構。授權表格可能如下所示：

授權表格

存取	NTNAME	個人	位置	權限
使用者	ACME\JRL	John	產品長	人資
使用者	ACME\CAH	Carol	執行長	執行長
使用者	ACME\JER	James	工程主任	工程
使用者	ACME\DBK	Diana	財務長	財務
使用者	ACME\RNL	Bob	營運長	銷售額
使用者	ACME\LFD	Larry	技術長	產品

在此情況下，*Carol* 可以查看與 *CEO* 和以下有關的所有內容；*Larry* 可以查看 *Product* 組織；*James* 只能查看 *Engineering* 組織。

範例：

通常階層會儲存在相鄰節點表格中。在此範例中，若要解決此，可以使用 HierarchyBelongsTo 載入相鄰節點表格，並命名上階欄位 *Tree*。

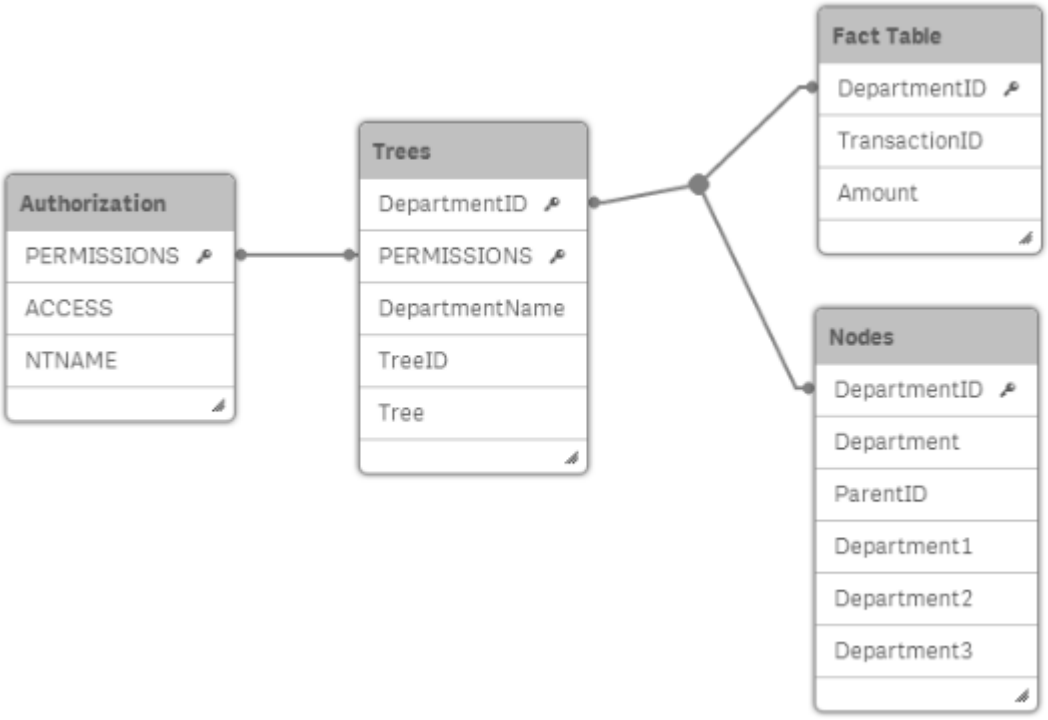
若您想要使用 Section Access，請載入 *Tree* 的大寫複本，並叫用此新欄位 *PERMISSIONS*。最後，您需要載入驗證表格。可以使用下列指令碼行完成這兩個最後步驟。請注意，TempTrees 表格是由 HierarchyBelongsTo 陳述式建立的表格。

請注意，這只是範例。在 Qlik Sense 中沒有要完成的附帶活動。

```
Trees:
LOAD *,
    Upper(Tree) as PERMISSIONS
    Resident TempTrees;
Drop Table TempTrees;

Section Access;
Authorization:
LOAD    ACCESS,
        NTNAME,
        UPPER(Permissions) as PERMISSIONS
From Organization;
Section Application;
```

