



Tutorial - Scripting for Beginners

Qlik Sense®

June 2018

Copyright © 1993-2018 QlikTech International AB. All rights reserved.



Copyright © 1993-2018 QlikTech International AB. All rights reserved.

Qlik®, QlikTech®, Qlik Sense®, QlikView®, Sense® and the Qlik logo are trademarks which have been registered in multiple countries or otherwise used as trademarks by QlikTech International AB. Other trademarks referenced herein are the trademarks of their respective owners.

1 Welcome to this tutorial!	5
1.1 About this tutorial	5
1.2 Prerequisites	5
Creating a folder data connection in Qlik Sense Enterprise	6
Configuring a Microsoft Access ODBC data connection on a server	6
Placing tutorial source files in Qlik Sense Desktop	6
1.3 Further reading and resources	7
2 Introduction to data loading in Qlik Sense	8
3 Basic concepts	9
3.1 Data connection types	9
3.2 Supported file formats	9
3.3 LOAD and SELECT statements	10
3.4 ODBC	10
ODBC data connection settings	11
Adding ODBC drivers	11
64-bit and 32-bit versions of ODBC configuration	11
Creating ODBC data sources	12
4 Selecting and loading ODBC data	13
5 Selecting and loading file data	16
6 Editing the data load script	19
6.1 Accessing syntax help for commands and functions	19
Accessing the help portal	19
Using the auto-complete function	19
6.2 Inserting a prepared test script	19
Indenting code	20
6.3 Searching and replacing text	20
Searching for text	20
Replacing text	20
6.4 Commenting in the script	21
Commenting	21
Uncommenting	21
6.5 Selecting all code	22
7 Transforming data	23
7.1 Preceding LOAD	23
7.2 Resident LOAD	24
7.3 Concatenation	25
Automatic concatenation	25
Forced concatenation	26
Preventing concatenation	27
7.4 Circular references	28
Resolving circular references	29
7.5 Synthetic keys	30
Resolving synthetic keys	32

8 Debugging the load script	34
8.1 Debug toolbar	34
8.2 Output	34
9 Using data in an app	36
9.1 Creating a sheet	36
9.2 Adding a chart	36
9.3 Adding dimensions and measures	37
Creating and adding dimensions	37
Creating and adding measures	37
9.4 Thank you!	40

1 Welcome to this tutorial!

Welcome to this tutorial, which will introduce you to basic scripting in Qlik Sense. Qlik Sense is a software product that is used to present data in an intuitive and easy-to-use interface. You extract data by making selections in Qlik Sense. When you make a selection, the app immediately filters the data and presents all associated items. If you want to learn more about selections, go through the *Tutorial - Beginning with the Basics* that is available at help.qlik.com.

You use the data manager or the data load editor to load data into Qlik Sense. When you want to create, edit and run a data load script you use the data load editor. In Qlik Sense, scripting is mainly used to specify what data to load from your data sources. In this tutorial you will learn how to load data from databases and files using the data load editor, and then using them in an app.

1.1 About this tutorial

This tutorial guides you through some basic steps of scripting, using the example files provided.

These are some of the subjects in this tutorial:

- Loading data
- Editing scripts
- Transforming data

When you have completed the tutorial, you should have a fair understanding of the basic steps involved in scripting in Qlik Sense.

The screenshots in this tutorial are taken in Qlik Sense Enterprise. Some differences may occur if you are using Qlik Sense Desktop.

1.2 Prerequisites

To get the most out of this tutorial, we recommend that you have fulfilled the following prerequisites before you begin:

- You have read the ReadMe file that is included in the zip file and have followed its instructions.
- Qlik Sense Desktop is installed or you have access to Qlik Sense Enterprise.


To install Qlik Sense Desktop, follow the instructions available at help.qlik.com.

- You are familiar with the basics of Qlik Sense.

That is, you know how to make selections in an app and how to interpret the results of your selection. You have also successfully loaded data into Qlik Sense. If you are not familiar with how to do these things, you can learn all about them in the tutorial *Tutorial - Building an App*, available at help.qlik.com.

Creating a folder data connection in Qlik Sense Enterprise

If you are using Qlik Sense Enterprise you need some help from your system administrator before you can start the tutorial. You might need help to access the hub and the *Tutorials source* folder must be saved by the system administrator on the server machine. Also, before you can start loading the data files, your system administrator must prepare the data connection for you, by doing the following:

1. Save the *Tutorials source* folder on the server machine.
2. Open the hub and open an unpublished app. If there are no unpublished apps, click **Create new app**. Give the app a name, click **Create** and then click **Open app**.
3. Click  and select **Data load editor**.
4. Click **Create new connection**.
5. Select **All files**.
6. Browse to the *Tutorials source* folder stored on the server machine.
7. Type *Tutorials source* in the **Name** field.
8. Click **Create**.
9. Make sure you have access rights to use the connection. The access rights for the data connection are managed from the Qlik Management Console (QMC):
 - a. Select **Data connections** on the QMC start page.
 - b. Select the *Tutorials source* connection and click **Edit**.
 - c. Select **Security rules** under **Associated items**.
 - d. Either create a new rule or edit an existing rule to give you access to the connection.
 - e. Edit the security rule for administrative access of the data connection and click **Apply**.

The connection is now prepared for you to use.

Configuring a Microsoft Access ODBC data connection on a server

If you are using Qlik Sense Enterprise you may need some help from your system administrator to be able to follow *Selecting and loading ODBC data* (page 13). The system administrator needs to configure the data source on the server by creating a Microsoft Access DSN pointing to Sales.accdb .



For 64-bit Windows platforms, you have to enable 32-bit Microsoft Access drivers using c:\windows\sysWOW64\odbcad32.exe. The drivers can be downloaded from Microsoft if they are not installed on the system.

Placing tutorial source files in Qlik Sense Desktop

If you are using Qlik Sense Desktop, before you begin this tutorial, you need to place the *Tutorials source* folder in the *Sense* folder. The folder *Tutorials source* is included in the zip file and contains the data files.

Do the following:

1. Open the folder *Documents*. (It is sometimes called *My Documents*.) From there, the path is *Qlik\Sense*.
2. Place the *Tutorials source* folder in the *Sense* folder.

1.3 Further reading and resources

Qlik offers a wide variety of resources when you want to learn more.

At help.qlik.com, you find the Qlik Sense online help and a number of downloadable guides.

If you visit www.qlik.com, you will find the following:

- Training, including free online courses
- Demo apps
- Qlik Community, where you will find discussion forums, blogs, and more

These are all valuable sources of information and are highly recommended.

2 Introduction to data loading in Qlik Sense

Qlik Sense uses a data load script, which is managed in the data load editor, to connect to and retrieve data from various data sources. In the script, the fields and tables to load are specified. It is also possible to manipulate the data structure by using script statements and expressions. It is also possible to load data into Qlik Sense using the data manager, but when you want to create, edit and run a data load script you use the data load editor.

During the data load, Qlik Sense identifies common fields from different tables (key fields) to associate the data. The resulting data structure of the data in the app can be monitored in the data model viewer. Changes to the data structure can be achieved by renaming fields to obtain different associations between tables.

After the data has been loaded into Qlik Sense, it is stored in the app. The app is the heart of the program's functionality and it is characterized by the unrestricted manner in which data is associated, its large number of possible dimensions, its speed of analysis and its compact size. The app is held in RAM when it is open.

Before we can start looking more closely at the script in the data load editor, you need to create an empty app and a couple of data connections to be able to load data into Qlik Sense. We will go through this in detail in later sections.

The example files that you need for this tutorial are:

- *Sales.accdb*
- *Customers.xlsx*
- *Dates.xlsx*
- *Region.txt*

3 Basic concepts

In the data load editor in Qlik Sense, under the section **Data connections**, you can save shortcuts to the data sources you commonly use: databases, local files or remote files. **Data connections** lists the connections you have saved in alphabetical order. You can use the search/filter box to narrow the list down to connections with a certain name or type.

3.1 Data connection types

The following types of connections exist in Qlik Sense:

- Standard connectors:
 - **All files** connections that define a path for local or network file folders.
 - **ODBC** database connections.

ODBC (Open Database Connectivity), is a way for apps to communicate with databases. This is the most common type of data connection used in Qlik Sense.
 - **OLE DB** database connections.

OLE DB (Object Linking and Embedding, Database), is another way for apps to communicate with databases. Different types of data sources can be read using this interface, notably ODBC data sources.
 - **Web file** connections used to select data from files located on a web URL.
- Custom connectors:

Custom developed connectors for data sources are not directly supported by Qlik Sense. Custom connectors are developed using the Qlik QVX SDK or supplied by Qlik Sense or third-party developers. In a standard Qlik Sense installation you will not have any custom connectors available.

To be able to connect to a data source you first need to create and configure the connection.

3.2 Supported file formats

Qlik Sense can read in data from files in a variety of formats:

- Text files, where data in fields is separated by delimiters such as commas, tabs, or semicolons (comma-separated variable (CSV) files)
- dif files (Data Interchange Format)
- fix files (fixed record length)
- HTML tables
- Excel files
- XML files
- Qlik Sense native QVD and QVX files

3.3 LOAD and SELECT statements

You can load data into Qlik Sense using the **LOAD** and **SELECT** statements. Each of these statements generates an internal table. **LOAD** is used to load data from files, while **SELECT** is used to load data from databases.

