Eğitim - Kodlamanın Sonraki Adımları

Qlik Sense®
May 2024

Contents

1 Bu derse hoş geldiniz!

Qlik Sense uygulamasında ileri düzey kodlamayı anlatan bu eğitime hoş geldiniz.

Kodlamanın temellerini öğrendikten sonra, Qlik Sense ortamına veri yüklerken daha karmaşık işlemler yapmaya başlayabilirsiniz. Bu örneğin, çapraz tablo kullanarak verileri dönüştürme, veri temizleme ve QVD dosyaları olarak bilinen Qlik veri dosyalarından veri oluşturma ve yükleme işlemlerini içerir.

1.1 Ne öğreneceksiniz?

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

1.2 Kimler bu eğitime katılmalı?

Qlik Sense ortamındaki kodlamanın temellerine aşina olduğunuzu düşünüyoruz. Tabii bu, kodları kullanarak veri yüklediyseniz ve verileri değiştirdiyseniz geçerlidir.

Henüz yapmadıysanız yeni başlayanlar için Kodlama eğitimini tamamlamanızı öneririz.

Veri yükleme düzenleyicisine erişebiliyor olmanız gerekir ve Qlik Sense Enterprise on Windows ortamında veri yükleme izniniz olmalıdır.

Talimatlar genel olarak Qlik Sense Cloud Business için de geçerlidir.

1.3 Paket içerikleri

İndirdiğiniz sıkıştırılmış paket, dersi tamamlamanız için gerekli olan aşağıdaki veri dosyalarını içerir:

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

Bu paket aynı zamanda *Gelişmiş Kodlama Öğreticisi* uygulamasının da bir kopyasını içerir. Uygulamadaki ek kodlama bölümleri bu öğreticide oluşturduğunuz diğer uygulamaların kodlarını içerir. Uygulamayı hub'ınıza yükleyebilirsiniz.

Öğreniminizi en üst seviyeye taşımak için öğreticide anlatılan uygulamayı kendiniz oluşturmanız önerilir. Ek olarak, veri yüklerinin çalışması için veri dosyalarınızı öğreticide anlatıldığı gibi yüklemeniz ve bağlamanız gereklidir.

Fakat sorunla karşılaşırsanız uygulama size sorun giderme konusunda yardımcı olabilir. Hangi kod parçasının hangi dersle ilgili olduğunu belirttik.

1.4 Bu öğreticideki dersler

Bu öğretici, Qlik Sense deneyiminize bağlı olarak 3-4 saat sürecektir. Konular, sırayla tamamlanmak üzere tasarlanmıştır. Fakat istediğiniz zaman ara verip tekrar dönebilirsiniz. İyi bir haber olarak da eğitimde herhangi bir sınav bulunmadığını belirtmek isteriz.

Verileri dönüştürme

Crosstable önekini kullanma

Join ve Keep ile tabloları birleştirme

Kayıtlar arası fonksiyonları kullanma: Peek, Previous ve Exists

Aralıkları ve yinelemeli yüklemeyi eşleştirme

Veri temizleme

Hiyerarşik verileri işleme

QVD dosyaları

1.5 Daha fazla bilgi ve kaynaklar

- Qlik, daha fazla bilgi almak isterseniz çeşitli kaynaklar sağlar.
- Qlik çevrimiçi yardımı mevcuttur.
- Ücretsiz çevrimiçi dersler içeren eğitime, Qlik Continuous Classroom bölümünden erişilebilir.
- Tartışma forumları, bloglar ve daha fazlasını, Qlik Community bölümünde bulabilirsiniz.

2 LOAD ve SELECT deyimleri

LOAD ve SELECT deyimlerini kullanarak Qlik Sense uygulamasına veri yükleyebilirsiniz. Bu deyimlerin her biri dahili tablo oluşturur. Dosyalardan veri yüklemek için LOAD, veritabanlarından veri yüklemek için ise SELECT kullanılır.

Bu eğitimde, dosyalardan veriler kullanacağımız için LOAD deyimlerini kullanacağız.

Ayrıca, öncelikli LOAD deyimini kullanarak yüklenen verilerin içeriğini değiştirebilirsiniz. Örneğin, alanların yeniden adlandırılması LOAD deyiminde yapılmalıdır; SELECT deyimi alan adlarında herhangi bir değişiklik yapılmasına izin vermez.

Qlik Sense içine veri yüklerken aşağıdaki kurallar geçerlidir:

- Qlik Sense, bir LOAD veya SELECT deyimiyle oluşturulan tablolar arasında herhangi bir fark oluşturmaz. Bu, birkaç tablonun yüklenmesi durumunda, tabloların LOAD veya SELECT ifadeleriyle veya ikisinin bir birleşimiyle yüklenmiş olmasının fark yaratmayacağı anlamına gelir.
- Deyimdeki veya veritabanında bulunan ilk tablodaki alanların sıralaması, Qlik Sense mantığına göre önemsizdir.
- Alan adları büyük-küçük harf duyarlıdır ve veri tabloları arasında ilişkilendirme kurmak için kullanılır. Bu sebeple istenen veri modelini elde etmek için komut dosyasında alanların yeniden adlandırılması gereklidir.

3 Verileri dönüştürme

Uygulamanızdaki verileri kullanmadan önce Veri yükleme düzenleyicisindeki verileri dönüştürebilir ve değiştirebilirsiniz.

Veri yönetiminin avantajlarından biri de veri işlemeyi daha verimli kılmak için bir tablodan seçilen birkaç sütun gibi bir dosyadaki verilerin yalnızca bir alt kümesini yüklemeyi seçebilmenizdir. Ham verileri birkaç yeni mantıksal tabloya bölmek için verileri bir kereden fazla da yükleyebilirsiniz. Qlik Sense uygulamasında birden fazla kaynaktan veriler yükleyebilir ve bunları tek bir tabloda birleştirebilirsiniz.

Aşağıdaki işlemlerde Crosstable öneki kullanarak nasıl veri yükleyebileceğinizi göreceksiniz. Tabloları birleştirmeyi, Peek ve Previous gibi kayıtlar arası fonksiyonları kullanmayı ve While Load öğesini kullanarak aynı satırı birkaç kez yüklemeyi de öğreneceksiniz.

3.1 Crosstable önekini kullanma

Çapraz tablolar, üst bilgi verilerinden oluşan iki dikey liste arasında bir değer matrisi içeren genel bir tablo türüdür. Verilerin çapraz tablosuna sahip olduğunuzda, verileri dönüştürmek ve istenen alanları oluşturmak için Crosstable önekini kullanılabilirsiniz.

Crosstable öneki

Aşağıdaki *Product* tablosunda, ay başına bir sütununuz ve ürün başına bir satırınız vardır.

Ürün tablosu

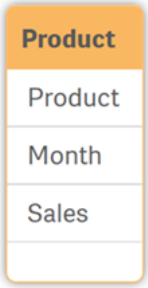
Ürün	Jan 2014	Feb 2014	Mar 2014	Apr 2014	May 2014	Jun 2014
A	100	98	100	83	103	82
B	284	279	297	305	294	292
C	50	53	50	54	49	51

Tabloyu yüklediğinizde sonuç olarak *Product* için bir alan ve her ay için birer alan elde edilir.

Product alanına ve her ay için birer alana sahip Product tablosu

Bu verileri analiz etmek istiyorsanız tüm rakamların bir alanda ve tüm ayların başka bir alanda olması işleri kolaylaştırır. Bu durumda, her kategori için (*Product, Month, Sales*) bir sütuna sahip olan üç sütunlu bir tablodur.

Product, Month ve Sales alanlarına sahip Product tablosu



Crosstable öneki, verileri *Month* için bir sütun ve *Sales* için bir sütunun bulunduğu bir tabloya dönüştürür. Bunu ifade etmenin diğer yolu, alan adlarını alıp bunları alan değerlerine dönüştürdüğüdür.

Aşağıdakileri yapın:

1. Yeni bir uygulama oluşturun ve uygulamaya *Gelişmiş Kodlama Öğreticisi* adını verin.
2. **Veri yükleme düzenleyicisi**'ne yeni bir kod bölümü ekleyin.
3. Bölüme *Product* adını verin.
4. Sağ menüdeki **AttachedFiles** altında **Veri seç**'e tıklayın.
5. Karşıya yükleyin ve ardından *Product.xlsx* öğesini seçin.
6. **Veri seçme kaynağı:** penceresinde *Product* tablosunu seçin.

   > ***Alan adları*** *altında, veri yüklerken tablo alanlarının adlarını dahil etmek için* ***Eklenmiş alan adları****'nın seçildiğinden emin olun.*

7. **Kod ekle**'ye tıklayın.

Kodunuz şöyle görünmelidir:

```
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. **Verileri yükle**'ye tıklayın.
9. **Veri modeli görüntüleyicisi**'ni açın. Veri modeli artık şöyle görünür:

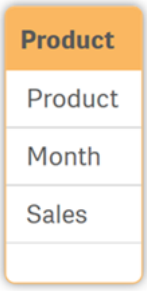
   Product alanına ve her ay için birer alana sahip Product tablosu

10. **Veri yükleme düzenleyicisi**'nde, *Product* sekmesine tıklayın.
11. LOAD deyiminin üzerine şunu girin:

    ```
    CrossTable(Month, Sales)
    ```

12. **Verileri yükle**'ye tıklayın.
13. **Veri modeli görüntüleyicisi**'ni açın. Veri modeli artık şöyle görünür:

    Product, Month ve Sales alanlarına sahip Product tablosu

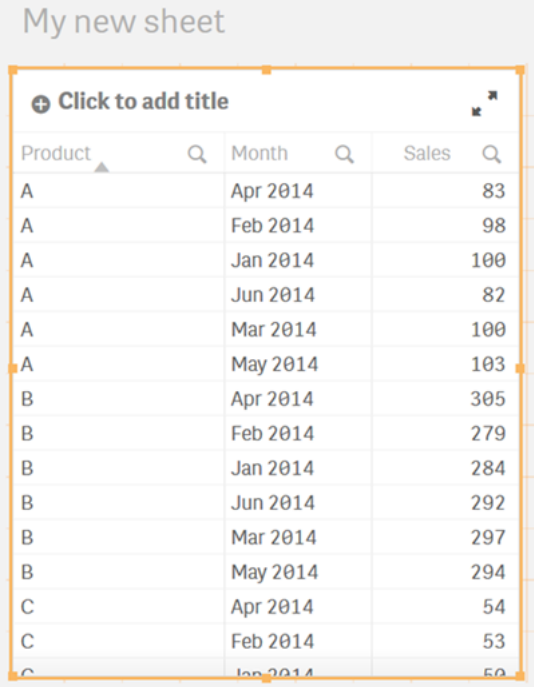


Giriş verilerinin, genellikle dahili anahtar (bu örnekte *Product*) olarak niteleyici alan olan yalnızca bir sütunu olduğunu unutmayın. Ancak sizin birkaç tane olabilir. Bu durumda, tüm niteleyici alanlar LOAD deyiminde öznitelik alanlarından önce listelenmelidir ve niteleyici

alanların sayısını tanımlamak için Crosstable önekinin üçüncü parametresi kullanılmalıdır. Crosstable anahtar sözcüğünden önce LOAD deyimi veya bir örnek koyamazsınız. Fakat otomatik birleştirme kullanabilirsiniz.

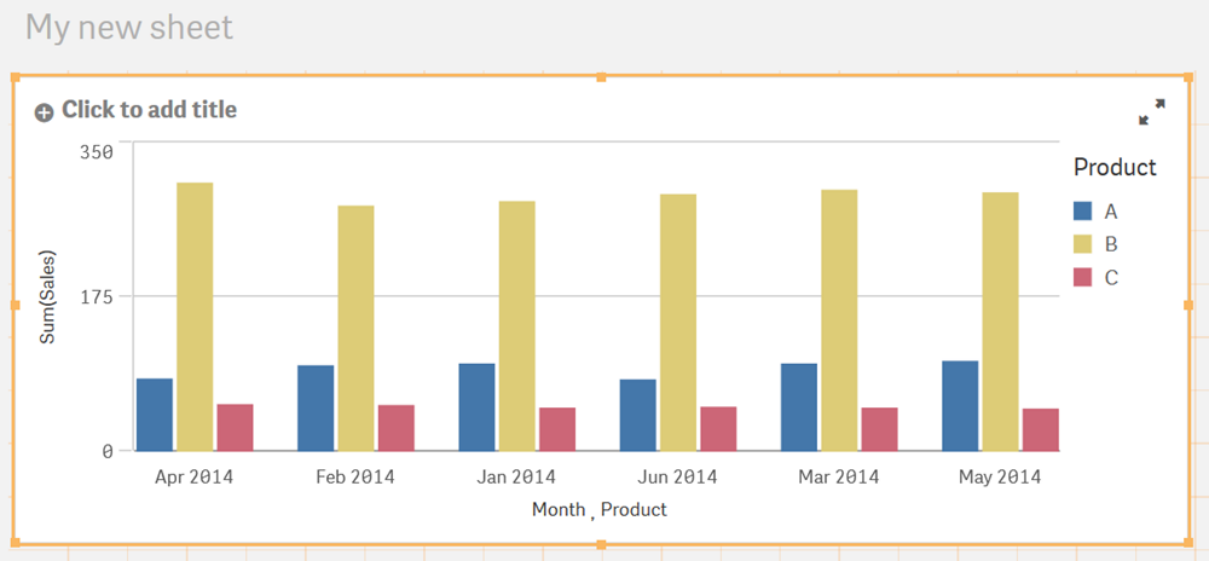
Qlik Sense tablosunda verileriniz şu şekilde görünür:

Crosstable öneki kullanılarak yüklenen veriyi gösteren tablo



Örneğin şimdi de verileri kullanarak bir sütun grafik oluşturun:

Crosstable öneki kullanılarak yüklenen veriyi gösteren sütun grafik



Crosstable hakkında daha fazla bilgi almak için Qlik Community bölümündeki blog gönderisine göz atın: The Crosstable Load (Crosstable Yükleme)*. Davranışlar QlikView bağlamında açıklanmıştır. Fakat kullanılan mantık, Qlik Sense için de geçerlidir.*

Sayısal yorumlama, öznitelik alanları için kullanılmayacaktır. Bu, sütun üst bilgileri olarak elinizde ayarlar olduğunda, bunların otomatik olarak yorumlanmayacağı anlamına gelir. Bunun çözümü, geçici bir tablo oluşturmak için Crosstable öneki kullanmak ve yorumlamaları aşağıdaki örnekte gösterildiği gibi yapmak için ikinci bir geçiş gerçekleştirmektir:

Bunun yalnızca bir örnek olduğunu unutmayın. Qlik Sense içinde tamamlanması gereken başka bir alıştırma yoktur.