此範例會產生下列資料模型:

資料模型:Authorization、Trees、Fact 和 Nodes 表格



6 QVD 檔案

QVD (QlikView Data) 檔案是包含從 Qlik Sense 或 QlikView 所匯出之資料表格的檔案。QVD 是原生 Qlik 格式，而且只可寫入 Qlik Sense 或 QlikView 並由其讀取。從 Qlik Sense 指令碼讀取資料時，檔案格式的速度已經過最佳化處理，但是仍然十分精簡。從 QVD 檔案讀取資料通常比從其他資料來源讀取資料快 10-100 倍。

可用兩種模式讀取 QVD 檔案：標準 (快速) 與最佳化 (更快速)。Qlik Sense 指令碼引擎會自動決定選取的模式。雖然允許重新命名欄位，但只有在單純讀取所有載入的欄位，而不轉換 (在欄位上作用的公式) 的情況下，才能使用最佳化模式。造成 Qlik Sense 解壓縮記錄的 Where 子句也會停用最佳化載入。

QVD 檔案剛好只有一個資料表格，且包含三個部分：

- 描述表格中欄位的 XML 標頭 (為 UTF-8 字元集)、後續資訊的版面配置以及部分其他中繼資料。
- 位元組填充格式的符號表格。
- 些許填充格式的實際資料表。

QVD 檔案可用於多種用途。有四個主要用途。超過一種以上的用途適用於任何指定情況：

- 提高資料載入速度

  藉由緩衝 QVD 檔案中輸入資料的不變或緩慢變更的區塊，指令碼在大型資料集的執行可變得相當快速。

- 減少資料庫伺服器上的負載

  另外也會大幅減少從外部資料來源擷取的資料量。這可減少外部資料庫上的工作負載與網路流量。此外，當數個 Qlik Sense 指令碼共用相同的資料時，只需要將資料從來源資料庫載入 QVD 檔案一次即可。其他應用程式可透過此 QVD 檔案利用相同的資料。

- 合併來自多個 Qlik Sense 應用程式的資料。

  可使用 Binary 指令碼陳述式將只來自單一 Qlik Sense 應用程式的資料載入另一個應用程式，但若使用 QVD 檔案，Qlik Sense 指令碼可合併來自任何數目的 Qlik Sense 應用程式的資料。這樣應用程式就可以從不同的業務單位合併相似的資料。

- 累加載入

  在許多常見案例中，可使用 QVD 功能來加速累加載入，亦即獨立從增長中的資料庫載入新記錄。

若要查看 Qlik 社群如何使用 Qlik 應用程式自動化 改善 QVD 載入時間，請參閱 如何使用自動化分割 QVD 以改善載入。

6.1 建立 QVD 檔案

建立 QVD 檔案的方法有兩種：

- 使用 Qlik Sense 指令碼中的 Store 命令可明確建立與命名。
  在指令碼中陳述要將先前讀取的表格或部分匯出到您所選位置中明確命名的檔案。
- 從指令碼自動建立與維護。
  在 LOAD 或 SELECT 陳述式前面加上 Buffer 前置詞，Qlik Sense 會自動建立 QVD 檔案，在某些狀況中可在重新載入資料時使用此 QVD 檔案而不使用原始資料來源。

在讀取速度方面，產生的 QVD 檔案之間不會有任何差異。

Store

此指令碼陳述式可建立明確命名的 QVD、CSV 或 txt 檔案。

語法：

```
Store[ *fieldlist from] table into filename [ format-spec ];
```

陳述式只能從一個資料表格匯出欄位。如果要匯出數個表格中的欄位，則必須在指令碼中提前加入明確的 join 前置詞，以建立應該匯出的資料表格。

文字值會以 UTF-8 格式匯出到 CSV 檔案中。可指定分隔符號，請參閱 **LOAD**。在 CSV 檔案中使用 store 陳述式不支援 BIFF 匯出。

範例：

```
Store mytable into [lib://AttachedFiles/xyz.qvd];
Store * from mytable into [lib://FolderConnection/xyz.qvd];
Store myfield from mytable into 'lib://FolderConnection/xyz.qvd';
Store myfield as renamedfield, myfield2 as renamedfield2 from mytable into
[lib://AttachedFiles/xyz.qvd];
Store mytable into 'lib://FolderConnection/myfile.txt';
Store * from mytable into 'lib://FolderConnection/myfile.csv';
```