You can also use a preceding **LOAD** to be able to manipulate the content, for example, if you want to rename some fields this has to be done in the **LOAD** statement, as the **SELECT** statement does not permit any changes.

The selection of fields from an ODBC data source or OLE DB provider is made through standard SQL **SELECT** statements. However, whether the **SELECT** statements are accepted depends on the ODBC driver or OLE DB provider used.

The following rules apply when loading data into Qlik Sense:

- Qlik Sense does not differentiate between tables generated by a **LOAD** or a **SELECT** statement. This means that if several tables are loaded, it does not matter whether the tables are loaded by **LOAD** or **SELECT** statements or by a mix of the two.
- The order of the fields in the statement or in the original table in the database is unimportant to the Qlik Sense logic.
- Field names are case sensitive, which often makes it necessary to rename fields in the script. Field names are used to identify fields and making associations.

3.4 ODBC

To access a DBMS (Database Management System) via ODBC with Qlik Sense, you have two options.

- Install an ODBC driver for the DBMS in question, and create a data source DSN. This is described in this section.
- Use the Database connectors in the Qlik ODBC Connector Package that supports the most common ODBC sources. This lets you define the data source in Qlik Sense without the need to use the Microsoft Windows **ODBC Data Source Administrator**. To connect directly to a database through one of the Qlik-licensed ODBC drivers in the ODBC Connector Package, see the instructions for Database connectors on the Qlik Connectors help site.



*The **Create new connection (ODBC)** dialog displays the **User DSN** connections that have been configured. When you are using the Qlik Sense Desktop, the list of DSN connections displays the ODBC drivers included in the ODBC Connector Package. They are identified by the "Qlik-" attached to the name (for example, Qlik-db2). These drivers cannot be used to create a new ODBC connection. They are used exclusively by the database connectors in the ODBC Connector Package. The ODBC drivers from the ODBC Connector Package are not displayed when you are using Qlik Sense in a server environment.*

The alternative is to export data from the database into a file that is readable to Qlik Sense.

ODBC data connection settings

Setting	Description
User DSN System DSN	<p>Select which type of DSN to connect to.</p> <p>For User DSN sources you need to specify if a 32-bit driver is used with Use 32-bit connection.</p> <p>System DSN connections can be filtered according to 32-bit or 64-bit.</p>
Single Sign-On	<p>You can enable Single Sign-On (SSO) when connecting to SAP HANA data sources.</p> <p>If this option is not selected, Engine service user credentials are used, unless you specify credentials in Username and Password.</p> <p>If this option is selected, Engine service user or Username / Password credentials are used to do a Windows logon, followed by a subsequent logon to SAML (SAP HANA) using current user credentials.</p>
Username	<p>User name to connect with, if required by the data source.</p> <p>Leave this field empty if you want to use Engine service user credentials, or if the data source does not require credentials.</p>
Password	<p>Password to connect with, if required by the data source.</p> <p>Leave this field empty if you want to use Engine service user credentials, or if the data source does not require credentials.</p>
Name	Name of the data connection.

Adding ODBC drivers

An ODBC driver for your DBMS (DataBase Management System) must be installed for Qlik Sense to be able to access your database. Please refer to the documentation for the DBMS that you are using for further details.

64-bit and 32-bit versions of ODBC configuration

A 64-bit version of the Microsoft Windows operating system includes the following versions of the Microsoft Open DataBase Connectivity (ODBC) Data Source Administrator tool (*Odbcad32.exe*):

- The 32-bit version of the *Odbcad32.exe* file is located in the *%systemdrive%\Windows\SysWOW64* folder.
- The 64-bit version of the *Odbcad32.exe* file is located in the *%systemdrive%\Windows\System32* folder.

Creating ODBC data sources

An ODBC data source must be created for the database you want to access. This can be done during the ODBC installation or at a later stage.



*Before you start creating data sources, a decision must be made whether the data sources should be **User DSN** or **System DSN** (recommended). You can only reach user data sources with the correct user credentials. On a server installation, typically you need to create system data sources to be able to share the data sources with other users.*

Do the following:

1. Open *Odbcad32.exe*.
2. Go to the tab **System DSN** to create a system data source.
3. Click **Add**.

The **Create New Data Source** dialog appears, showing a list of the ODBC drivers installed.


4. Select **Microsoft Access Driver (*.mdb, *.accdb)** and click **Finish**.



If you cannot find this driver in the list you can download it from Microsoft's downloads website and install it.

5. Name the data source *Scripting tutorial ODBC*.
6. Under **Database:**, click **Select....**
7. Under **Directories**, navigate to the location of your *Sales.accdb* file (a tutorial example file).
8. When the file *Sales.accdb* is visible in the text box on the left, click on it to make it the database name.
9. Click **OK** three times to close all the dialogs.

4 Selecting and loading ODBC data

The easiest way to get started loading data from a database that can be accessed through an ODBC data source, for example Microsoft Access, is by using the data selection dialog () under **Data connections** in the data load editor. However, to be able to load data from the database, you first need to connect to it.




You need to have an ODBC data source for the database you want to access. This is configured through Odbcad32.exe. If you do not have a data source already, you need to add it and configure it to point to, for example, a Microsoft Access database. See: ODBC (page 10)

Do the following:

1. Open Qlik Sense.
2. Click **Create new app**.
3. Give the app the name *Scripting tutorial* and click **Create**.
4. Click **Open app**.





*Before you load data into your app for the first time, there is an option to use **Add data** to easily load data from files. However, in this tutorial we want to see the script, so we will use the data load editor.*

5. Click  and then **Data load editor**.
6. Click **Create new connection** and select **ODBC**.
7. Select **System DSN64-bit** or **32-bit** depending on which ODBC driver you have installed and then select *Scripting tutorial ODBC*.
8. Click **Create**.

Now you can see the created data connection under **Data connections**.

The data connection has been created and you are now ready to connect to the database and to start selecting which data to load.

Do the following:

1. Add a new script section by clicking on  above the section named *Main*.
Using more than one section makes it easy to keep your script organized.
2. Give the section a name by typing *Sales* and press Enter.
3. Click  on the data connection you just created to open the data selection dialog.
This is where you select the fields to load from the database tables or views of the data source.

4 Selecting and loading ODBC data

- Under **Tables**, select *Sales data* and deselect any other tables that are selected if necessary.
By selecting *Sales data*, you are selecting all the fields in the table. Let us remove a couple of fields that we are not going to use.
- Deselect the fields *# of Days Late* and *# of Days to Ship*. You might need to click on the field headings to see the complete field names.
- Type *date* in the search field for the field names.
- Click on the heading *Invoice Date* and type *Bill Date* to rename the field and press Enter.

Select data to load

Database: C:\Users\...\
Owner: Undefined

Selections summary
C:\Users\...\\Tutorial - Scripting for Beginners\\Tutorials source\\Sales.accdb 1 table 18 columns

Tables
Filter tables
MSysAccessStorage
MSysACEs
MSysComplexColu...
MSysIMEXColumns
MSysIMEXSpecs
MSysNameMap
MSysNavPaneGroup...
MSysNavPaneGroups
MSysNavPaneGroup...
MSysNavPaneObjec...
MSysObjects
MSysQueries
MSysRelationships
MSysResources
Sales data 18

Sales data
Filter data Rows: 221053

Fields
Data preview Metadata
date

Date	Bill Date	Promised Delivery Date
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012
12/31/2012	12/31/2012	12/31/2012

```
REM LOAD BackOrder,  
Cost,  
"Customer Number",  
"Date",  
GrossSales,  
"Invoice Date" AS "Bill Date",  
"Invoice Number",  
"Item Desc",  
"Item Number",  
Margin,  
"Open Qty",  
OpenOrder,  
"Order Number",  
"Promised Delivery Date",
```

☒ Include LOAD statement


Cancel Insert script

- Click **Insert script**.
- In the inserted script, remove the word **REM**, so that it just says **LOAD** on line 3 of the script.
REM means that the lines that follow, until there is a semicolon, are remarks (comments) in the script. Comments are not executed when the script runs. By removing **REM** the lines will be executed with the rest of the script.

Now let us make an adjustment to the script to ensure that the dates are interpreted correctly.




- Add a little text to the date field on line 6, so that the script for loading this field says:
`Date#(`Date` , 'MM/DD/YYYY') as "Date",`

Now you have created the script to load the selected data from the file *Sales data.accdb*. It is time to load the data.



11. In the upper right corner, click **Load data** .

This will load the data into the app. A script execution progress window is displayed. When it is finished you will see a summary of possible errors and synthetic keys even if there are none.

12. Click **Close**.

13. Click  and select  to the right of **Data model viewer** to verify that your data is loaded. By clicking  the data model viewer will open in a new tab.

Sales data
BackOrder
Cost
Customer Number
Date
GrossSales
Bill Date
Invoice Number
Item Desc
Item Number
Margin
Open Qty
OpenOrder
Order Number
Promised Delivery Date
Sales
Sales Qty
Sales Rep Number
SalesKey

14. Select  and  to show the internal table view, that is used in this tutorial.
15. Click **Save**.

If your table is not displayed properly, you can remove the existing load script and build the script again.


5 Selecting and loading file data


Loading data from files, such as Microsoft Excel or any other supported file formats, is easily done by using the data selection dialog in the data load editor.