```
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;
```

Önbelleği temizleme

Önbelleği temizlemek için oluşturduğunuz tabloları silebilirsiniz. Önceki bölümde olduğu gibi geçici bir yükleme yaparsanız, artık ihtiyaç kalmadığında bunu bırakmanız gerekir. Örneğin:

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

Ayrıca alanları da bırakabilirsiniz. Örneğin:

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

Görebileceğiniz üzere, TABLE ve FIELD anahtar sözcükleri tekil veya çoğul olabilir.

3.2 Join ve Keep ile tabloları birleştirme

Join, iki tabloyu kullanan ve bunları bir tabloda birleştiren bir işlemidir. Sonuçta ortaya çıkan tablonun kayıtları, orijinal tablolardaki kayıtların birleşimleridir; bu genellikle sonuçta ortaya çıkan tablodaki herhangi bir birleşimin bir veya birkaç ortak alan için ortak bir değere sahip olması şeklinde, yani bir natural join ile, gerçekleştirilir. Qlik Sense uygulamasında, join işlemleri kod içinde gerçekleştirilebilir ve mantıksal tablolar oluşturur.

Zaten kodda bulunan tabloları birleştirmek mümkündür. Bu durumda Qlik Sense mantığı ayrı tabloları görmek yerine, tek bir dahili tablo olan birleştirme sonucunu görür. Bazı durumlarda bu gereklidir; ancak bazı dezavantajları da mevcuttur:

- Yüklü tablolar genellikle daha büyük hale gelir ve Qlik Sense daha yavaş çalışır.
- Bazı bilgiler kaybedilebilir: ilk tablodaki sıklık (kayıtların sayısı) artık kullanılamayabilir.

Tablolar Qlik Sense uygulamasında depolanmadan önce iki tablodan birini veya her ikisini tablo verilerinin kesişimine azaltma etkisine sahip olan Keep fonksiyonu, açık birleştirmelerin kullanılması gereken durumların sayısını azaltmak üzere tasarlanmıştır.

Bu belgede, birleştirme terimi genellikle dahili tablolar oluşturulmadan önce gerçekleştirilen birleştirmeler için kullanılmaktadır. Ancak, dahili tablolar oluşturulduktan sonra gerçekleştirilen ilişkilendirme temelde bir birleştirmedir.

Join

Bir birleştirme gerçekleştirmenin en basit yolu, dahili tabloyu başka bir adlandırılmış tabloyla veya önceden oluşturulan son tabloyla birleştiren Join önekini kod içinde kullanmaktır. Birleştirme, iki tabloya ait değerlerin tüm olası bileşimlerini oluşturan bir dış birleştirme olur.

Örnek:

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

Sonuçta ortaya çıkan dahili tablo a, b, c ve d alanlarına sahiptir. Kayıtların sayısı, iki tablonun alan değerlerine bağlı olarak değişiklik gösterir.

Üzerinde birleştirme gerçekleştirilecek alanların adları tam olarak aynı olmalıdır. Üzerinde birleştirme gerçekleştirilecek alanların sayısı rastgeledir. Genellikle tabloların bir veya daha az ortak alanı olmalıdır. Hiçbir ortak alan olmaması, tabloların kartezyen ürününü oluşturur. Tüm alanların ortak olması da mümkündür, ancak genellikle bir anlam ifade etmez. Join deyiminde önceden yüklenmiş bir tablonun tablo adı belirtilmedikçe, Join öneki önceden oluşturulan son tabloyu kullanır. Bu nedenle, iki deyimin sıralaması rastgele değildir.

Join kullanma

Qlik Sense kod dilinde açık Join öneki, iki tablonun tam birleştirmesini gerçekleştirir. Sonuç bir tablodur. Bu tarz birleştirmeler çok büyük tabloların ortaya çıkmasına neden olabilir.

Aşağıdakileri yapın:

1. *Gelişmiş Kodlama Eğitimi* uygulamasını açın.
2. **Veri yükleme düzenleyicisi**'ne yeni bir kod bölümü ekleyin.

3. *Transactions* bölümünü çağırın.
4. Sağ menüdeki **AttachedFiles** altında **Veri seç**'e tıklayın.
5. Karşıya yükleyin ve ardından *Transactions.csv* öğesini seçin.

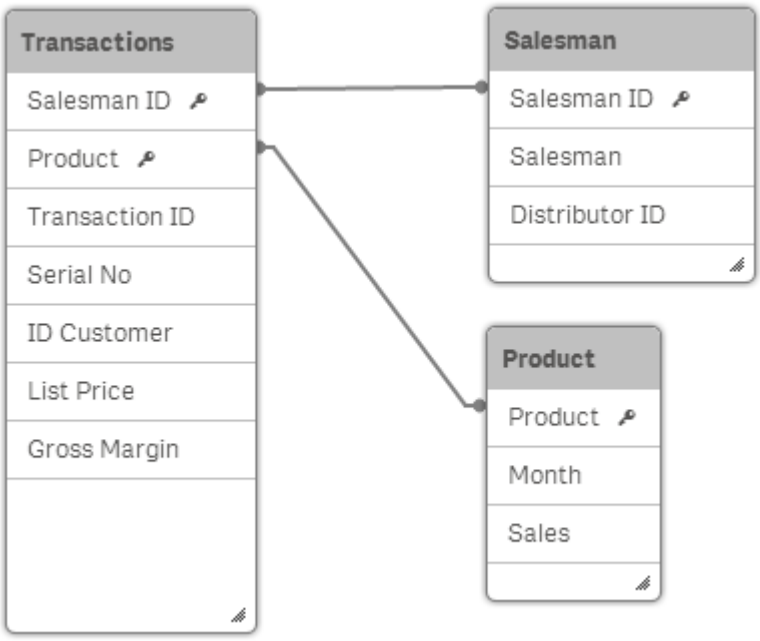
   ***Alan adları** altında, veri yüklerken tablo alanlarının adlarını dahil etmek için **Eklenmiş alan adları**'nın seçildiğinden emin olun.*

6. **Veri seçme kaynağı:** penceresinde **Kod yerleştir**'e tıklayın.
7. Karşıya yükleyin ve ardından *Salesman.xlsx* öğesini seçin.
8. **Veri seçme kaynağı:** penceresinde **Kod yerleştir**'e tıklayın.
   Kodunuz şöyle görünmelidir:

```
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. **Verileri yükle**'ye tıklayın.
10. **Veri modeli görüntüleyicisi**'ni açın. Veri modeli artık şöyle görünür:

Veri modeli: Transactions, Salesman ve Product tabloları



Ancak *Transactions* ve *Salesman* tablolarının ayrılması istenen sonucu vermeyebilir. İki tablonun birleştirilmesi daha iyi olabilir.

Aşağıdakileri yapın:

1. Birleştirilmiş tabloya bir ad vermek için ilk LOAD deyiminin üzerine şu satırı ekleyin:
   ```
   Transactions:
   ```
2. *Transactions* ve *Salesman* tablolarını birleştirmek için ikinci LOAD deyiminin üzerine şu satırı ekleyin:
   ```
   Join(Transactions)
   ```

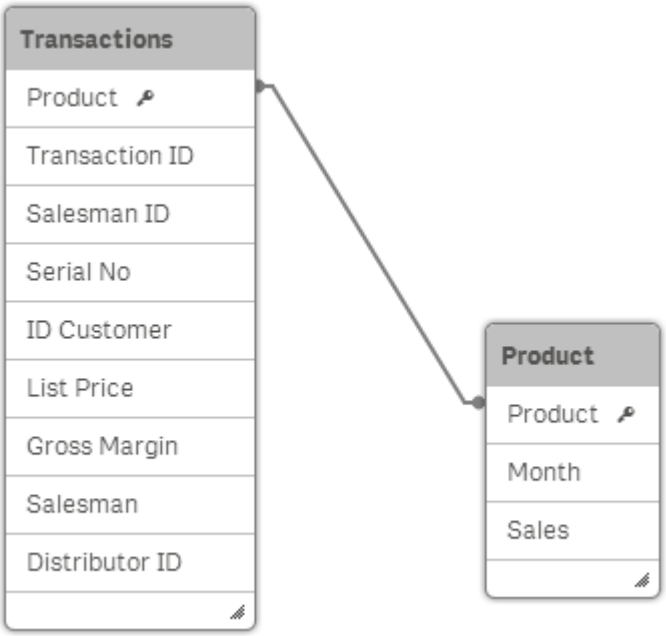
   Kodunuz şöyle görünmelidir:

   ```
   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. **Verileri yükle**'ye tıklayın.
4. **Veri modeli görüntüleyicisi**'ni açın. Veri modeli artık şöyle görünür:

   Veri modeli: Transactions ve Product tabloları



Transactions ve *Salesman* tablolarının tüm alanları artık tek bir *Transactions* tablosunda birleştirilmiştir.

Join fonksiyonunun ne zaman kullanılacağı hakkında daha fazla bilgi almak için Qlik Community bölümündeki blog gönderilerine göz atın: To Join or not to Join (Birleştirmek ya da Birleştirmemek), Mapping as an Alternative to Joining (Birleştirmeye Alternatif olarak Eşleme)*. Davranışlar QlikView bağlamında açıklanmıştır. Fakat kullanılan mantık, Qlik Sense için de geçerlidir.*

Keep

Qlik Sense uygulamasının ana özelliklerinden biri, tabloları birleştirmek yerine tablolar arasında ilişkilendirmeler gerçekleştirmektir; bu da bellekteki alanı azaltır, hızı artırır ve büyük ölçüde esneklik sağlar. Keep fonksiyonu, açık birleştirmelerin kullanılması gereken durumların sayısını azaltmak üzere tasarlanmıştır.

İki LOAD veya SELECT deyimi arasındaki Keep öneki, tablolar Qlik Sense ortamında depolanmadan önce iki tablodan birini veya her ikisini tablo verilerinin kesişimine azaltır. Keep önekinin öncesinde her zaman Inner, Left veyaRight anahtar sözcüklerinden biri gelmelidir. Tablolardan kayıt seçimi, karşılık gelen birleştirme ile aynı şekilde gerçekleştirilir. Ancak, iki tablo birleştirilmez ve Qlik Sense içinde iki ayrı ayrı adlandırılmış tablo olarak depolanır.

Inner

Veri komut dosyasındaki Join ve Keep öneklerinden önce Inner öneki gelebilir.

Join öğesinden önce kullanılması durumunda, iki tablo arasındaki birleştirmenin iç birleştirme olması gerektiğini belirtir. Sonuç olarak elde edilen tablo yalnızca her iki tarafa ait tam veri kümesiyle iki tablo arasındaki bileşimleri içerir.

Keep öğesinden önce kullanılması durumunda, iki tablonun Qlik Sense içinde depolanmadan önce ortak kesişimlerine azaltılması gerektiğini belirtir.

Örnek:

Bu örneklerde *Table1* ve *Table2* kaynak tablolarını kullanıyoruz.

Bunların yalnızca örnek olduğunu unutmayın. Qlik Sense içinde tamamlanması gereken başka bir alıştırma yoktur.

Table 1

A	B
1	aa
2	cc
3	ee

Table2

A	C
1	xx
4	yy

Inner Join

İlk olarak tablolar üzerinde Inner Join yapıyoruz ve her iki tablodan birleştirilen verilerle yalnızca bir satır (her iki tabloda da bulunan tek kayıt) içeren *VTable* ortaya çıkıyor.

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

VTable

A	B	C
1	aa	xx

Inner Keep

Bunun yerine Inner Keep uygularsanız hala iki tablonuz olur. İki tablo, *A* ortak alanı aracılığıyla ilişkilendirilir.

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

VTab1

A	B
1	aa

VTab2

A	C
1	xx

Left

Veri kod dosyasındaki Join ve Keep öneklerinden önce left öneki gelebilir.

Join öğesinden önce kullanılması durumunda, iki tablo arasındaki birleştirmenin sol birleştirme olması gerektiğini belirtir. Sonuç olarak elde edilen tablo yalnızca birinci tabloya ait tam veri kümesiyle iki tablo arasındaki bileşimleri içerir.

Keep öğesinden önce kullanılması durumunda, ikinci tablonun Qlik Sense içinde depolanmadan önce birinci tabloyla ortak kesişimine azaltılması gerektiğini belirtir.

Örnek:

Bu örneklerde *Table1* ve *Table2* kaynak tablolarını kullanıyoruz.

Table1

A	B
1	aa
2	cc
3	ee

Table2

A	C
1	xx
4	yy

İlk olarak tablolar üzerinde bir Left Join yapıyoruz ve *Table1*'den gelen tüm satırları, *Table2* içindeki eşleşen satırlardan gelen alanlarla birleştirilmiş olarak içeren *VTable* ortaya çıkıyor.

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

VTable

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

Bunun yerine Left Keep uygularsanız hala iki tablonuz olur. İki tablo, *A* ortak alanı aracılığıyla ilişkilendirilir.