請執行下列動作：

1. 開啟*進階指令碼教學課程*應用程式。
2. 按一下 *Product* 指令碼區段。
3. 將下列內容新增至指令碼結尾：

   ```
   Store * from Product into [lib://AttachedFiles/ProductData.qvd](qvd);
   ```

   您的指令碼應如下所示：

   ```
   CrossTable(Month, Sales)
   LOAD
       Product,
       "Jan 2014",
       "Feb 2014",
       "Mar 2014",
       "Apr 2014",
       "May 2014"
   ```

```
FROM [lib://AttachedFiles/Product.xlsx]
(ooxml, embedded labels, table is Product);

Store * from Product into [lib://AttachedFiles/ProductData.qvd](qvd);
```

4. 按一下**載入資料**。
   Product.qvd 檔案現在應於檔案清單中。

   此資料檔案是 **Crosstable** 指令碼產生的擁有三個資料行的表格，每個資料行代表一個類別 (Product, Month, Sales)。此資料檔案現可用於取代整個 *Product* 指令碼區段。

6.2 從 QVD 檔案讀取資料

可使用下列方法透過 Qlik Sense 讀取或存取 QVD 檔案：

- 將 QVD 檔案載入為明確的資料來源。QVD 檔案可由 Qlik Sense 指令碼中的 LOAD 陳述式參考，如任何其他文字檔案類型一樣 (csv、fix、dif、biff 等)。

  範例：

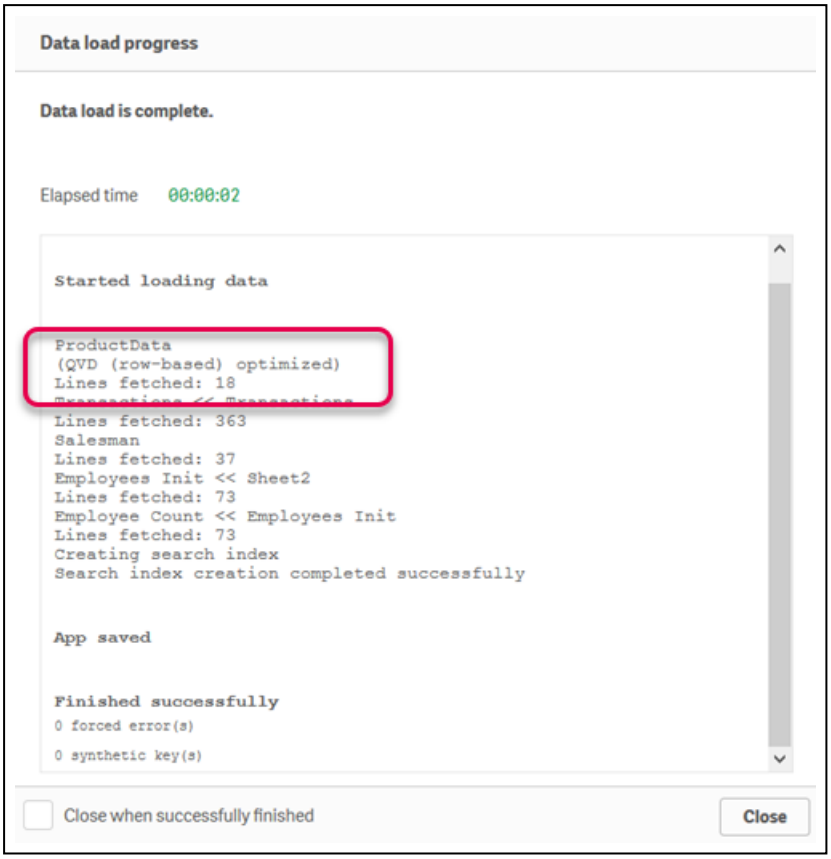
  ```
  LOAD * from 'lib://FolderConnection/xyz.qvd' (qvd);
  LOAD fieldname1, fieldname2 from [lib://FolderConnection/xyz.qvd] (qvd);
  LOAD fieldname1 as newfieldname1, fieldname2 as newfieldname2 from
  [lib://AttachedFiles/xyz.qvd](qvd);
  ```