Do the following:


1. Open the **Data load editor**.
2. Click **Create new connection** and select **All files**.
3. Locate the folder where the tutorial source files are saved and name the connection *Scripting tutorial files*.
4. Click **Create**.

The file connection is now complete.

5. Add a new script section by clicking on  in the upper left corner.
6. Give the section a name by typing *Dates* and press Enter.

If the new section *Dates* is not already placed below *Sales*, put the cursor on the  drag bars and drag the section down below the section *Sales* to rearrange the order.

When the script is run, it is executed in the order of the sections from top to bottom

7. Click on the top row of the script and click  .
Make sure `//` is added into the script.
8. Type the following text after `//`:

Loading data from Dates.xlsx

Your script should now look like this: `//Loading data from Dates.xlsx`

You have now added a comment to the script. You can insert comments and remarks in the script code, or deactivate parts of the script code by using comment marks. All text on a line that follows to the right of `//` (two forward slashes) will be considered a comment and will not be executed when the script is run.

9. Click  on the **Scripting tutorial files** data connection to select the file to load data from.
10. Select *Dates.xlsx* and click **Select**.
11. Select *Dates*.



Under **Field names**, make sure that **Embedded field names** is selected to include the names of the table fields when you load the data.

5 Selecting and loading file data

Select data from Dates.xlsx

Hide script

Tables

Filter tables

☒ Dates5

File format

Excel (XLSX)

Field names

Embedded field names

Header size

-0+

Fields

Filter fields

<input checked="" type="checkbox"/> Date	<input checked="" type="checkbox"/> Month	<input checked="" type="checkbox"/> Quarter	<input checked="" type="checkbox"/> Week	<input checked="" type="checkbox"/> Year
1/12/2011	Jan	Q1	3	2011
1/13/2011	Jan	Q1	3	2011
1/18/2011	Jan	Q1	3	2011
1/19/2011	Jan	Q1	4	2011
1/20/2011	Jan	Q1	4	2011
1/21/2011	Jan	Q1	4	2011
1/22/2011	Jan	Q1	4	2011
1/25/2011	Jan	Q1	4	2011
1/26/2011	Jan	Q1	5	2011
1/27/2011	Jan	Q1	5	2011
1/28/2011	Jan	Q1	5	2011
1/29/2011	Jan	Q1	5	2011
2/1/2011	Feb	Q1	5	2011

LOAD

"Date",
"Month",
Quarter,
"Week",
"Year"

FROM [lib://Scripting tutorial files ██████████] Dates.xlsx
(ooxml, embedded labels, table is Dates);

Cancel

Insert script

- Click **Insert script**.
- Add the following on the row above the **LOAD** statement to name the table *Table1*:
Table1:

To ensure that the Month column in the file Dates.xlsx is interpreted correctly in Qlik Sense we need to apply the Month function to the Month field.

- Edit the **LOAD** statement so that the row "Month" now looks like this:
Month (Date) as "Month",

```
1 // Loading data from Dates.xlsx
2 Table1:
3 LOAD
4     "Date",
5     Month(Date) as "Month",
6     Quarter,
7     "Week",
8     "Year"
9 FROM [lib://Scripting tutorial files (██████████)/Dates.xlsx]
10 (ooxml, embedded labels, table is Dates);
```

Now you have created a script to load the selected data from the file *Dates.xlsx*. It is time to load the data into the app.

5 Selecting and loading file data

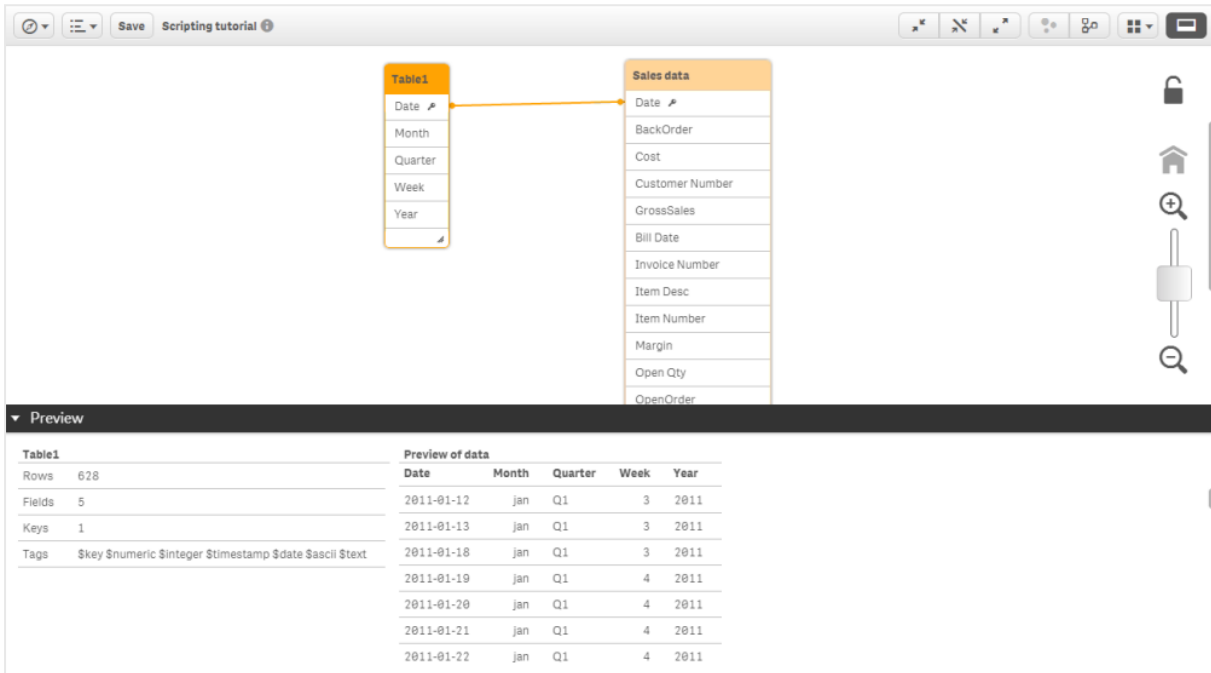
15. In the upper right corner, click **Load data**.

When you click **Load data**, the data is loaded into the app and the script is saved.

16. When the script execution is finished, click **Close**.

17. Click  and select **Data model viewer** to verify that your data is loaded.

Now you can see that a connection has been made between the two fields named *Date* in the two tables.



The screenshot shows the Qlik Sense Data Model Viewer interface. At the top, there's a toolbar with icons for navigation and actions. Below the toolbar, two tables are displayed: 'Table1' on the left and 'Sales data' on the right. An orange line connects the 'Date' field in 'Table1' to the 'Date' field in 'Sales data', indicating a relationship. The 'Table1' table has fields: Date, Month, Quarter, Week, and Year. The 'Sales data' table has fields: Date, BackOrder, Cost, Customer Number, GrossSales, Bill Date, Invoice Number, Item Desc, Item Number, Margin, Open Qty, and OpenOrder. Below the tables, there's a 'Preview' section. It contains a table with metadata for 'Table1' (Rows: 628, Fields: 5, Keys: 1, Tags: \$key \$numeric \$integer \$timestamp \$date \$ascii \$text) and a 'Preview of data' table showing the first few rows of data.

Table1	
Rows	628
Fields	5
Keys	1
Tags	\$key \$numeric \$integer \$timestamp \$date \$ascii \$text

Preview of data					
Date	Month	Quarter	Week	Year	
2011-01-12	jan	Q1	3	2011	
2011-01-13	jan	Q1	3	2011	
2011-01-18	jan	Q1	3	2011	
2011-01-19	jan	Q1	4	2011	
2011-01-20	jan	Q1	4	2011	
2011-01-21	jan	Q1	4	2011	
2011-01-22	jan	Q1	4	2011	

18. In the upper right corner, click  to show the **Preview** field. Click on the name of the table *Table1*.

This will display information about the table. In the **Preview** field you can see that 628 rows of data have been loaded into the internal table *Table1*. If you instead click on a field in the table, you will see information about the field.

6 Editing the data load script

You write the script in the text editor of the **Data load editor**. Here you can make manual changes to the **LOAD** or **SELECT** statements you have generated when selecting data, and type new script.

The script, which must be written using the Qlik Sense script syntax, is color coded to make it easy to distinguish the different elements. Comments are highlighted in green, whereas Qlik Sense syntax keywords are highlighted in blue. Each script line is numbered.


There are a number of functions available in the editor to assist you in developing the load script, and they are described in this section.

6.1 Accessing syntax help for commands and functions

There are several ways to access syntax help for a Qlik Sense syntax keyword.

Accessing the help portal

You can access detailed help in the Qlik Sense help portal in two different ways.

- Click  in the toolbar to enter syntax help mode. In syntax help mode you can click on a syntax keyword (marked in blue and underlined) to access syntax help.
- Place the cursor inside or at the end of the keyword and press Ctrl+H.



You cannot edit the script in syntax help mode.

Using the auto-complete function

If you start to type a Qlik Sense script keyword, you get an auto-complete list of matching keywords to select from. The list is narrowed down as you continue to type, and you can select from templates with suggested syntax and parameters. A tooltip displays the syntax of the function, including parameters and additional statements, as well as a link to the help portal description of the statement or function.