```
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

Qlik Sense kod dilindeki Join ve Keep öneklerinden önce right öneki gelebilir.

Join öğesinden önce kullanılması durumunda, iki tablo arasındaki birleştirmenin Sağ Birleştirme olması gerektiğini belirtir. Sonuç olarak elde edilen tablo yalnızca ikinci tabloya ait tam veri kümesiyle iki tablo arasındaki bileşimleri içerir.

Keep öğesinden önce kullanılması durumunda, birinci tablonun Qlik Sense içinde depolanmadan önce ikinci tabloyla ortak kesişimine azaltılması gerektiğini belirtir.

Örnek:

Bu örneklerde *Table1* ve *Table2* kaynak tablolarını kullanıyoruz.

Table1

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

Table2

A	**C**
1	xx
4	yy

İlk olarak tablolar üzerinde bir Right Join yapıyoruz ve *Table2*'den gelen tüm satırları, *Table1* içindeki eşleşen satırlardan gelen alanlarla birleştirilmiş olarak içeren *VTable* ortaya çıkıyor.

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

VTable

A	**B**	**C**
1	aa	xx
4	-	yy

Bunun yerine Right Keep uygularsanız hala iki tablonuz olur. İki tablo, *A* ortak alanı aracılığıyla ilişkilendirilir.

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

VTab1

A	B
1	aa

VTab2

A	C
1	xx
4	yy

3.3 Kayıtlar arası fonksiyonları kullanma: Peek, Previous ve Exists

Bu fonksiyonlar, geçerli kaydın değerlendirilmesi için verilerin daha önceden yüklenmiş kayıtlarından bir değere ihtiyaç duyulduğunda kullanılır.

Dersin bu bölümünde, Peek(), Previous() ve Exists() fonksiyonlarını inceleyeceğiz.

Peek()

Peek(), bir tablodaki bir alanın zaten yüklenmiş bir satırının değerini döndürür. Satır numarası belirtilebilir (tabloda olduğu gibi). Satır numarası belirtilmezse, daha önce yüklenmiş son kayıt kullanılır.

Söz Dizimi:

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

Satır, bir tamsayı olmalıdır. 0 ilk kaydı ve 1 ikinci kaydı gösterir ve bu böyle devam eder. Negatif sayılar tablonun sonundan itibaren sırayı belirtir. -1 değeri, okunan son kaydı gösterir.

Herhangi bir satır belirtilmemişse, -1 kabul edilir.

Tablename sonunda iki nokta üst üste olmayan tablo etiketidir. *tablename* belirtilmezse geçerli tablo olduğu varsayılır. **LOAD** deyimi dışında kullanılırsa veya başka bir tabloya referansta bulunursa *tablename* dahil edilmelidir.

Previous()

Previous(), **where** cümlesi nedeniyle atılmamış önceki bir giriş kaydındaki verileri kullanarak **expr** ifadesinin değerini bulur. Bir iç tablonun ilk kaydında, bu fonksiyon NULL sonucunu döndürür.

Söz Dizimi:

```
Previous(expression)
```

Previous() fonksiyonu daha gerideki kayıtlara erişmek için iç içe geçirilebilir. Veriler, ilişkilendirilmiş veritabanında saklanmasa bile doğrudan giriş kaynağından alınır ve böylece Qlik Sense uygulamasına yüklenmemiş alanlara bile referansta bulunmayı mümkün kılar.

Exists()

Exists(), veri kod dosyasında alana daha önce belirli bir alan değerinin yüklenip yüklenmediğini belirler. Fonksiyon TRUE ya da FALSE sonucunu döndürdüğünden, bir **LOAD** deyiminin veya bir **IF** deyiminin **where** cümlesinde kullanılabilir.

Söz Dizimi:

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

Alan, kod tarafından yüklenen verilerde bulunmalıdır. *Expression*, belirtilen alanda aranacak alan değeri olarak değerlendiren bir ifadedir. Bu atlandığı takdirde, geçerli kaydın belirtilen alandaki değeri alınır.

Peek() ve Previous() kullanma

En basit biçimleriyle, Peek() ve Previous() bir tablo içindeki belirli değerleri tanımlamak için kullanılır. Bu alıştırmada yükleyeceğiniz *Employees* tablosundaki örnek verileri aşağıda bulabilirsiniz.

Employees tablosundaki örnek veriler

Tarih	İşe Alma	İşten Çıkarma
1/1/2011	6	0
2/1/2011	4	2
3/1/2011	6	1
4/1/2011	5	2

Şu anda bu, yalnızca ay, işe alma ve işten çıkarmalara yönelik verileri toplar, bu nedenle toplam çalışanlardaki aylık farkı görmek amacıyla Peek() ve Previous() fonksiyonlarını kullanarak *Employee Count* ve *Employee Var* için alanları ekleyeceğiz.

Aşağıdakileri yapın:

1. *Gelişmiş Kodlama Eğitimi* uygulamasını açın.
2. **Veri yükleme düzenleyicisi**'ne yeni bir kod bölümü ekleyin.
3. *Employees* bölümünü çağırın.
4. Sağ menüdeki **AttachedFiles** altında **Veri seç**'e tıklayın.
5. Karşıya yükleyin ve ardından *Employees.xlsx* öğesini seçin.

   > *Field names altında, veri yüklerken tablo alanlarının adlarını dahil etmek için Embedded field names öğesinin seçildiğinden emin olun.*

6. **Veri seçme kaynağı:** penceresinde **Kod yerleştir**'e tıklayın.
   Kodunuz şöyle görünmelidir:

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

7. Kodu şöyle görünecek şekilde değiştirin:

```
[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 sayfasının *Date* alanındaki tarihler AA/GG/YYYY biçimindedir. Tarihlerin sistem değişkenleri biçimi kullanılarak doğru şekilde yorumlanması için *Date* alanına Date işlevi uygulanmıştır.

   Peek() fonksiyonu, belirli bir alan için yüklenmiş herhangi bir değeri tanımlamanızı sağlar. İfadede ilk olarak rowno() öğesinin 1 değerine eşit olup olmadığına bakarız. 1'e eşitse *Employee Count* mevcut olmayacaktır, bu nedenle alanı *Hired* eksi *Terminated* ile doldururuz.

   rowno() 1'den büyükse önceki ayın *Employee Count* değerine bakar ve o ayın *Hired* değeri eksi *Terminated* çalışanlar sonucuna ilgili sayıyı ekleriz.

   Peek() fonksiyonunda da (-1) kullandığımıza dikkat edin. Bu, Qlik Sense uygulamasına geçerli kaydın üzerindeki kayda bakmasını söyler. (-1) belirtilmezse, Qlik Sense önceki kayda bakmak istediğinizi varsayar.

8. Kodunuzun sonuna şunu ekleyin:

```
[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];
```

   Previous() fonksiyonu, belirli bir alan için yüklenmiş son değeri tanımlamanızı sağlar. Bu ifadede ilk olarak rowno() öğesinin 1 değerine eşit olup olmadığına bakarız. 1'e eşitse önceki ayın *Employee Count* değeri için hiçbir kayıt olmadığından Employee Var olmayacağını biliriz. Bu nedenle, bu değer için 0 gireriz.

rowno() 1'den büyükse, *Employee Var* olacağını biliriz, bu nedenle geçen ayın *Employee Count* değerine bakar ve *Employee Var* alanında değeri oluşturmak için bu sayıyı o ayın *Employee Count* değerinden çıkarırız.

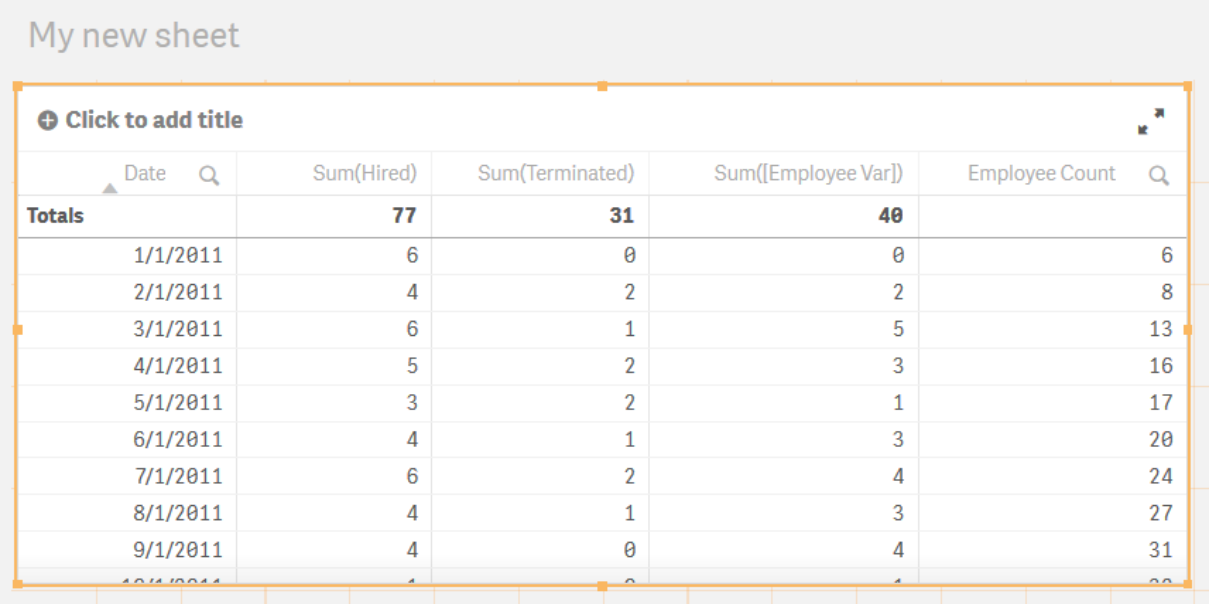
Kodunuz şöyle görünmelidir:

```
[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. **Verileri yükle**'ye tıklayın.

   Uygulamaya genel bakışta yeni bir sayfada, tablonun sütunları olarak *Date, Hired, Terminated, Employee Count* ve *Employee Var* kullanarak yeni bir tablo oluşturun. Sonuçta elde edilen tablo şöyle görünmelidir:

Kodda Peek ve Previous kullanımı sonrasındaki tablo



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() ve Previous(), bir tablo içindeki tanımlı satırları hedeflemenizi sağlar. İki fonksiyon arasındaki en büyük fark, Peek() fonksiyonunun kullanıcının daha önce koda yüklenmemiş bir alana bakmasını sağlar, Previous() fonksiyonu ise yalnızca daha önce yüklenen bir alana bakabilir. Previous(), LOAD deyiminin girdisi üzerinden, Peek() ise LOAD deyiminin çıktısı üzerinden işler. (RecNo() ile RowNo() arasındaki farkla aynıdır.) Bu, Where cümleniz varsa iki fonksiyonun farklı şekilde hareket edeceği anlamına gelir.

Bu nedenle Previous() fonksiyonu, kullanıcının önceki değerle geçerli değerin karşılaştırmasını görmesi gerektiğinde daha uygun olur. Örnekte, aydan aya çalışan farkını hesapladık.

Peek() fonksiyonu, tabloya daha önce yüklenmemiş bir alanı hedeflediğinizde veya belirli bir satırı hedeflemeniz gerektiğinde daha uygun olur. Bu, önceki ayın *Employee Count* değerine bakarak ve o ay işe alınan ve işten çıkarılan çalışanlar arasındaki farkı ekleyerek *Employee Count* değerini hesapladığımız örnekte gösterilmiştir. *Employee Count* alanının orijinal dosyadaki bir alan olmadığını unutmayın

Peek() ve Previous() öğelerinin ne zaman kullanılacağı hakkında daha fazla bilgi almak için Qlik Community bölümündeki blog gönderisine göz atın: Peek() vs Previous() – When to Use Each*. Davranışlar QlikView bağlamında açıklanmıştır. Fakat kullanılan mantık, Qlik Sense için de geçerlidir.*

Exists() kullanma

Exists() fonksiyonu, çoğu zaman veri modeline zaten ilgili veriler yüklenmişse koddaki Where cümlesiyle birlikte kullanılır.

Aşağıdaki örnekte, dizelere sayısal değerleri atamak için Dual() fonksiyonunu da kullanırız.