- 自動載入緩衝的 QVD 檔案。在 load 或 select 陳述式使用 buffer 前置詞時，不需要用於讀取的明確陳述式。Qlik Sense 會判斷其從 QVD 檔案使用資料的範圍，而不是透過原始 LOAD 或 SELECT 陳述式來取得資料。
- 透過指令碼存取 QVD 檔案。可使用數個指令碼函數 (皆以 QVD 開頭) 來擷取 QVD 檔案 XML 標頭上所找到資料的各種資訊。

請執行下列動作：

1. 註解 *Product* 指令碼區段中的整個指令碼。
2. 輸入以下指令碼：
   ```
   Load * from [lib://AttachedFiles/ProductData.qvd](qvd);
   ```
3. 按一下**載入資料**。
   資料從 QVD 檔案載入。

資料載入進度視窗



*若要瞭解關於使用 QVD 檔案進行累加載入的資訊，請參閱 Qlik Community 中的此篇部落格貼文：*Qlik 累加載入概述

Buffer

透過 Buffer 前置詞可以自動建立和維護 QVD 檔案。這個前置詞可以用於指令碼的大多數 LOAD 和 SELECT 陳述式。其表示 QVD 檔案會用於快取/緩衝陳述式的結果。

語法：

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

如果不使用任何選項，則指令碼第一次執行所建立的 QVD 緩衝區就無使用期限。

範例：

```
Buffer load * from MyTable;
```

stale [after] amount [(days | hours)]

Amount 是指定時間週期的數字。可能會使用 Decimals。如果未指定單位，一律假設單位為天數。

stale after 選項一般會用於其中原始資料沒有簡單時間戳記的資料庫來源。stale after 子句只是陳述從建立 QVD 緩衝區之後經過多久才會失效的時間週期。在這段時間之前，將使用 QVD 緩衝區做為資料的來源，過了這段時間之後，則將使用原始資料來源。此後 QVD 緩衝區檔案將自動更新，並且重啟新的週期。

範例：

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

Incremental

incremental 選項能夠唯讀基礎檔案的一部分。先前的檔案大小儲存於 QVD 檔案的 XML 標頭中。這特別適用於記錄檔。先前載入的所有記錄都是從 QVD 檔案讀取的，而後續的新記錄則是從原始來源讀取的，最後會建立更新的 QVD 檔案。

請注意，incremental 選項只能用於 LOAD 陳述式和文字檔，只要變更或刪除舊檔案，就無法使用累加載入。

範例：

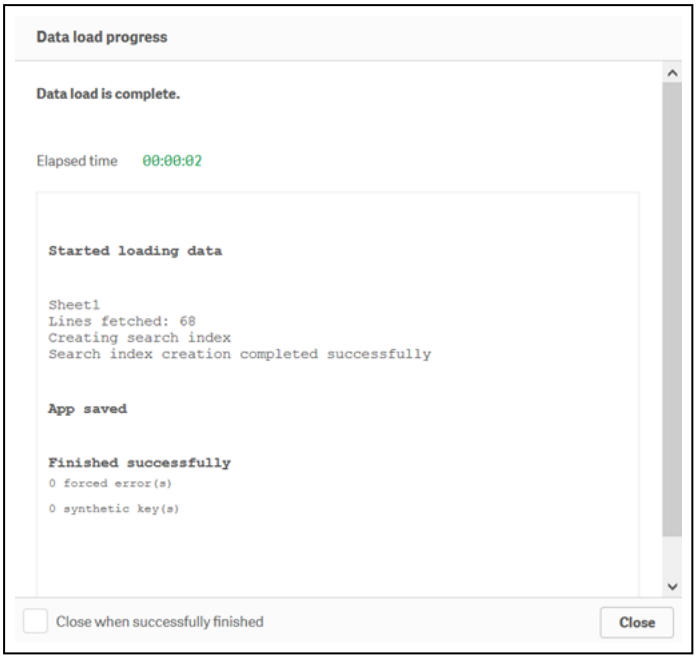
```
Buffer (incremental) load * from MyLog.log;
```