You can also use the keyboard shortcut Ctrl+Space to show the keyword list, and Ctrl+Shift+Space to show a tooltip.

6.2 Inserting a prepared test script

You can insert a prepared test script that will load a set of inline data fields. You can use this to quickly create a data set for test purposes.

Do the following:



- Press Ctrl + 00.

The test script code is inserted into the script.

Indenting code

You can indent the code to increase readability.

Do the following:

1. Select one or several lines to change indentation.
2. Click  to indent the text (increase indentation) or, click  to outdent the text (decrease indentation).



You can also use keyboard shortcuts:

Tab (indent)





Shift+Tab (outdent)

6.3 Searching and replacing text

You can search and replace text throughout script sections.

Searching for text

Open the data load editor. Do the following:


1. Click  in the toolbar.
The search drop-down dialog is displayed.
2. In the search box, type the text you want to find.
The search results are highlighted in the current section of the script code. Also, the number of text instances found is indicated next to the section label.
3. You can browse through the results by clicking  and .
4. Click  in the toolbar to close the search dialog.






*Also, you can select **Search in all sections** to search in all script sections. The number of text instances found is indicated next to each section label. You can select **Match case** to make case sensitive searches.*

Replacing text

Do the following:

1. Click  in the toolbar.
The search drop-down dialog is displayed.
2. Type the text you want to find in the search box.

3. Type the replacement text in the replace box and click **Replace**.
4. Click  to find next instance of search text and do one of the following:
 - Click **Replace** to replace text.
 - Click  to find next.
5. Click  in the toolbar to close the search dialog.



*You can also click **Replace all in section** to replace all instances of the search text in the current script section. The replace function is case sensitive, and replaced text will have the case given in the replace field. A message is shown with information about how many instances that were replaced.*


6.4 Commenting in the script

You can insert comments in the script code, or deactivate parts of the script code by using comment marks. All text on a line that follows to the right of `//` (two forward slashes) will be considered a comment and will not be executed when the script is run.

The data load editor toolbar contains a shortcut for commenting or uncommenting code. The function works as a toggle. That is, if the selected code is commented out it will be commented, and vice versa.

Commenting


Do the following:

1. Select one or more lines of code that are not commented out, or place the cursor at the beginning of a line.
2. Click , or press Ctrl + K.

The selected code is now commented out.

Uncommenting

Do the following:

1. Select one or more lines of code that are commented out, or place the cursor at the beginning of a commented line.
2. Click , or press Ctrl + K.

The selected code will now be executed with the rest of the script.



There are more ways to insert comments in the script code:

- Using the **Rem** statement.
- Enclosing a section of code with `/*` and `*/`.

Example:

```
Rem This is a comment ;
```

```
/* This is a comment  
   that spans two lines */
```

```
// This is a comment as well
```

6.5 Selecting all code

You can select all code in the current script section.

Do the following:

- Press Ctrl + A.

All script code in the current section is selected.

7 Transforming data

This section introduces you to basic transformation and manipulation of data that you can perform through the data load editor before using the data in your app.

One of the advantages to data manipulation is that you can choose to load only a subset of the data from a file, such as a few chosen columns from a table, to make the data handling more efficient. You can also load the data more than once to split up the raw data into several new logical tables. It is also possible to load data from more than one source and merge it into one table in Qlik Sense.

In this section you will learn how to load data using **Resident** or preceding **LOAD**. You will also learn how to handle table concatenation, circular references, and synthetic keys.

7.1 Preceding LOAD

If you load data from a database using a **SELECT** statement, you cannot use Qlik Sense functions to interpret data in the **SELECT** statement. The solution is to add a **LOAD** statement, where you perform data transformation, above the **SELECT** statement.

Basically, it is a **LOAD** statement that loads from the **LOAD** or **SELECT** statement below, without specifying a source qualifier such as **From** or **Resident** that you would normally do. You can stack any number of **LOAD** statements this way. The statement at the bottom will be evaluated first, then the statement above that, and so on until the top statement has been evaluated.

When we loaded data from the database earlier, we selected **Include LOAD statement**. That was a preceding **LOAD**. Let us take a look in the data load editor to see what it looks like.

Do the following:

1. Open the **Data load editor**.
2. Open the script section named *Sales*. There you will see the following:

```
LIB CONNECT TO 'Scripting tutorial ODBC';
LOAD BackOrder,
Cost,
"Customer Number",
Date#(`Date`, 'MM/DD/YYYY') as "Date",
GrossSales,
"Invoice Date" as "Bill Date",
"Invoice Number",
"Item Desc",
"Item Number",
Margin,
"Open Qty",
OpenOrder,
"Order Number",
"Promised Delivery Date",
Sales,
"Sales Qty",
"Sales Rep Number",
Saleskey;
SQL SELECT BackOrder,
```

```
Cost,  
"Customer Number",  
"Date",  
GrossSales,  
"Invoice Date",  
"Invoice Number",  
"Item Desc",  
"Item Number",  
Margin,  
"Open Qty",  
openOrder,  
"Order Number",  
"Promised Delivery Date",  
Sales,  
"Sales Qty",  
"Sales Rep Number",  
Saleskey  
FROM "...\\Tutorials source\\Sales.accdb"."Sales data";
```

In the script above, you can verify how the field *Invoice Date* was renamed to *Bill Date* in the **LOAD** statement, using **as**.

You can also do transformations like these using **Resident**, but in most cases a preceding **LOAD** will be faster. In the next section you will learn when **Resident** is a better choice.

7.2 Resident LOAD

You can use the **Resident** source qualifier in a **LOAD** statement to load data from a previously loaded table. This is useful when you want to perform calculations on data loaded with a **SELECT** statement where you do not have the option to use Qlik Sense functions, such as date or numeric value handling.

There are some cases where you need to use a **Resident** load instead:

- If you want to use the **Order by** clause to sort the records before processing the **LOAD** statement.
- If you want to use any of the following prefixes, in which cases, preceding **LOAD** is not supported:
 - **Crosstable**
 - **Join**
 - **Intervalmatch**

Example:

This example is not related to the data we are loading in this tutorial. It is only used to show an example of what a **Resident** load can look like. In the example script below, the date interpretation is performed in the **Resident** load as it cannot be done in the initial **Crosstable** load.

```
PreBudget:  
Crosstable (Month, Amount, 1)  
LOAD Account,  
Jan,  
Feb,  
Mar,  
...
```



```
From Budget;  
Budget:  
Noconcatenate  
LOAD  
Account,  
Month(Date#(Month,'MMM')) as "Month",  
Amount  
Resident PreBudget;  
Drop Table PreBudget;
```

7.3 Concatenation

Concatenation is an operation that takes two tables and combines them into one.

The two tables are added to each other by stacking one on top of the other, with a column for each distinct column name. The data is not changed and the resulting table contains the same number of records as the two original tables together. Several **Concatenate** operations can be performed sequentially, so that the resulting table is concatenated from more than two tables.


Automatic concatenation

If the field names and the number of fields of two or more loaded tables are exactly the same, Qlik Sense will automatically concatenate the content of the different statements into one table.

Do the following:

1. On a new line in the script in the section *Dates*, copy and paste the **LOAD** statement for *Table1* (so that the same data is loaded twice), and give the second table the label *Table2*. Your script should end up looking like this:

```
Table1:  
LOAD  
"Date",  
Month (Date) as "Month",  
Quarter,  
"Week",  
"Year"  
FROM 'lib://Scripting tutorial files/Dates.xlsx'  
(ooxml, embedded labels, table is Dates);  
Table2:  
LOAD  
"Date",  
Month (Date) as "Month",  
Quarter,  
"Week",  
"Year"  
FROM 'lib://Scripting tutorial files/Dates.xlsx'  
(ooxml, embedded labels, table is Dates);
```

2. Click **Load data**.
3. When the script execution is finished, click **Close**.
4. Click  and select **Data model viewer** to verify that your data is loaded.

Now you can see that no new table has been created.

5. Click **Show preview** and select the table *Table1*.

The screenshot shows the Qlik Sense Scripting tutorial interface. At the top, there are two tables: **Table1** and **Sales data**. **Table1** has fields: Date, Month, Quarter, Week, and Year. **Sales data** has fields: Date, BackOrder, Cost, Customer Number, GrossSales, Bill Date, Invoice Number, Item Desc, Item Number, and Margin. A line connects the **Date** field of **Table1** to the **Date** field of **Sales data**. Below the tables is a **Preview** section for **Table1**. It shows a table with 1256 rows and 5 fields. The preview data is as follows:

Date	Month	Quarter	Week	Year
01/12/2011	jan	Q1	3	2011
01/13/2011	jan	Q1	3	2011
01/18/2011	jan	Q1	3	2011
01/19/2011	jan	Q1	4	2011
01/20/2011	jan	Q1	4	2011
01/21/2011	jan	Q1	4	2011
01/22/2011	jan	Q1	4	2011

The resulting internal table has the fields *Date*, *Month*, *Quarter*, *Week*, and *Year*. The number of records is the sum of the numbers of records in the table *Table1*. You can see in the **Preview** field that the number of rows now is 1256, twice as many as after our first load.



*The number and names of the fields must be exactly the same for automatic concatenation to take place. The order of the two **LOAD** statements is arbitrary, but the table will be given the name of the table that is loaded first.*

Forced concatenation

Even if two or more tables do not have exactly the same set of fields, it is still possible to force Qlik Sense to concatenate the two tables. This is done with the **Concatenate** prefix in the script, which concatenates a table with another named table or with the most recently created table.