Aşağıdakileri yapın:

1. Yeni bir uygulama oluşturun ve ona bir ad verin.
2. **Veri yükleme düzenleyicisi**'ne yeni bir kod bölümü ekleyin.
3. *People* bölümünü çağırın.
4. Aşağıdaki kodu girin:

```
//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)
```

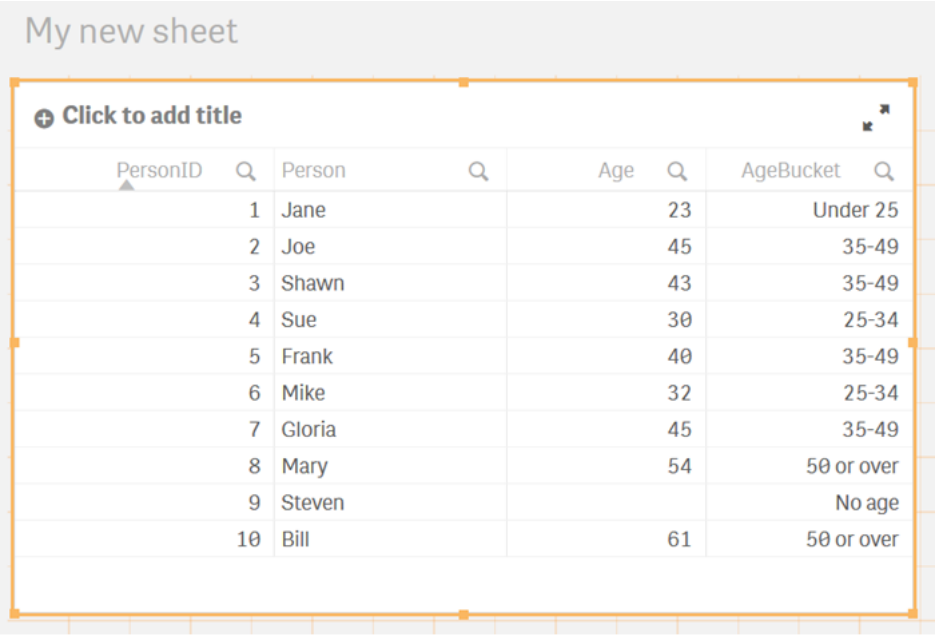
```
LOAD
    PersonID,
    Age,
       If(IsNull(Age) or Age='', Dual('No age', 5),
        If(Age<25, Dual('Under 25', 1),
         If(Age>=25 and Age <35, Dual('25-34', 2),
          If(Age>=35 and Age<50, Dual('35-49' , 3),
           If(Age>=50, Dual('50 or over', 4)
            ))))) as AgeBucket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

DROP Table AgeTemp;
```

5. **Verileri yükle**'ye tıklayın.
   Kodda, *Age* ve *AgeBucket* alanları yalnızca *PersonID* veri modeline zaten yüklenmişse yüklenir.

   AgeTemp tablosunda *PersonID* 11 ve 12 için listelenen yaşlar olduğuna, ancak bu kimlikler veri modelinde (*People* tablosunda) yüklenmediği için Where Exists(PersonID) cümlesi tarafından hariç tutulduklarına dikkat edin. Bu cümle, şu şekilde de yazılabilir: Where Exists(PersonID, PersonID).

   Kodun çıktısı şu şekilde görünmelidir:

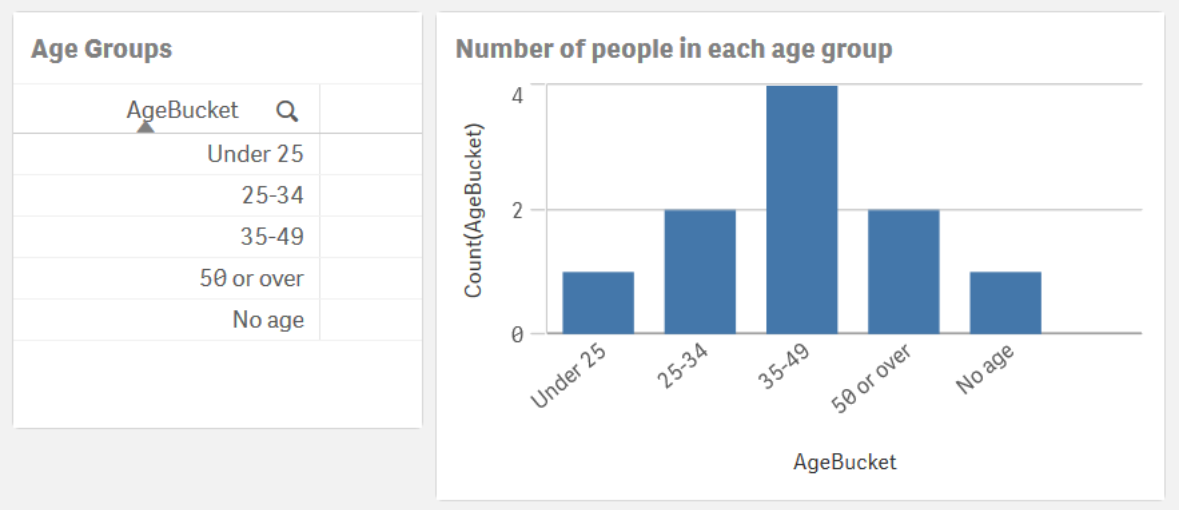
   Kodda Exists kullanımı sonrasındaki tablo



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

   AgeTemp tablosundaki *PersonID*'lerin hiçbiri veri modeline yüklenmezse, *Age* ve *AgeBucket* alanları *People* tablosuyla birleştirilmez. Exists() fonksiyonunun kullanılması, veri modelinde artık kayıtların/verilerin, yani herhangi bir kişiyle ilişkilendirilmemiş *Age* ve *AgeBucket* alanlarının olmasını önler.

6. Yeni bir sayfa oluşturun ve sayfaya bir ad verin.
7. Yeni sayfayı açın ve **Sayfayı düzenle**'ye tıklayın.
8. Sayfaya *AgeBucket* boyutuna sahip bir standart tablo ekleyin ve görselleştirmeye *Yaş Grupları* adını verin.
9. *AgeBucket* boyutu ve *Count([AgeBucket])* hesaplamasıyla sayfaya sütun grafik ekleyin. Görselleştirmeye *Number of people in each age group* adını verin.
10. Tablonun ve sütun grafiğin özelliklerini isteğinize göre düzenleyin ve ardından **Bitti** seçeneğine tıklayın.
    Sayfanız şöyle görünür:
    Yaşa göre gruplandırmaya sahip sayfa



Dual() fonksiyonu, dizeye sayısal bir değer atamak gerektiğinde kodda veya grafik ifadesinde faydalı olur.

Yukarıdaki kodda yaşları yükleyen bir uygulamanız vardır ve gerçek yaşlara karşılık yaş demetlerini temel alan görselleştirmeler oluşturmak için bu yaşları demetlere ayırmaya karar verirsiniz. 25 yaşında altındaki kişiler, 25 ve 35 yaş arası kişiler, vb. yaş grupları için bir demet olur. Dual() fonksiyonu kullanılarak, yaş demetlerine daha sonra liste kutusunda veya grafikte yaş demetlerini sıralamak için kullanılabilecek sayısal bir değer atanabilir. Bu nedenle, uygulama sayfasında olduğu gibi sıralama "Yaş yok"u listenin sonuna alır.

Exists() ve Dual() hakkında daha fazla bilgi almak için Qlik Community bölümündeki blog gönderisine göz atın: Dual & Exists – Useful Functions (Dual & Exists – Faydalı Fonksiyonlar)

3.4 Aralıkları ve yinelemeli yüklemeyi eşleştirme

Bir LOAD veya SELECT deyimine yönelik Intervalmatch öneki, ayrık sayısal değerleri bir veya daha fazla sayısal aralığa bağlamak için kullanılır. Bu, örneğin üretim ortamlarında kullanılabilen oldukça etkili bir özelliktir.

IntervalMatch() önekini kullanma

En temel aralık eşleşmesi, tek bir tabloda sayıların ve tarihlerin listesine ve ikinci bir tabloda aralıkların listesine sahip olduğunuzda gerçekleştirilir. Amaç, iki tabloyu bağlamaktır. Genel olarak arada çokluk ilişkisi vardır; yani aralığa ait birçok tarih olabilir ve tarih ise birçok aralığa ait olabilir. Bunu çözmek için iki orijinal tablo arasında bir köprü tablosu oluşturmanız gerekir. Bunu yapmanın birçok yolu vardır.

Qlik Sense uygulamasında bunu çözmenin en kolay yolu, LOAD veya SELECT deyiminin önünde IntervalMatch() önekini kullanmaktır. LOAD/SELECT deyiminin yalnızca iki alanı içermesi gerekir: Aralıkları tanımlayan From ve To alanları. IntervalMatch() öneki, yüklenen aralıklar ile daha önce yüklenmiş ve önekin parametresi olarak belirtilmiş sayısal alan arasında tüm kombinasyonları oluşturacaktır.

Aşağıdakileri yapın:

1. Yeni bir uygulama oluşturun ve ona bir ad verin.
2. **Veri yükleme düzenleyicisi**'ne yeni bir kod bölümü ekleyin.
3. *Events* bölümlerini çağırın.
4. Sağ menüdeki **AttachedFiles** altında **Veri seç**'e tıklayın.
5. Karşıya yükleyin ve ardından *Events.txt* öğesini seçin.
6. **Veri seçme kaynağı:** penceresinde **Kod yerleştir**'e tıklayın.
7. Karşıya yükleyin ve ardından *Intervals.txt* öğesini seçin.
8. **Veri seçme kaynağı:** penceresinde **Kod yerleştir**'e tıklayın.
9. Kodda, ilk tabloyu *Events* olarak adlandırın ve ikinci tabloyu *Intervals* olarak adlandırın.
10. İlk iki tablo arasında köprü olan üçüncü bir tablo oluşturmak için kod sonuna IntervalMatch ekleyin:

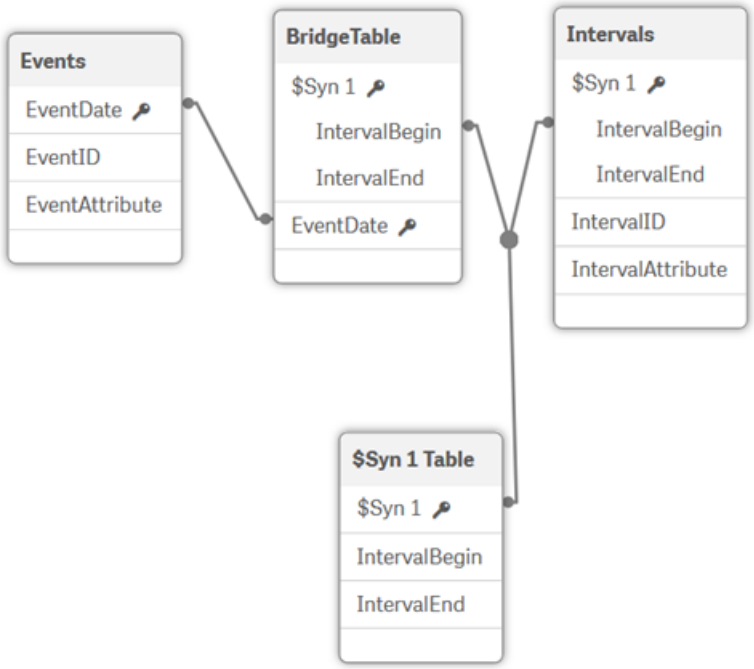
```
BridgeTable:
IntervalMatch (EventDate)
LOAD distinct IntervalBegin, IntervalEnd
Resident Intervals;
```