建立 QVD 緩衝區的應用程式之中的完整指令碼執行完全不再參考該緩衝區時，或者建立該緩衝區的應用程式不再存在時，一般都會移除該緩衝區。如果您想將緩衝區的內容保留為 QVD 或 CSV 檔案，應使用 Store 陳述式。

請執行下列動作：

1. 建立新的應用程式並命名。
2. 在**資料載入編輯器**中新增新的指令碼區段。
3. 在右側功能表的 **AttachedFiles** 之下，按一下**選取資料**。
4. 上傳然後選取 *Cutlery.xlsx*。
5. 在**從以下項目選取資料**視窗中，按一下**插入指令碼**。
6. 註解載入陳述式中的欄位，並將載入陳述式變更為下列內容：
   ```
   Buffer LOAD *
   ```
   您的指令碼應如下所示：
   ```
   Buffer LOAD *
          //     "date",
          //     item,
          //     quantity
          FROM [lib://AttachedFiles/Cutlery.xlsx]
          (ooxml, embedded labels, table is Sheet1);
   ```
7. 按一下**載入資料**。
   首次載入資料時，會從 *Cutlery.xlsx* 載入。

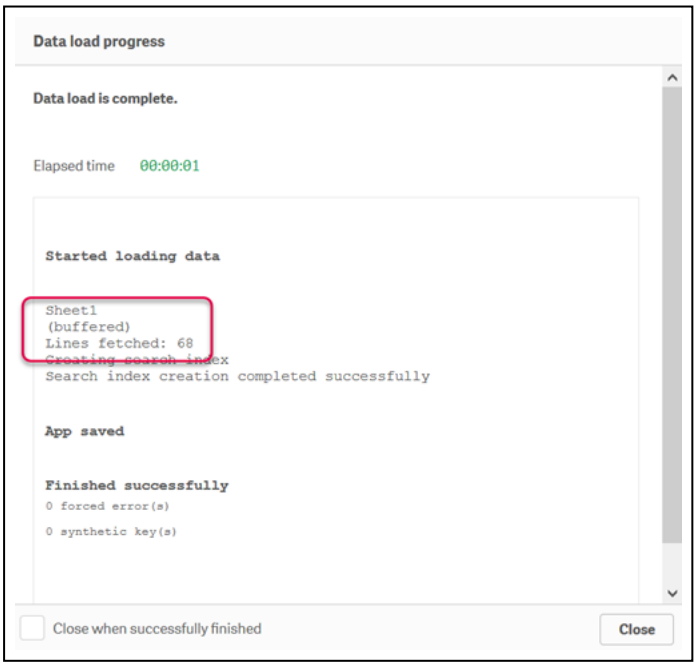
資料載入進度視窗



Buffer 陳述式也會建立 QVD 檔案並儲存在 Qlik Sense。在 Qlik Sense Enterprise on Windows 部署中，這儲存在 Qlik Sense 伺服器上的目錄。

8. 再次按一下**載入資料**。
9. 這次會從首次載入資料時透過 Buffer 陳述式建立的 QVD 檔案載入資料。

資料載入進度視窗



6.3 謝謝您!

現在,您已完成此教學課程,希望您已獲得有關 Qlik Sense 中指令碼的更多知識。請瀏覽我們的網站,瞭解更多培訓詳情。