Do the following:

1. Edit the **LOAD** statement for *Table2*, adding *Concatenate* and commenting out *Week*, so that it looks like this:

```
Table2:
Concatenate LOAD
  "Date",
  Month (Date) as "Month",
  Quarter,
  //"Week",
  "Year"
```

```
FROM 'lib://Scripting tutorial files/Dates.xlsx'  
(ooxml, embedded labels, table is Dates);
```

By commenting out *Week*, we make sure that the tables are not identical.

2. Click **Load data**.
3. When the script execution is finished, click **Close**.
4. Click  and select **Data model viewer** to verify that your data is loaded.

Now you can see that *Table2* has not been created.

The resulting internal table has the fields *Date*, *Month*, *Quarter*, *Week*, and *Year*. The field *Week* is still showing, because it was loaded from *Table1*. But as you can see in the **Preview** in the data model viewer, the value for *Week* from *Table2* is showing that **Non-null values** is 628, while it is 1256 for all other fields. The number of records in the resulting table is the sum of the numbers of records in *Table1* and *Table2*.



*The number and names of the fields must be exactly the same. Unless a table name of a previously loaded table is specified in the **Concatenate** statement the **Concatenate** prefix uses the last previously created table. The order of the two statements is therefore not arbitrary.*


Preventing concatenation

If the field names and the number of fields of two or more loaded tables are exactly the same, Qlik Sense will automatically concatenate the content of the different statements into one table. This can be prevented with a **NoConcatenate** statement. The table loaded with the associated **LOAD** or **SELECT** statement will then not be concatenated with the existing table.

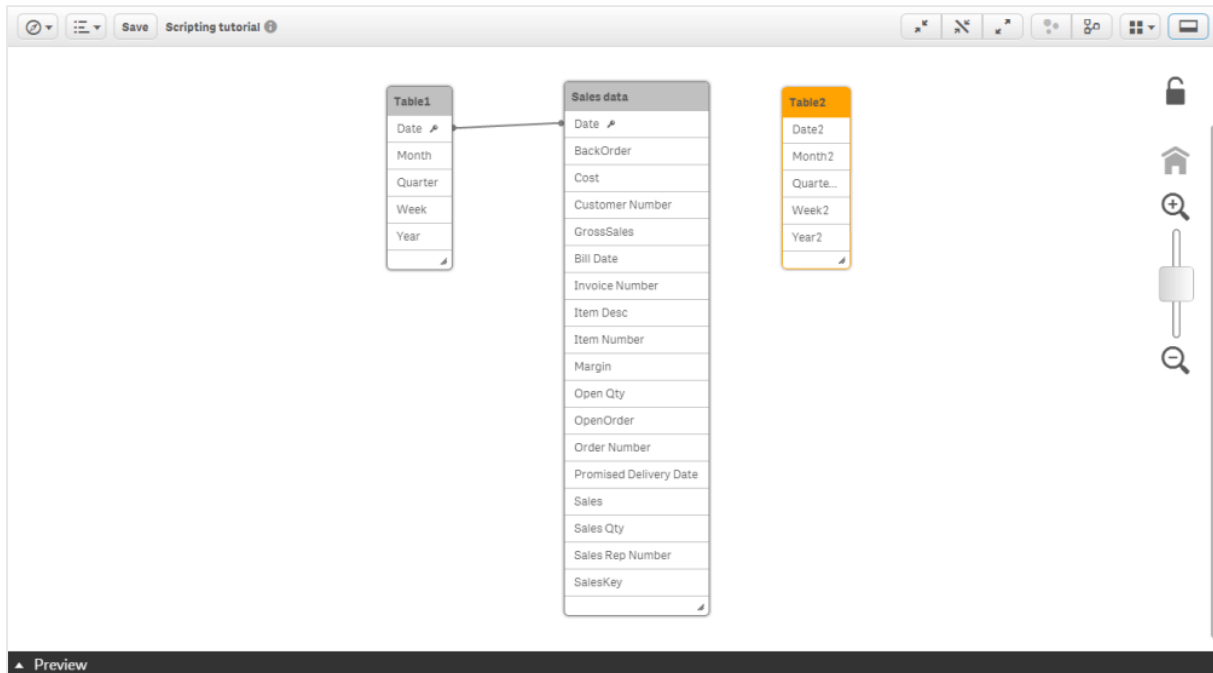
Do the following:

1. To be able to separate the content of the two tables completely, edit the **LOAD** statement and uncomment *Week* in *Table2* so that it looks like this:

```
Table2:  
NoConcatenate LOAD  
"Date" as "Date2",  
Month (Date) as "Month2",  
Quarter as "Quarter2",  
"Week" as "Week2",  
"Year" as "Year2"  
FROM 'lib://Scripting tutorial files/Dates.xlsx'  
(ooxml, embedded labels, table is Dates);
```

2. Click **Load data**.
3. When the script execution is finished, click **Close**.
4. Click  and select **Data model viewer** to verify that your data is loaded.

Now you can see that the two tables are completely separated.



Now that we have finished demonstrating concatenation, we will no longer need *Table2*.

1. Delete all the rows in the **LOAD** statement for *Table2*.
2. Click **Load data**.

It is time to take a look at another event than can occur while loading data: circular references.

7.4 Circular references

If there are circular references (loops) in a data structure, the tables are associated in such a way that there is more than one path of associations between two fields.

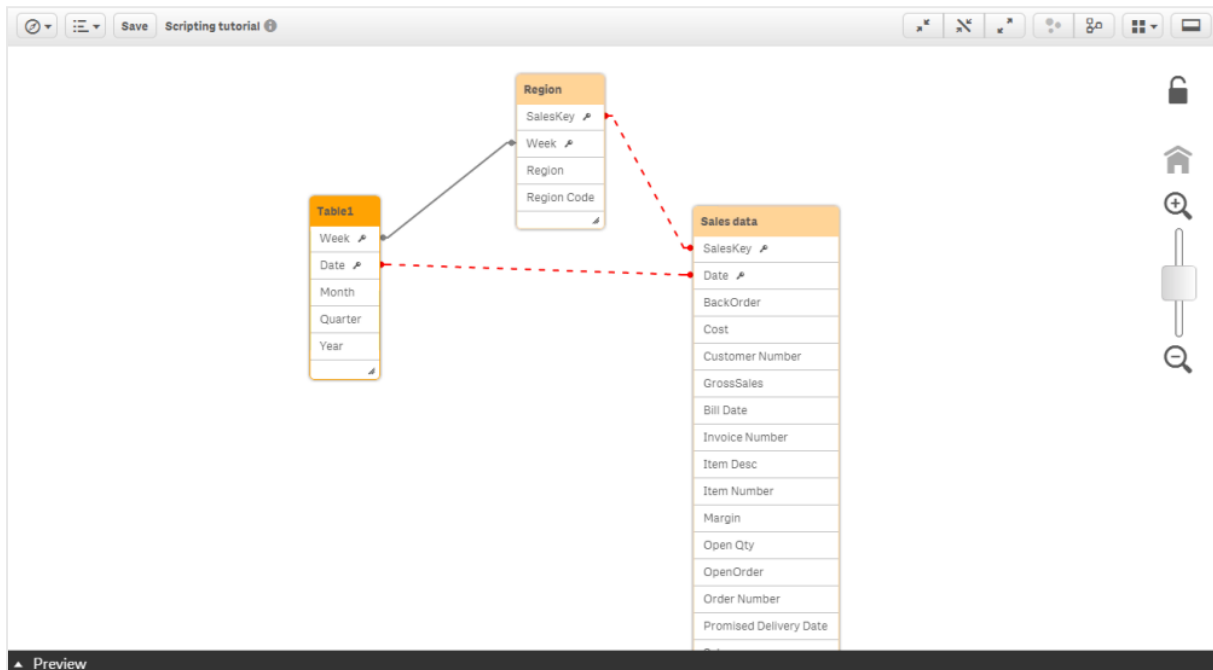
Do the following:

1. Open the **Data load editor**.
2. Add a new script section by clicking on **+** in the upper left corner.
3. Give the section a name by typing *Region* and press Enter.
4. Click **File** on the data connection *Scripting tutorial files* to select a file to load data from.
5. Select *Region.txt* and click **Select**.
6. Select all fields and make sure that **Embedded field names** under **Field names** is selected to include the names of the table fields when you load the data.
7. Click **Insert script**.
8. In the upper right corner, click **Load data** **▶**.

This time it appears that something has gone wrong with your data load. A circular reference has been created. The **Data load progress** window will show an error message, stating that a circular reference was found during the load. However, the load is completed and the app is saved.

9. Click **Close**.
10. Click  and select **Data model viewer**.

You can drag the tables around to reorganize them in a way so that the connections between the tables are easy to see.



The red dotted lines indicate that a circular reference has been created. This is something that you should avoid, because it may cause ambiguities in the data interpretation.

Resolving circular references

To be able to understand what caused the circular references, let us take a closer look at your tables in the **Data model viewer**.

If you look at the table *Table1* and the table *Sales data* in the screenshot above, you will see that they have the field *Date* in common. You can also see that the table *Sales data* and the table *Region* have the field *SalesKey* in common. Finally, notice that the tables *Table1* and *Region* have the field *Week* in common. This means that a loop, a circular reference, has been created. Since this can cause problems in the data analysis later on, let us remove it.