11. Kodunuz şöyle görünmelidir:

```
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. **Verileri yükle**'ye tıklayın.
13. **Veri modeli görüntüleyicisi**'ni açın. Veri modeli artık şöyle görünür:
    Veri modeli: Events, BridgeTable, Intervals ve $Syn1 tabloları



Veri modeli, Qlik Sense yapay anahtarı olarak gösterilecek bileşik anahtar (*IntervalBegin* ve *IntervalEnd* alanları) içerir.

Temel tablolar şunlardır:

- Olay başına tam olarak bir kayıt içeren *Events* tablosu.
- Aralık başına tam olarak bir kayıt içeren *Intervals* tablosu.
- Olay ve aralık kombinasyonu başına tam olarak bir kayıt içeren ve önceki iki tabloyu bağlayan köprü tablosu.

Aralıklar çakışıyorsa olay birkaç aralığa ait olabilir. Bir aralığın da ona ait birkaç olayı olabilir.

Bu veri modeli, normalleştirildiği ve kompakt olduğu için optimumdur. Hem *Events* tablosu hem de *Intervals* tablosu değiştirilmemiştir ve orijinal sayıda kayıt içerir. Bu tablolarda çalışan Count(EventID) gibi tüm Qlik Sense hesaplamaları doğru şekilde çalışacak ve değerlendirilecektir.

IntervalMatch() hakkında daha fazla bilgi almak için Qlik Community bölümündeki blog gönderisine göz atın: Using IntervalMatch() (IntervalMatch() Kullanma)

While döngüsü ve IterNo() yinelemeli yüklemesi kullanma

While döngüsü ve aralığın alt ve üst sınırları arasında numaralandırılabilir değerler oluşturan IterNo() kullanarak neredeyse aynı köprü tablosunu elde edebilirsiniz.

While cümlesi kullanılarak LOAD deyiminin içinde bir döngü oluşturulabilir: Örneğin:

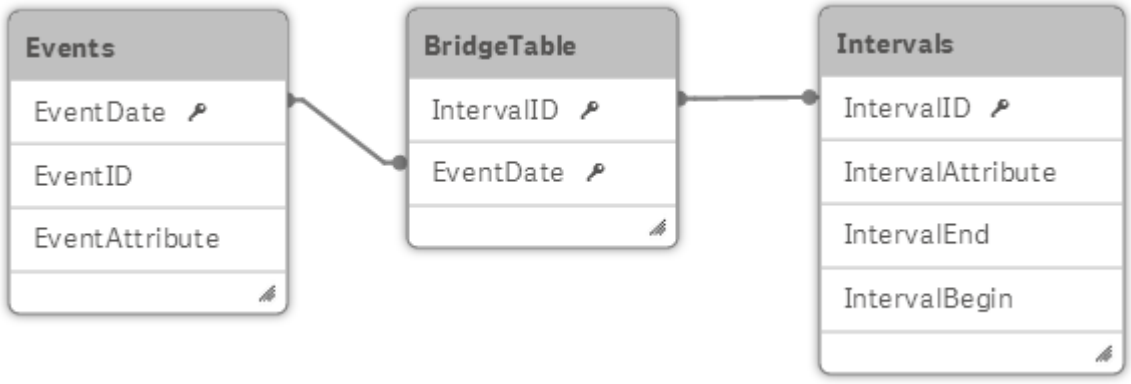
```
LOAD Date, IterNo() as Iteration From … While IterNo() <= 4;
```

Böyle bir LOAD deyimi, her giriş kaydının üzerinde döngüye girecek ve While cümledeki ifade doğru olduğu sürece bunu tekrar tekrar yükleyecektir. IterNo() fonksiyonu ilk yinelemede “1”i ve ikincide “2”yi döndürür (bu düzende devam eder).

Aralıklar için birincil IntervalID anahtarınız vardır, bu nedenle koddaki tek fark köprü tablosunun nasıl oluşturulduğudur:

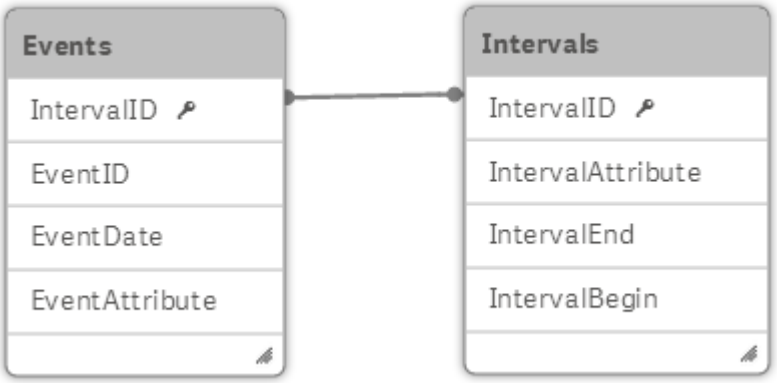
Aşağıdakileri yapın:

1. Mevcut Bridgetable deyimlerini aşağıdaki kodla değiştirin:
   ```
   BridgeTable:
   LOAD distinct * Where Exists(EventDate);
   LOAD IntervalBegin + IterNo() - 1 as EventDate, IntervalID
       Resident Intervals
       While IntervalBegin + IterNo() - 1 <= IntervalEnd;
   ```
2. **Verileri yükle**'ye tıklayın.
3. **Veri modeli görüntüleyicisi**'ni açın. Veri modeli artık şöyle görünür:
   Veri modeli: Events, BridgeTable ve Intervals tabloları



Genel olarak üç tablolu çözüm, aralıklar ve olaylar arasında çokluk ilişkisine imkan tanıdığı için en ideal olandır. Ancak bir olayın yalnızca tek bir aralığa ait olabileceğini bilmeniz de sık karşılaşılan bir durumdur. Bu durumda köprü tablosu gerekli değildir. *IntervalID* doğrudan olay tablosunda depolanabilir. Bunu yapmanın birkaç yolu vardır, ancak en kullanışlı olanı Bridgetable öğesini *Events* tablosuyla birleştirmektir.

4. Kodunuzun sonuna şu kodu ekleyin:
```
Join (Events)
LOAD EventDate, IntervalID
Resident BridgeTable;

Drop Table BridgeTable;
```
5. **Verileri yükle**'ye tıklayın.
6. **Veri modeli görüntüleyicisi**'ni açın. Veri modeli artık şöyle görünür:
Veri modeli: Events ve Intervals tabloları



Açık ve kapalı aralıklar

Bir aralığın açık veya kapalı olduğu, uç noktalarının aralığa dahil olup olmasına göre belirlenir.

- Uç noktaları dahilse, kapalı bir aralıktır:
$[a,b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$
- Uç noktaları dahil değilse, açık bir aralıktır:
$]a,b[= \{x \in \mathbb{R} \mid a < x < b\}$
- Bir uç noktası dahilse, yarı açık bir aralıktır:
$[a,b[= \{x \in \mathbb{R} \mid a \leq x < b\}$

Aralıkların çakıştığı ve bir sayının birden fazla aralığa ait olabileceği bir durumla karşılaşırsanız, genellikle kapalı aralıkları kullanmanız gerekir.

Ancak, bazı durumlarda çakışan aralıklar istemezsiniz, bir sayının yalnızca bir aralığa ait olmasını istersiniz. Bu nedenle, bir nokta aralığın sonundaysa ve aynı anda diğerinin başındaysa bir sorunla karşılaşırsınız. Bu değere sahip bir sayı, iki aralığa da bağlanabilir. Bu nedenle, yarı açık aralıklar istersiniz.

Bu sorunun pratik bir çözümü, tüm aralıkların son değerinden çok küçük bir miktarı çıkarmak ve böylece kapalı, çakışmayan aralıklar oluşturmaktır. Sayılarınız tarihse, bunu yapmanın en kolay yolu günün son milisaniyesini döndüren DayEnd() fonksiyonunu kullanmaktır:

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

Küçük bir miktarı manuel olarak da çıkarabilirsiniz. Bunu yaparsanız, işlem 52 anlamlı ikili basamağa (14 ondalık basamak) yuvarlanacağı için çıkarılan miktarın çok küçük olmadığından emin olun. Çok küçük bir miktar kullanırsanız fark çok önemli olmayacak ve orijinal sayıyı tekrar kullanacaksınız.

4 Veri temizleme

Qlik Sense içine yüklediğiniz kaynak verilerinin Qlik Sense uygulamasında istediğiniz gibi olmadığı zamanlar olabilir. Qlik Sense, verilerimizi işinize yarayan bir biçime dönüştürmenizi sağlayan çok sayıda fonksiyon ve deyim sağlar.

Kod çalıştırıldığında alan değerlerini veya adlarını değiştirmek için Qlik Sense kodunda eşleme kullanılabilir; bu nedenle eşleme, verileri temizlemek ve daha tutarlı kılmak veya alan değerinin bir kısmını ya da tümünü değiştirmek için kullanılabilir.

Farklı tablolardan veri yüklediğinizde, aynı şeyi belirten alan değerleri her zaman tutarlı şekilde adlandırılmaz. Bu tutarlılık eksikliği ilişkileri engellediğinden, sorunun çözülmesi gerekir. Bu, alan değerlerinin karşılaştırılması için bir eşleme tablosu oluşturarak düzenli bir şekilde gerçekleştirilebilir.

4.1 Eşleme tabloları

Mapping yükleme veya Mapping seçme aracılığıyla yüklenen tablolar, diğer tablolardan farklı bir şekilde işlenir. Bunlar, belleğin ayrı bir alanında depolanır ve yalnızca kod çalıştırıldığında eşleme tabloları olarak kullanılır. Kod çalıştırıldıktan sonra, bu tablolar otomatik olarak bırakılır.

Kurallar:

- Eşleme tablosu, birincisi karşılaştırma değerlerini içerecek ve ikincisi istenen eşleme değerlerini içerecek şekilde iki sütuna sahip olmalıdır.
- İki sütun adlandırılmalıdır ancak adlar kendi içinde ilgili olmamalıdır. Sütun adları, normal dahili tablolardaki alan adlarıyla bağlantılı değildir.

4.2 Mapping işlevleri ve deyimleri

Bu derste aşağıdaki eşleme fonksiyonları/deyimleri ele alınacaktır:

• Mapping öneki

• ApplyMap()

• MapSubstring()

• Map ... Using deyimi

• Unmap deyimi

4.3 Mapping öneki

Mapping öneki, bir kodda eşleme tablosu oluşturmak için kullanılır. Eşleme tablosu ApplyMap() fonksiyonu, MapSubstring() fonksiyonu veya Map ... Using deyimi ile kullanılabilir.

Aşağıdakileri yapın:

1. Yeni bir uygulama oluşturun ve ona bir ad verin.
2. **Veri yükleme düzenleyicisi**'ne yeni bir kod bölümü ekleyin.
3. *Countries* bölümünü çağırın.
4. Aşağıdaki kodu girin:

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

   CountryMap tablosu iki sütunu depolar: *Country* ve *NewCountry*. *Country* sütunu, *Country* alanına ülke girmek için kullanılan çeşitli yöntemleri depolar. *NewCountry* sütunu, değerlerin nasıl eşleneceğini depolar. Bu eşleme tablosu, *Country* alanında tutarlı *US* ülke değerlerini depolamak için kullanılacaktır. Örneğin, *Country* alanında *U.S.A.* depolanırsa, *US* ile eşleyin.

4.4 ApplyMap() fonksiyonu

Bir alandaki verileri daha önce oluşturulan eşleme tablosuna göre değiştirmek için ApplyMap() öğesini kullanın. ApplyMap() fonksiyonunun kullanılabilmesi için önce eşleme tablosunun yüklenmesi gerekir. Yükleyeceğiniz *Data.xlsx* tablosundaki veriler şu şekilde görünür:

Veri tablosu

Kimlik	Ad	Ülke	Kod
1	John Black	U.S.A.	SDFGBS1DI
2	Steve Johnson	U.S.	2ABC
3	Mary White	Birleşik Devletler	DJY3DFE34
4	Susan McDaniels	u	DEF5556
5	Dean Smith	US	KSD111DKFJ1

Ülkenin farklı şekillerde girildiğine dikkat edin. Ülke alanını tutarlı kılmak için eşleme tablosu yüklenir ve **ApplyMap()** fonksiyonu kullanılır.

Aşağıdakileri yapın:

1. Yukarıda girdiğiniz kodun altına *Data.xlsx* dosyasını seçin ve yükleyin, ardından kodu yerleştirin.
2. Yeni oluşturulan LOAD deyiminin üzerine şunu girin:

```
Data:
```

Kodunuz şöyle görünmelidir:

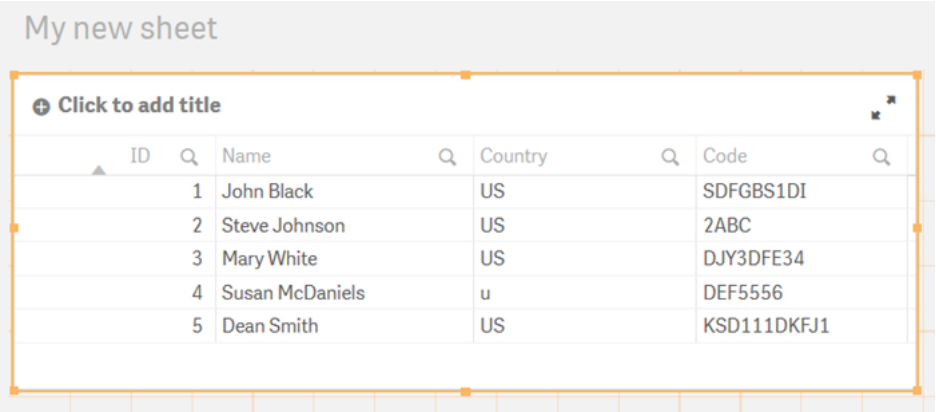
```
CountryMap:
MAPPING LOAD * INLINE [
    Country, NewCountry
    U.S.A., US
    U.S., US
    United States, US
    United States of America, US
];

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

3. `Country`, içeren satırı aşağıdaki gibi değiştirin:
   `ApplyMap('CountryMap', Country) as Country,`

   ApplyMap() fonksiyonunun ilk parametresi, tek tırnak içine alınmış eşleme adına sahiptir. İkinci parametre, değiştirilecek verileri içeren alandır.

4. **Verileri yükle**'ye tıklayın.
   Elde edilen tablo şöyle görünür:

   ApplyMap() işlevi kullanılarak yüklenen verileri gösteren tablo



   United States'ın tüm yazımları *US* olarak değiştirilmiştir. Doğru yazılmamış bir kayıt vardır ve bu nedenle ApplyMap() fonksiyonunu bu alan değerini değiştirmemiştir. ApplyMap() fonksiyonunu kullanarak, eşleme tablosunda eşleşen değer yoksa varsayılan bir ifade eklemek için üçüncü parametreyi kullanabilirsiniz.