The easiest way to resolve this is to rename or remove one of the fields. In our case, we have loaded some data that we do not need for our app, and we can remove it.

Do the following:

1. Open the **Data load editor**.
2. Click on the section *Region* and delete the following two rows in the **LOAD** statement:
"week",
Saleskey

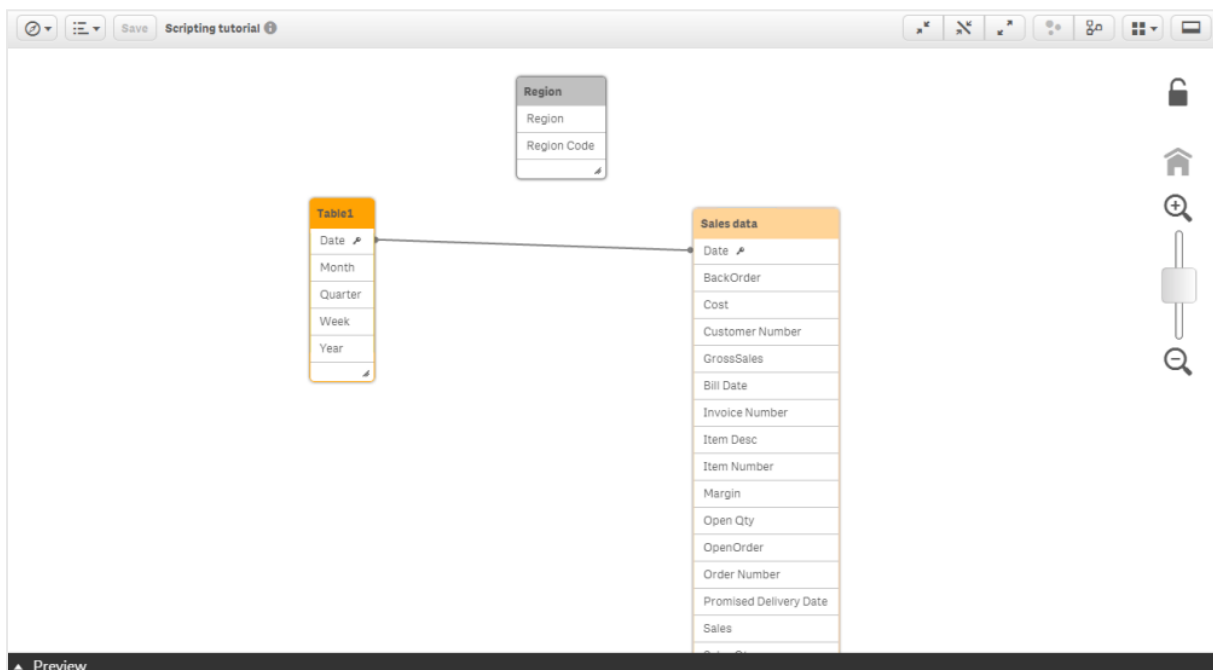
3. Make sure to also remove the comma (,) after "Region Code".

Your script in the section *Region* should now look like this:

```
LOAD
Region,
"Region Code"
FROM 'lib://Scripting tutorial files/Region.txt'
(txt, codepage is 1252, embedded labels, delimiter is '\t', msq);
```

4. Click **Load data**.
5. Click **Close**.
6. Open the **Data model viewer**.

Now the undesired references to *Region* have been removed.





7.5 Synthetic keys

When two or more internal tables have two or more fields in common, this implies a composite key relationship. Qlik Sense handles this through synthetic keys. These keys are anonymous fields that represent all occurring combinations of the composite key.

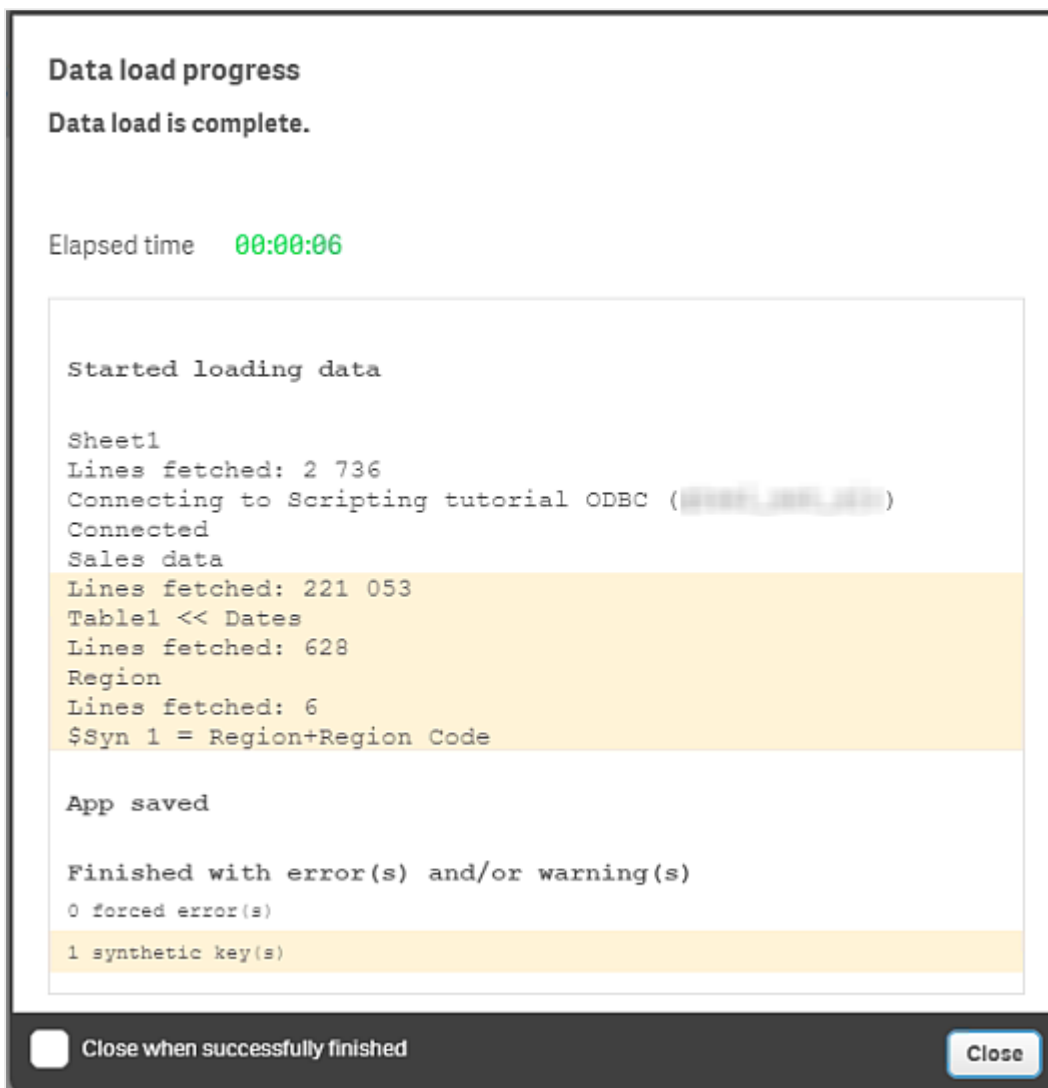
When the number of composite keys increases, depending on data amounts, table structure and other factors, Qlik Sense may or may not handle them gracefully. For example, they may affect performance and increase memory usage. Whenever there are several synthetic keys that are dependent on each other, it is good practice to remove them.

Now it is time to load our final set of data. Do the following:

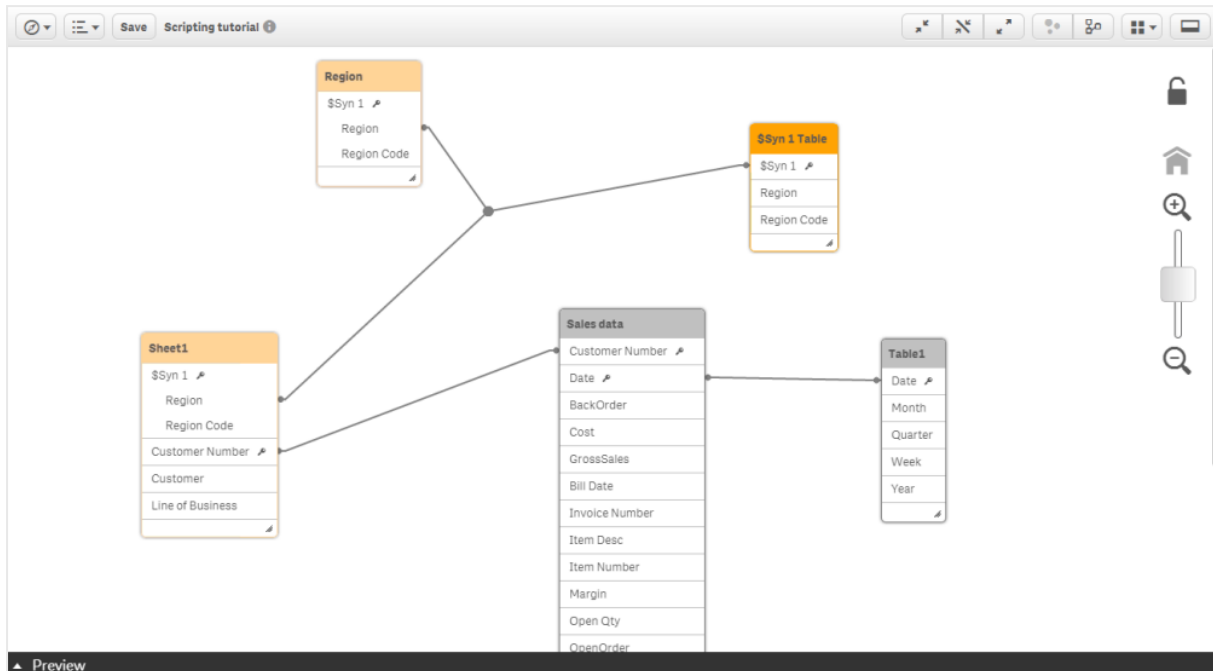
1. Open the **Data load editor**.
2. Add a new script section by clicking on **+** in the upper left corner.

3. Change the section name to *Customers* and press Enter.
4. Click  on the data connection *Scripting tutorial files* to select a file to load data from.
5. Select *Customers.xlsx* and click **Select**.
6. Select *Sheet1*.
7. Click **Insert script**.
8. Click **Load data** .

Now you can see in the data load progress window that a synthetic key has been created. Let us have a look at it.



9. Click **Close**.
10. Click  and select **Data model viewer**.



We can see that a synthetic key has been created by seeing that a new table, *\$Syn 1 Table*, has been created. It contains all the fields, *Region* and *Region code*, that the connected tables *Sheet1* and *Region* have in common. In this case it makes the connections a bit confusing and misleading, so it is not desirable to keep. Continue with the next section to find out an easy way how to remove the synthetic key.

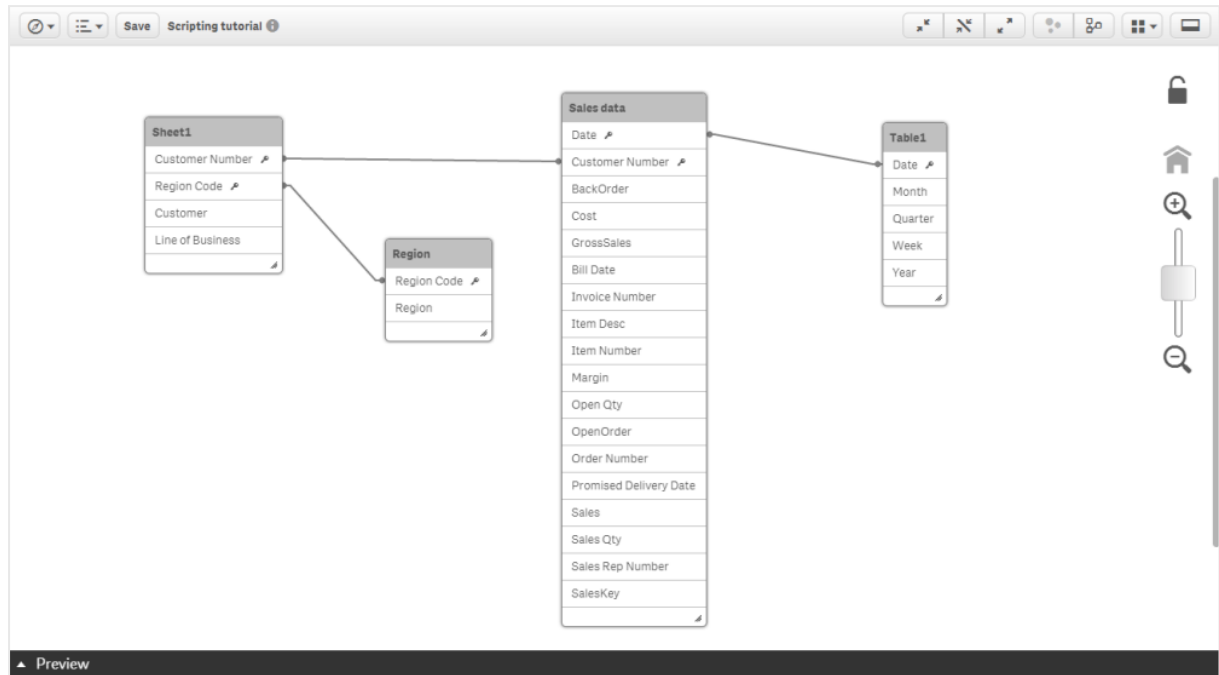
Resolving synthetic keys

The easiest way to eliminate synthetic keys is to rename one or more fields in the tables. This can be done when loading the data. Now we will go through the steps of how to get rid of a synthetic key.

Do the following:


1. Open the **Data load editor**.
 2. Click on the section *Customers* and delete the row in the **LOAD** statement saying:
Region,
 3. Click **Load data**.
 4. Click **Close**.
 5. Open the **Data model viewer**.
- Now the synthetic key has been removed.

7 Transforming data



8 Debugging the load script

To show the debug panel, do the following:





- Click  in the data load editor toolbar.
The debug panel is opened at the bottom of the data load editor.




You cannot create connections, edit connections, select data, save the script or load new data while you are running in debug mode, that is, from when you have started debug execution until the script is executed or execution has been ended.


8.1 Debug toolbar

The debug panel for the data load editor has a toolbar with the following options to control the debug execution:

Limited load	<p>Select the check box to limit how many rows of data to load from each data source. This is useful for large data sources, as it reduces the execution time.</p> <p>Enter the number of rows you want to limit the load to.</p> <div>  <i>This only applies to physical data sources. Automatically-generated and inline loads will not be limited, for example.</i> </div>
	Start or continue execution in debug mode until the next breakpoint is reached.
	Step to the next line of code.
	End execution here.

8.2 Output

Output displays all messages that are generated during debug execution. You can select to lock the output from scrolling when new messages are displayed by clicking .

Additionally, the output menu () contains the following options:

Clear	Click this to delete all output messages.
Select all text	Click this to select all output messages.
Scroll to bottom	Click this to scroll to the last output message.





Variables and Breakpoints will not be handled in this tutorial.




9 Using data in an app

To finish off this tutorial, it is time for you to put your loaded data into a visualization in your app.

9.1 Creating a sheet

Do the following:


1. Click on  and select **App overview**.
2. Click **Create new sheet** and name the sheet *Product sales* and press Enter.
3. Click the sheet *Product sales* to open it.
4. Click  **Edit** to start editing the sheet.

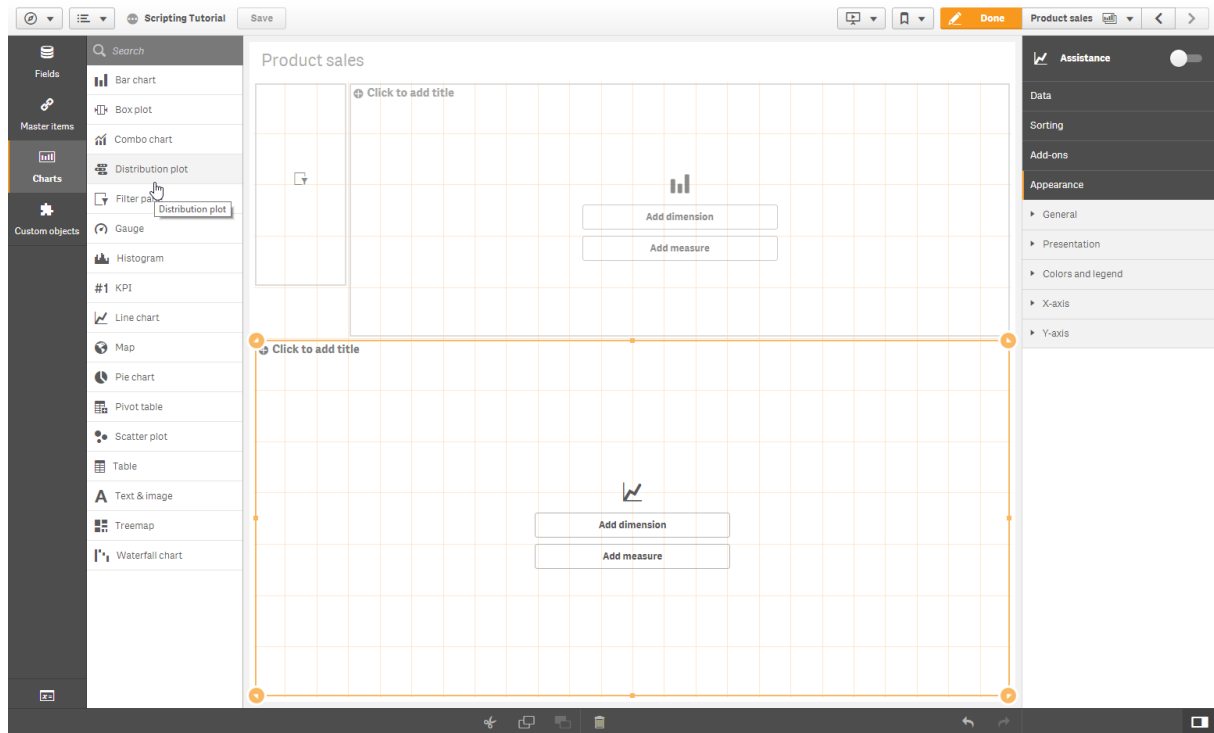
This will open the assets panel on the left and, as soon as you add a chart, the properties panel on the right opens. The assets panel has four tabs:  (**Charts**), **Custom objects**,  (**Master items**) and  (**Fields**). **Charts** is open so that you can start adding charts to the sheet.