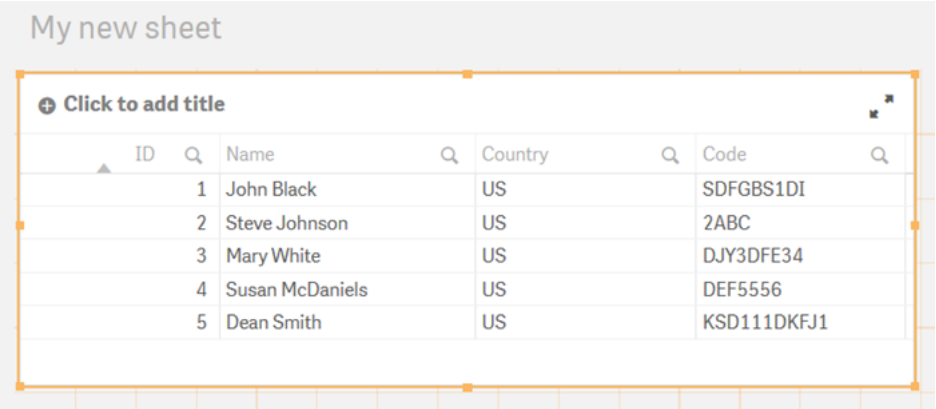
5. Ülkenin yanlış girilebileceği durumları yönetmek için ApplyMap() fonksiyonunun üçüncü parametresi olarak `'US'` ekleyin:
   `ApplyMap('CountryMap', Country, 'US') as Country,`

Kodunuz şöyle görünmelidir:

```
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. **Verileri yükle**'ye tıklayın.
   Elde edilen tablo şöyle görünür:

   ApplyMap işlevi kullanılarak yüklenen verileri gösteren tablo



ApplyMap() hakkında daha fazla bilgi almak için Qlik Community bölümündeki blog gönderisine göz atın: Don't join - use Applymap instead (Join yerine Applymap kullanın)

4.5 MapSubstring() fonksiyonu

MapSubstring() fonksiyonu alanın bölümlerini eşlemenizi sağlar.

ApplyMap() tarafından oluşturulan tabloda sayıların metin olarak yazılmasını istiyoruz, bu nedenle sayısal verileri metinle değiştirmek için MapSubstring() fonksiyonu kullanılacaktır.

Bunu yapmak için önce bir eşleme tablosunun oluşturulması gerekir.

Aşağıdakileri yapın:

1. *CountryMap* bölümünden sonra ve *Data* bölümünden önce, aşağıdaki kod satırlarını ekleyin.

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

   CodeMap tablosunda, 1 ile 5 arasındaki sayılar ve 11 eşlenir.

2. Kodun *Data* bölümünde `Code` deyimini aşağıdaki gibi değiştirin:

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

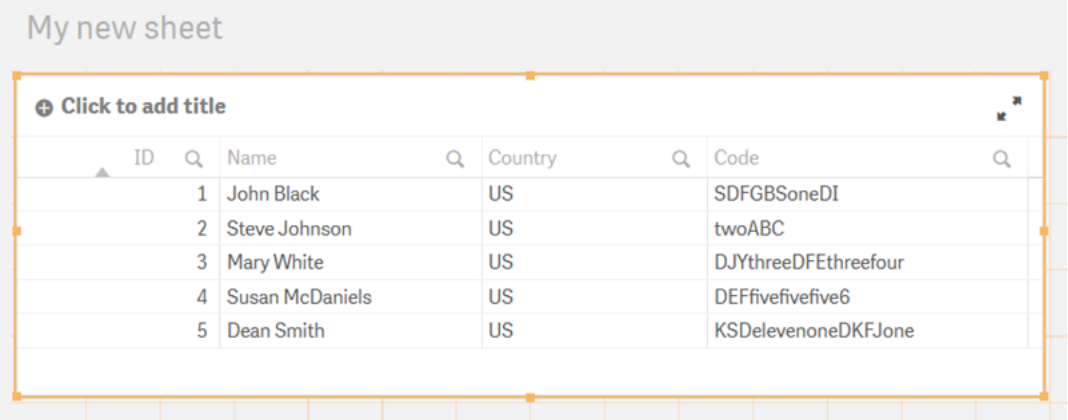
   Kodunuz şöyle görünmelidir:

```
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. **Verileri yükle**'ye tıklayın.

   Elde edilen tablo şöyle görünür:

MapSubString işlevi kullanılarak yüklenen verileri gösteren tablo



Sayısal karakterler, *Code* alanındaki metinle değiştirilmiştir. Bir sayı ID=3 ve ID=4 için bir kereden fazla görünürse, metin de tekrarlanır. ID=4. *Susan McDaniels* kodunda 6 vardır. 6 *CodeMap* tablosunda eşlenmediği için değiştirilmeden kalır. ID=5, *Dean Smith* kodunda 111 vardır. Bu 'elevenone' olarak eşlenmiştir.

MapSubstring() hakkında daha fazla bilgi almak için Qlik Community bölümündeki blog gönderisine göz atın: Mapping ... and not the geographical kind (Mapping ancak coğrafi olmayan türden)

4.6 Map ... Using

Alana bir eşlemeyi uygulamak için Map ... Using deyimi de kullanılabilir. Fakat ApplyMap() deyiminden farklı çalışır. Alan adıyla her karşılaşıldığında eşlemeyi ApplyMap() yönetir, dahili tablodaki alan adının altında değer depolandığında ise eşlemeyi Map ... Using yönetir.

Şimdi bir örneği inceleyelim. *Country* alanını kodda bir kereden fazla yüklediğimizi ve alan her yüklendiğinde bir eşleme uygulamak istediğimizi varsayalım. Bu derste daha önce açıklandığı şekilde ApplyMap() fonksiyonu veya Map ... Using kullanılabilir.

Map ... Using kullanılırsa, alan dahili tabloda depolandığında eşleme alana uygulanır. Bu nedenle aşağıdaki örnekte, eşleme Data1 tablosundaki *Country* alanına uygulanır, ancak *Data2* tablosundaki *Country2* alanına uygulanmaz. Bunun nedeni, Map ... Using deyiminin yalnızca *Country* adlı alanlara uygulanmasıdır. *Country2* alanı dahili tabloda depolandığında, artık *Country* olarak adlandırılmaz. Eşlemenin *Country2* tablosuna uygulanmasını istiyorsanız, ApplyMap() fonksiyonunu kullanmanız gerekir.

Unmap deyimi Map ... Using deyimini sonlandırır, bu nedenle Unmap deyiminden sonra *Country* yüklenecekse, *CountryMap* uygulanmaz.

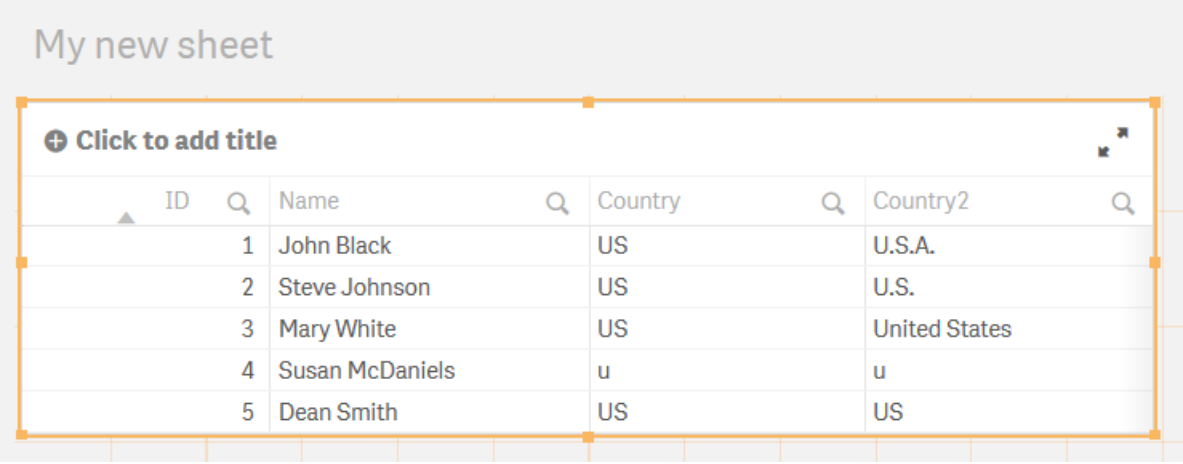
Aşağıdakileri yapın:

1. *Data* tablosunun kodunu aşağıdaki ile değiştirin:

```
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. **Verileri yükle**'ye tıklayın.
   Elde edilen tablo şöyle görünür:

   Map ... Using işlevi kullanılarak yüklenen verileri gösteren tablo



5 Hiyerarşik verileri işleme

Hiyerarşiler, tüm iş zekası çözümlerinin farklı ayrıntı düzeyleri içeren boyutları açıklamak için kullanılan önemli bir parçasıdır. Bazıları basit ve kolay, bazıları ise karmaşıktır ve doğru bir şekilde modellenmesi için üzerlerinde çok fazla çalışmanız gerekir.

Hiyerarşinin en üstünden en altına doğru, üyeler kademeli olarak daha ayrıntılı olur. Örneğin, Pazar, Ülke, Eyalet ve Şehir düzeylerine sahip bir boyutta, Kuzey ve Güney Amerika hiyerarşinin en üstteki düzeyinde, ABD üyesi ikinci düzeyde, California üyesi üçüncü düzeyde ve San Francisco en alttaki düzeyde görünür. California, ABD'ye göre daha geneldir ve San Francisco da California'ya göre daha geneldir.

Hiyerarşilerin ilişkisel modelde depolanması, birden fazla çözümde sık karşılaşılan bir zorluktur. Birkaç yaklaşım bulunur:

- Yatay hiyerarşi
- Bitişiklik liste modeli
- Yol numaralandırma yöntemi
- İçe içe setler modeli
- Üst öğe listesi

Bu dersin amaçları doğrultusunda, hiyerarşiyi doğrudan sorguda kullanılabilen bir formda sunduğu için bir Üst öğe listesi oluşturacağız. Diğer yaklaşımlar hakkında daha fazla bilgi, Qlik Community bölümünde bulunabilir.

5.1 Hierarchy öneki

Hierarchy öneki, bitişik düğümler tablosu yükleyen bir LOAD veya SELECT deyiminin önüne yerleştirdiğiniz bir kod komutudur. LOAD deyiminin en az üç alana sahip olması gerekir: Düğüm için benzersiz anahtar olan bir kimlik, üst öğenin referansı ve ad.

Önek, yüklenen bir tabloyu genişletilmiş düğümler tablosuna dönüştürecektir; bu tablo, her hiyerarşi seviyesi için birer tane olmak üzere birçok ek sütuna sahip olur.

Aşağıdakileri yapın:

1. Yeni bir uygulama oluşturun ve ona bir ad verin.
2. **Veri yükleme düzenleyicisi**'ne yeni bir kod bölümü ekleyin.
3. *Wine* bölümünü çağırın.
4. Sağ menüdeki **AttachedFiles** altında **Veri seç**'e tıklayın.
5. Karşıya yükleyin ve ardından *Winedistricts.txt* öğesini seçin.
6. **Veri seçme kaynağı** penceresinde, yüklenmemeleri için *Lbound* ve *RBound* alanlarının işaretini kaldırın.
7. **Kod ekle**'ye tıklayın.

8. LOAD deyiminin üzerine şunu girin:
   ```
   Hierarchy (NodeID, ParentID, NodeName)
   ```

   Kodunuz şöyle görünmelidir:

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

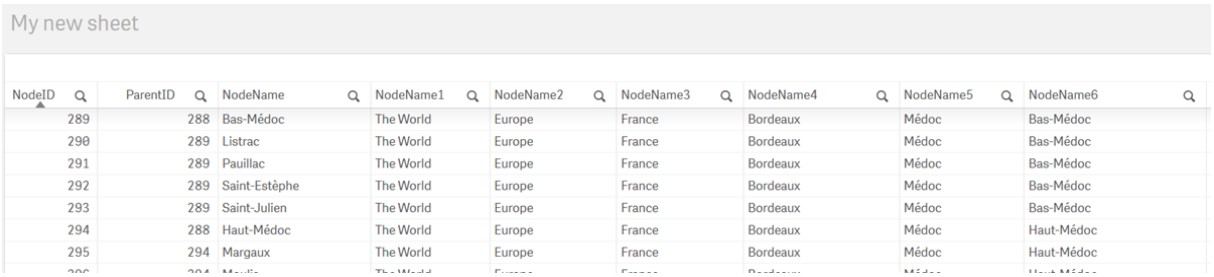
9. **Verileri yükle**'ye tıklayın.
10. Ortaya çıkan tabloyu görüntülemek için **Veri modeli görüntüleyicisi**'nin **Önizleme** bölümünü kullanın.

    Elde edilen genişletilmiş düğümler tablosu, kaynak tablosuyla tam olarak aynı sayıda kayda sahiptir: Düğüm başına bir. Genişletilmiş düğümler tablosu, ilişkisel modelde bir hiyerarşiyi analiz etmeye yönelik birçok gereksinimi karşıladığı için çok kullanışlıdır:

    - Tüm düğüm adları, aramalarda kullanılabilmeleri için tek ve aynı sütunda yer alır.
    - Ayrıca, farklı düğüm düzeylerinin her biri tek bir alana, başka bir deyişle detaya inme gruplarında veya pivot tablolarda boyut olarak kullanılabilen alanlara genişletilmiştir.
    - Ayrıca, farklı düğüm düzeylerinin her biri tek bir alana, yani detaya inme gruplarında kullanılabilen alanlara genişletilmiştir.
    - Düğüm için benzersiz olan ve sağ taraftaki tüm üst öğeleri listeleyen bir yolu içerecek şekilde ayarlanabilir.
    - Düğümün derinliğini, yani köke mesafesini içerecek şekilde de ayarlanabilir.

    Elde edilen tablo şöyle görünür:

    Hierarchy öneki kullanılarak yüklenen örnek veriyi gösteren tablo

    
    My new sheet

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 öneki

Hierarchy öneki gibi, HierarchyBelongsTo öneki de bitişik düğümler tablosu yükleyen bir LOAD veya SELECT deyiminin önüne yerleştirdiğiniz bir kod komutudur.

Burada da LOAD deyiminin en az üç alana sahip olması gerekir: Düğüm için benzersiz anahtar olan bir kimlik, üst öğenin referansı ve ad. Önek, yüklenen tabloyu üst öğe tablosuna, yani ayrı bir kayıt olarak listelenen her üst öğe ve alt öğe kombinasyonuna sahip bir tabloya dönüştürecektir. Bu nedenle, belirli bir düğümün tüm üst öğelerini ve alt öğelerini bulmak çok kolaydır.

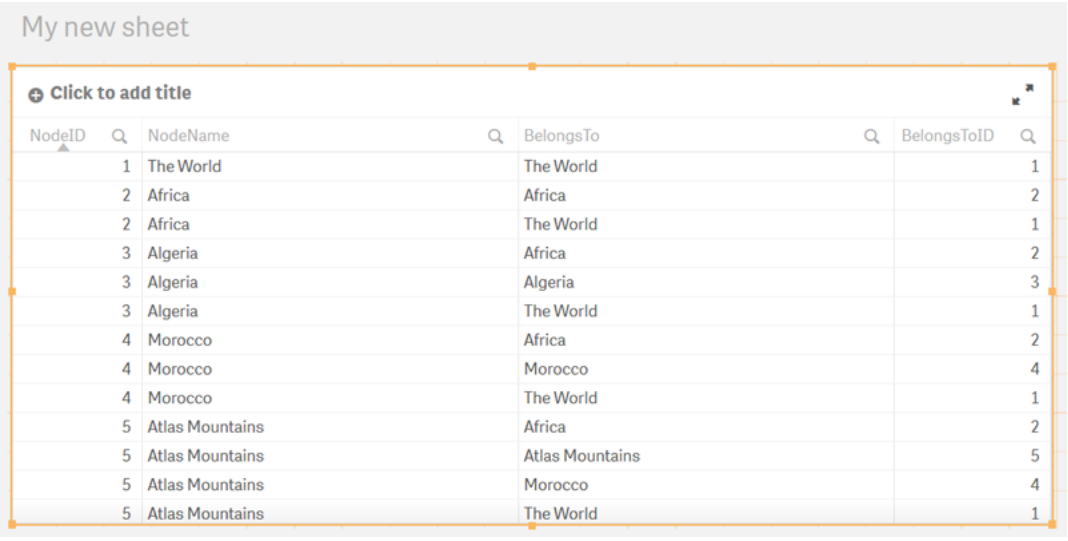
Aşağıdakileri yapın:

1. **Veri yükleme düzenleyicisi**'nde Hierarchy deyimini şu şekilde değiştirin:
   `HierarchyBelongsTo (NodeID, ParentID, NodeName, BelongsToID, BelongsTo)`
2. **Verileri yükle**'ye tıklayın.
3. Ortaya çıkan tabloyu görüntülemek için **Veri modeli görüntüleyicisi**'nin **Önizleme** bölümünü kullanın.
   Üst öğe tablosu, ilişkisel modelde bir hiyerarşiyi analiz etmeye yönelik birçok gereksinimi karşılar:

   - Düğüm kimliği tekli düğümleri temsil ediyorsa, üst kimlik hiyerarşinin tüm ağaçlarını ve alt ağaçlarını temsil eder.
   - Tüm düğüm adları, hem düğümler hem de ağaçlar olarak rol alır ve her iki rol de aramalar için kullanılabilir.
   - Düğüm derinliği ve kök ile alt ağaç arasındaki mesafe olan üst öğe derinliği arasındaki derinlik farkını içerecek şekilde ayarlanabilir.

   Elde edilen tablo şöyle görünür:

   HierarchyBelongsTo öneki kullanılarak yüklenen veriyi gösteren tablo



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

Yetkilendirme

Bir hiyerarşinin yetkilendirme için kullanılması yaygın bir durumdur. Bir örneği organizasyonel hiyerarşidir. Her yönetici, tüm alt departmanlar da dahil olmak üzere kendi departmanlarına ilişkin her şeyi görme hakkına sahiptir. Ancak diğer departmanları görme hakkına sahip olmamaları gerekir.

Organizasyonel hiyerarşi örneği

Bu, farklı insanların organizasyonun farklı alt ağaçlarını görmesine izin verileceği anlamına gelir. Yetkilendirme tablosu şöyle görünebilir:

Yetkilendirme tablosu

ACCESS	NTNAME	PERSON	POSITION	PERMISSIONS
KULLANICI	ACME\JRL	John	CPO	HR
KULLANICI	ACME\CAH	Carol	CEO	CEO
KULLANICI	ACME\JER	James	Mühendislik Direktörü	Mühendislik
KULLANICI	ACME\DBK	Diana	CFO	Finans
KULLANICI	ACME\RNL	Bob	COO	Satışlar
KULLANICI	ACME\LFD	Larry	CTO	Ürün

Bu durumda, *Carol*'ın *CEO* ve alt departmanlarla ilgili her şeyi görmesine izin verilecek; *Larry*'nin *Product* organizasyonunu görmesine izin verilecek ve *James*'in yalnızca *Engineering* organizasyonunu görmesine izin verilecektir.

Örnek:

Hiyerarşi genellikle bitişik düğümler tablosunda depolanır. Bu örnekte, sorunu çözmek için HierarchyBelongsTo kullanarak bitişik düğümler tablosunu yükleyip *Tree* üst öğe alanını adlandırabilirsiniz.

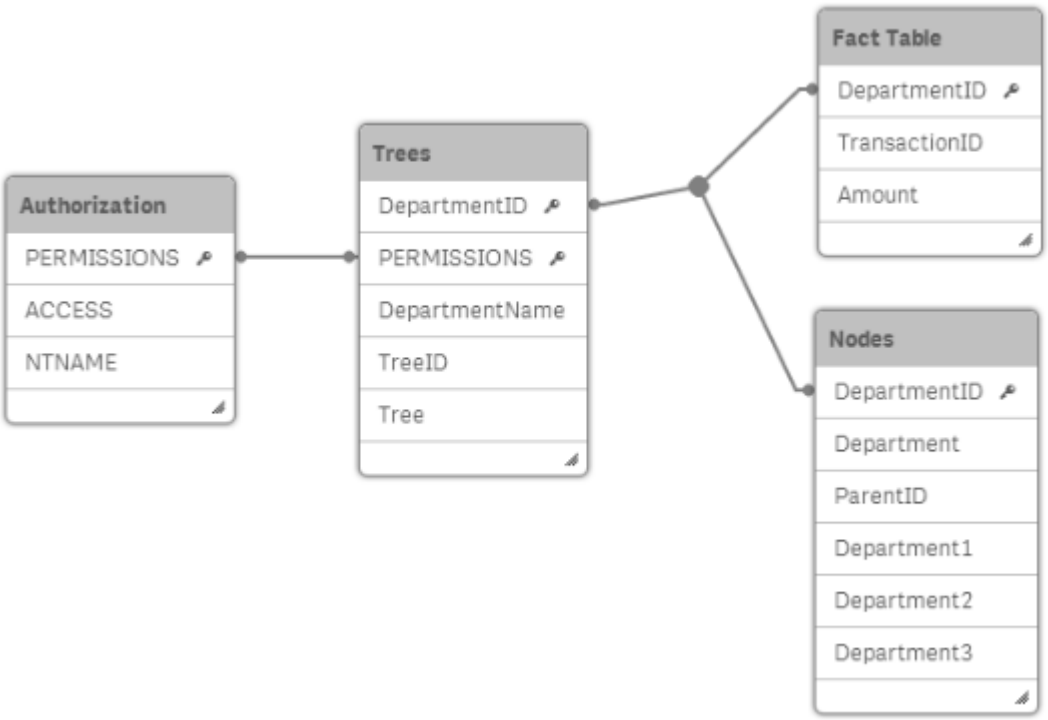
Section Access kullanmak istiyorsanız *Tree* alanının büyük harfli bir kopyasını yüklemeniz ve bu yeni *PERMISSIONS* alanını çağırmanız gerekir. Son olarak, yetkilendirme tablosunu yüklemeniz gerekir. Bu son iki adım şu kod satırı kullanılarak gerçekleştirilir. TempTrees tablosunun HierarchyBelongsTo deyimi tarafından oluşturulan bir tablo olduğunu unutmayın.

Bunun yalnızca bir örnek olduğunu unutmayın. Qlik Sense içinde tamamlanması gereken başka bir alıştırma yoktur.

```
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;
```

Bu örnek, aşağıdaki veri modelini oluşturur:

Veri modeli: Authorization, Trees, Fact ve Nodes tabloları



6 QVD dosyaları

QVD (QlikView Data) dosyası, Qlik Sense veya QlikView uygulamasından dışa aktarılmış bir veri tablosu içeren bir dosyadır. QVD, yerel bir Qlik biçimidir ve yalnızca Qlik Sense uygulamasına yazılabilir ve QlikView tarafından okunabilir. Dosya biçimi bir Qlik Sense kodundan veri okuma sırasında hız sağlamak için optimize edilmiştir, ancak oldukça kompakt olan yapısını korumaktadır. Dosya biçimi bir koddan veri okuma sırasında hız sağlamak için optimize edilmiştir, ancak oldukça kompakt olan yapısını korumaktadır. Bir QVD dosyasından veri okumak, normalde diğer veri kaynaklarından okumaktan 10-100 kat daha hızlıdır.

QVD dosyaları iki modda okunabilir: standart (hızlı) ve optimize (daha hızlı). Seçilen mod, Qlik Sense kod altyapısı tarafından otomatik olarak belirlenir. Optimize mod yalnızca tüm yüklü alanların herhangi bir dönüşüm (alanlar üzerinde etkili olan formüller) olmadan okunduğu durumlarda kullanılabilir, ancak alanların yeniden adlandırılmasına izin verilir. Qlik Sense uygulamasının kayıtların paketini açmasına neden olan bir Where cümlesi de optimize yüklemeyi devre dışı bırakır.

QVD dosyası tam olarak bir veri tablosu içerir ve üç parçadan oluşur:

- Tablodaki alanları, sonraki bilgilerin düzenini ve bazı diğer meta verileri açıklayan bir XML üst bilgisi (UTF-8 karakter kümesinde).
- Bayt dolgulamalı biçimdeki sembol tabloları.
- Bit dolgulamalı biçimdeki gerçek tablo verisi.

QVD dosyaları birçok amaçla kullanılabilir. Dört ana kullanım kolayca tanımlanabilir. Birden fazla kullanım, belirli bir durum için geçerli olabilir:

- Veri yükleme hızını artırma

  Değişmeyen veya yavaş değişen giriş verilerinin QVD dosyalarında ara belleğe alınmasıyla, kod yürütme büyük veri kümeleri için önemli ölçüde hızlandırılır.

- Veritabanı sunucularında yükü azaltma

  Harici veri kaynaklarından alınan verilerin miktarı da önemli ölçüde azaltılabilir. Bu, harici veritabanları ve ağ trafiği üzerindeki iş yükünü azaltır. Ayrıca, birkaç Qlik Sense kodunun aynı verileri paylaştığı durumlarda, verilerin kaynak veritabanından bir QVD dosyasına bir kez yüklenmesi gerekir. Diğer uygulamalar aynı verileri bu QVD dosyasıyla kullanabilir.

- Birden çok Qlik Sense uygulamasına ait verileri birleştirme.

  Binary kod deyimiyle, tek bir Qlik Sense uygulamasından başka bir tanesine veri yüklemek mümkündür; ancak QVD dosyalarıyla, bir Qlik Sense kodu herhangi bir sayıda Qlik Sense uygulamasına ait verileri birleştirebilir. Bu, farklı iş birimlerine ait benzer verileri birleştiren uygulamalar için yeni olasılıkların önünü açar.

- Artışlı yükleme

  Yaygın olarak görülen birçok durumda QVD fonksiyonu, büyümekte olan bir veritabanından özel olarak yeni kayıtlar yüklenerek artışlı yüklemeyi kolaylaştırmak için kullanılabilmektedir.

Qlik Topluluğu'nun, QVD yükleme süresini kısaltmak amacıyla Qlik Application Automation uygulamasını nasıl kullandığını görmek için bkz.: Yeniden yüklemeleri iyileştirmek için otomasyon kullanarak QVD'leri bölme.

6.1 QVD dosyaları oluşturma

QVD dosyası iki şekilde oluşturulabilir:

- Qlik Sense kodundaki Store komutunu kullanarak açık oluşturma ve adlandırma.
  Kod içinde önceden okunan bir tablonun veya bunun bir parçasının seçtiğiniz bir konumdaki açıkça adlandırılmış bir dosyaya dışa aktarılmasını belirtin.
- Koddan otomatik oluşturma ve bakım.
  Qlik Sense bir load veya select deyiminden önce Buffer ön ekini kullanarak, belirli koşullar altında veriler yeniden yüklenirken orijinal veri kaynağının yerine kullanılabilecek bir QVD dosyasını otomatik olarak oluşturur.

Sonuç olarak elde edilen QVD dosyaları arasında örneğin okuma hızı açısından bir fark yoktur.

Store

Bu kod ifadesi, adları QVD, CSV veya txt olan bir dosya oluşturur.

Söz Dizimi:

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

Deyim yalnızca bir veri tablosundan alanları dışa aktarabilir. Birkaç tablodan alanlar dışa aktarılacaksa, dışa aktarılması gereken veri tablosunu oluşturmak için kodda önceden açık bir join yapılmalıdır.

Metin değerleri, CSV dosyasına UTF-8 biçiminde dışa aktarılır. Bir sınırlayıcı belirtilebilir, bkz. **LOAD**. Bir CSV dosyasına yönelik store deyimi BIFF dışa aktarımı desteklemez.

Örnekler:

```
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';
```

Aşağıdakileri yapın:

1. *Gelişmiş Kodlama Eğitimi* uygulamasını açın.
2. *Product* kod bölümüne tıklayın.
3. Kodun sonuna şunu ekleyin:
   ```
   Store * from Product into [lib://AttachedFiles/ProductData.qvd](qvd);
   ```

   Kodunuz şöyle görünmelidir:

   ```
   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. **Verileri yükle**'ye tıklayın.
   Product.qvd dosyası artık dosya listesinde olmalıdır.