9.2 Adding a chart

Adding a chart is the first step towards a visualization. It is not until you have also added the required dimensions and measures that a visualization is complete. You will start by adding the charts and then continue by adding dimensions and measures.

Do the following:

1. Drag a filter pane from the **Charts**  tab to the sheet and use the handles to resize it so that it is 3 cells wide and 3 cells tall. Place it in the top left corner of the sheet.
2. Drag a bar chart to the top right corner, make it 5 cells tall and wide enough to stretch until the side of the sheet.
3. Drag a line chart to the remaining space.




The icons on the sheet show what sort of chart that you have added. Now you can add dimensions and measures to your charts to complete them as visualizations.

9.3 Adding dimensions and measures

The next step is to add dimensions and measures. Begin by adding time dimensions to the top left filter pane. The benefit of a filter pane is that you save space. Instead of having one filter pane each for *Year*, *Quarter*, *Month*, and *Week*, you will use only one filter pane for the same purpose.

Creating and adding dimensions

Do the following:

1. On top of the assets panel to the left, click  to open **Fields**. Here you find all the fields in all the tables that you have loaded in the data load editor.
2. Scroll down to the bottom of the list, and click the field *Year*. Drag it to the center of the top left filter pane.
3. In the same way, add *Quarter*, *Month* and *Week* to the filter pane.


You have created a filter pane with four dimensions: *Year*, *Quarter*, *Month*, and *Week*.

Creating and adding measures

Most visualizations need both dimensions and measures. A measure is the result of an aggregation expression, which is, in many cases, a common function, such as **Sum**, **Max**, **Min**, **Avg** (average), or **Count**.

In the bar chart you will show the sales by region.

Do the following:


1. Open **Fields**  .
2. Click the field *Region* and drag it to the center of the bar chart area.
3. Click **Add" Region"**.
4. Click the field *Sales* and drag it to the center of the bar chart area.
5. Click **Add as measure > Sum(Sales)**.
6. In the properties panel on the right-hand side, click **Appearance** and then **Presentation**. Select **Horizontal**.
The bars are now displayed horizontally.
7. In the properties panel on the right-hand side, click **Sorting**.
The sorting order is displayed.
8. Drag *Sum([Sales])* above *Region* so that the dimensions are sorted by *Sum([Sales])* (measure value) instead of *Region* (dimension value, alphabetically).

The bar chart is complete, showing the sales results for the different regions. This is a basic bar chart. There are many options to enhance it in the properties panel (to the right). Just to show you one of the possibilities, let us use the title area for something more than just a title.

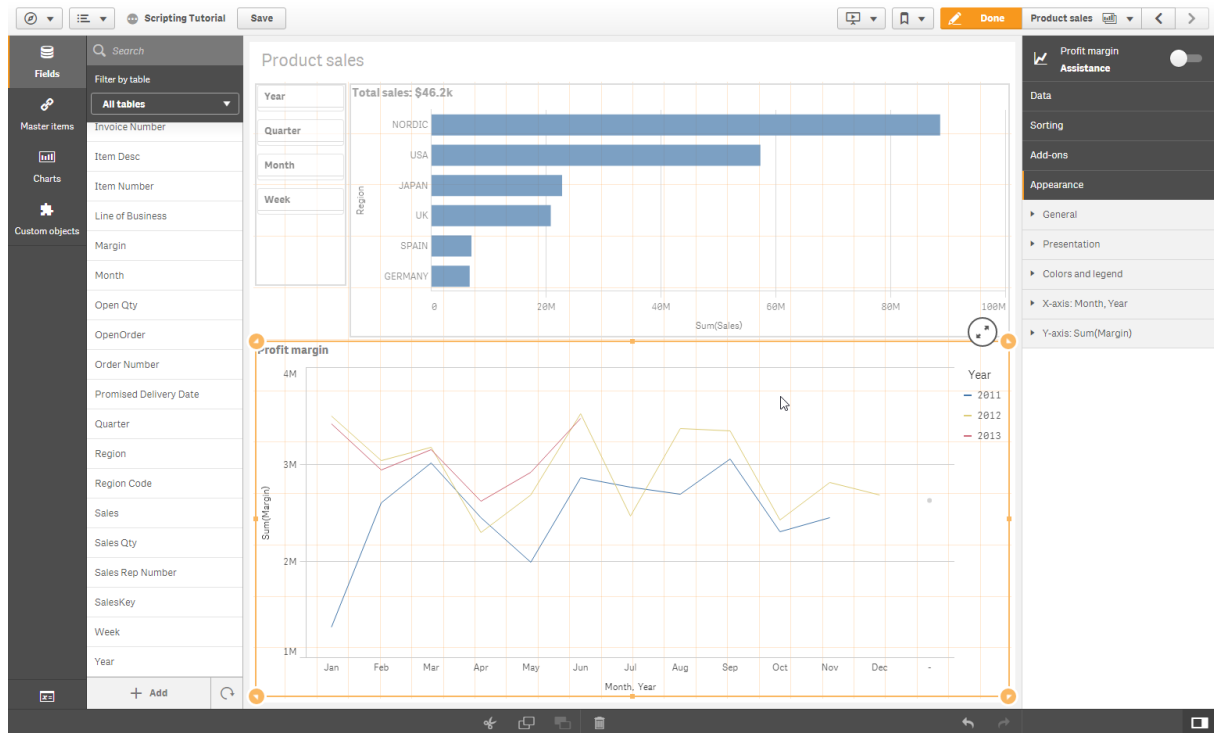
9. Copy and paste the following string, exactly as it is, into the title field of the bar chart:
`= 'Total sales: $' & Round(Sum(BackOrder)/1000, 0.1) & 'k'`
10. Press Enter.


The final visualization on this sheet is the line chart.

Do the following:

1. Open **Fields**  .
2. Click the field *Month* and drag it to the center of the line chart area.
3. Click **Add" Month"**.
4. Click the field *Year* and drag it to the center of the line chart area.
5. Click **Add" Year"**.
6. Click the field *Margin* and drag it to the center of the line chart area.
7. Click **Add as measure > Sum(Margin)**.
8. Add the title *Profit margin* on top of the line chart.

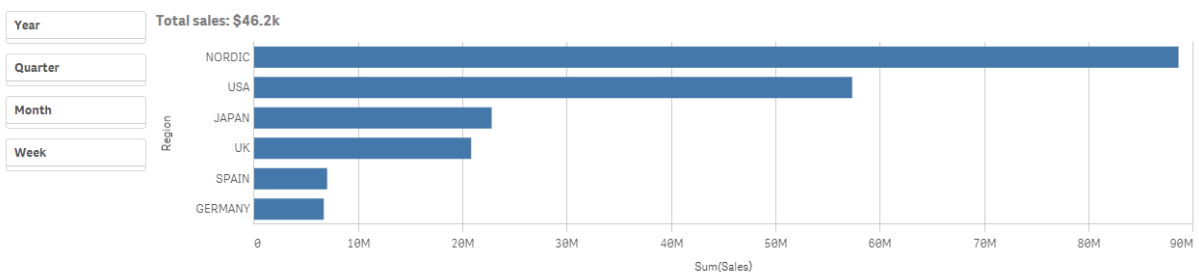
9 Using data in an app



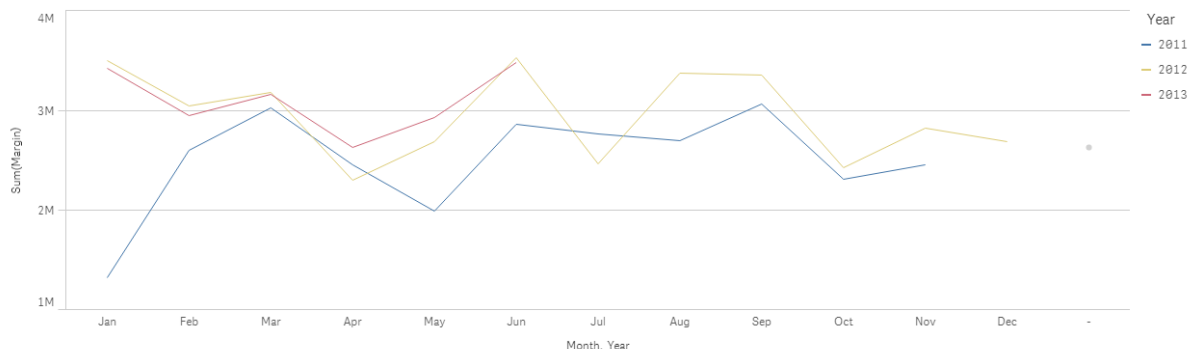
9. Click  **Done** to stop editing the sheet.

The sheet is now complete and you can begin to click around and interact with the contents of the sheet.

Product sales



Profit margin



9.4 Thank you!

Now you have finished this tutorial, and hopefully you have gained some basic knowledge about scripting in Qlik Sense. Please visit our website for more inspiration for your apps.