   Bu veri dosyası, **Crosstable** kodunun sonucudur ve her bir kategori (Product, Month, Sales) için bir sütun olmak üzere üç sütunlu bir tablodur. Bu veri dosyası, artık tüm *Product* kod bölümünü değiştirmek için kullanılabilir.

6.2 QVD dosyalarından veri okuma

Aşağıdaki yöntemlerle, QVD dosyası Qlik Sense uygulamasına okunabilir veya uygulama tarafından bu dosyaya erişilebilir:

- QVD dosyasını açık veri kaynağı olarak yükleme. QVD dosyalarına, tüm diğer metin dosyası türlerinde (csv, fix, dif, biff, vb.) olduğu gibi Qlik Sense kodundaki bir load deyimi ile referansta bulunulabilir.

  Örnekler:

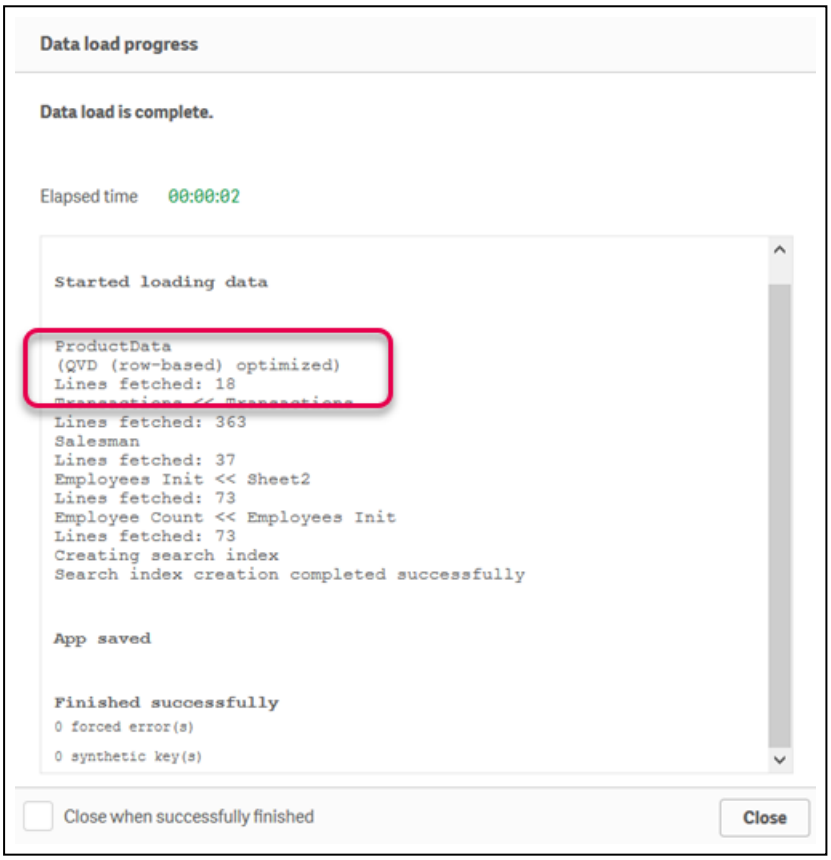
  ```
  LOAD * from 'lib://FolderConnection/xyz.qvd' (qvd);
  LOAD fieldname1, fieldname2 from [lib://FolderConnection/xyz.qvd] (qvd);
  LOAD fieldname1 as newfieldname1, fieldname2 as newfieldname2 from
  [lib://AttachedFiles/xyz.qvd](qvd);
  ```

- Arabelleğe alınan QVD dosyalarını otomatik olarak yükleme. Buffer ön eki load veya select deyimlerinde kullanıldığında, okumaya yönelik açık deyimler gerekli değildir. Qlik Sense, orijinal LOAD veya SELECT deyimiyle veri almaya kıyasla, QVD dosyasına ait verileri ne ölçüde kullanacağını belirler.
- QVD dosyalarına koddan erişme. Birkaç kod fonksiyonu (tümü QVD ile başlayan), QVD dosyasının XML üst bilgisinde bulunan verilerle ilgili çeşitli bilgileri almak için kullanılabilir.

Aşağıdakileri yapın:

1. *Product* kod bölümündeki kodun tamamını derleme dışında bırakın.
2. Aşağıdaki kodu girin:
   ```
   Load * from [lib://AttachedFiles/ProductData.qvd](qvd);
   ```
3. **Verileri yükle**'ye tıklayın.
   Veriler QVD dosyasından yüklenir.

   Veri yükleme ilerleme penceresi



Artışlı yüklemeler için QVD dosyalarını kullanma hakkında bilgi almak için Qlik Community bölümündeki blog gönderisine göz atın: Overview of Qlik Incremental Loading (Qlik Artışlı Yüklemelere Genel Bakış)

Buffer

QVD dosyaları Buffer ön ekiyle otomatik olarak oluşturulabilir ve korunabilir. Bu ön ek, koddaki çoğu LOAD ve SELECT deyiminde kullanılabilir. QVD dosyalarının deyimin sonucunu önbelleğe/arabelleğe almak için kullanıldığını belirtir.

Söz Dizimi:

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

Bir seçenek kullanılmazsa, kodun ilk yürütülmesiyle oluşturulan QVD belleği süresiz olarak kullanılır.

Örnek:

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

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

Amount, zaman dönemini belirten bir sayıdır. Decimals kullanılabilir. Atlandığında birimin günler olduğu varsayılır.

stale after seçeneği, orijinal verilerde basit bir zaman damgasının bulunmadığı durumlarda tipik olarak veritabanı kaynaklarıyla kullanılır. stale after cümlesi, basit bir şekilde, QVD belleğinin oluşturulma zamanında başlayan ve sonrasında geçerli sayılmayacağı bir zaman dönemi belirtir. Bu zamandan önce QVD belleği veriler için kaynak olarak kullanılır ve bundan sonra orijinal veri kaynağı kullanılır. Bu durumda QVD bellek dosyası otomatik olarak güncelleştirilir ve yeni bir dönem başlar.

Örnek:

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

Incremental

incremental seçeneği, temel bir dosyanın yalnızca bir bölümünü okuma özelliğini etkinleştirir. Dosyanın önceki boyutu, QVD dosyasının XML üst bilgisinde depolanır. Bu, özellikle günlük dosyalarıyla kullanışlıdır. Önceki bir durumda yüklenen tüm kayıtlar QVD dosyasından okunurken, takip eden yeni kayıtlar orijinal kaynaktan okunur ve son olarak güncelleştirilmiş bir QVD dosyası oluşturulur.

incremental seçeneğinin yalnızca LOAD deyimleriyle ve metin dosyalarıyla kullanılabileceğini ve eski verilerin değiştirildiği veya silindiği durumlarda artışlı yüklemenin kullanılamayacağını unutmayın.

Örnek:

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

QVD bellekleri normalde, oluşturduğu uygulamadaki tam kod yürütme boyunca herhangi bir konumda artık kendisine referansta bulunulmadığında veya oluşturduğu uygulama artık var olmadığında kaldırılır. Belleğin içeriklerini QVD veya CSVdosyası olarak tutmak isterseniz Store deyimi kullanılmalıdır.

Aşağıdakileri yapın:

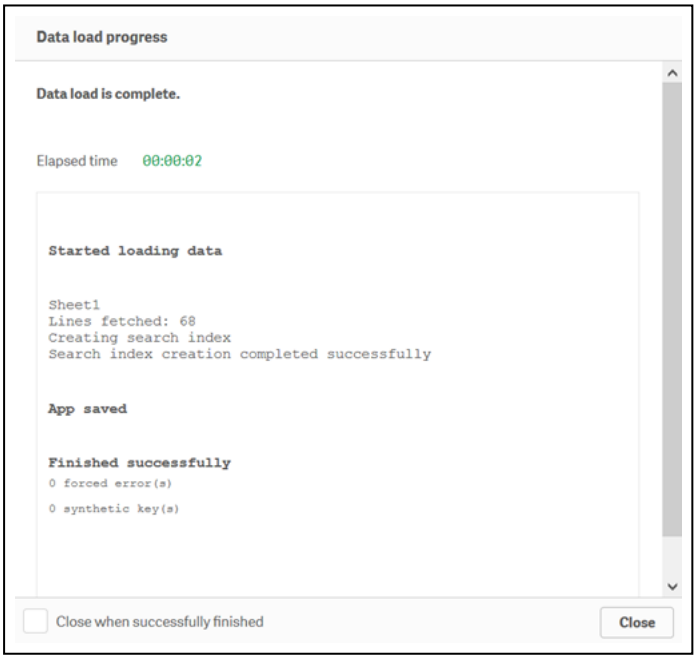
1. Yeni bir uygulama oluşturun ve ona bir ad verin.
2. **Veri yükleme düzenleyicisi**'ne yeni bir kod bölümü ekleyin.
3. Sağ menüdeki **AttachedFiles** altında **Veri seç**'e tıklayın.
4. Karşıya yükleyin ve ardından *Cutlery.xlsx* öğesini seçin.
5. **Veri seçme kaynağı:** penceresinde **Kod yerleştir**'e tıklayın.
6. Load deyimindeki alanları derleme dışında bırakın ve load deyimini şu şekilde değiştirin:
   ```
   Buffer LOAD *
   ```

   Kodunuz şöyle görünmelidir:

   ```
   Buffer LOAD *
          //     "date",
          //     item,
          //     quantity
          FROM [lib://AttachedFiles/Cutlery.xlsx]
          (ooxml, embedded labels, table is Sheet1);
   ```

7. **Verileri yükle**'ye tıklayın.
   İlk veri yüklemenizde veriler *Cutlery.xlsx* dosyasından yüklenir.

   Veri yükleme ilerleme penceresi

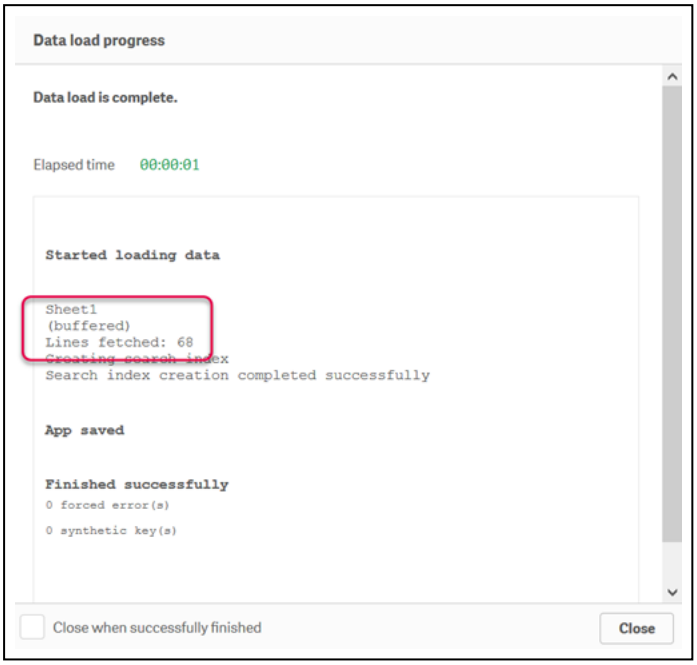
   

   Buffer deyimi de bir QVD dosyası oluşturarak Qlik Sense içinde depolar. Qlik Sense Enterprise on Windows dağıtımında Qlik Sense sunucusu üzerindeki bir dizinde depolanır.

8. Yeniden **Veri yükle**'ye tıklayın.

9. Bu sefer veriler, verileri ilk defa yüklediğinizde Buffer deyimi tarafından oluşturulan QVD dosyasından yüklenir.

   Veri yükleme ilerleme penceresi



6.3 Teşekkürler!

Bu dersi tamamladınız ve umarız Qlik Sense uygulamasında kodlama yapma hakkında daha fazla bilgi edindiniz. Diğer derslerle ilgili daha fazla bilgi için lütfen web sitemizi ziyaret edin.