Fatih Çelebi

Assist. Prof. dr. Uroš Čibej

A short report about Scripting with Qlik Sense.

06 August 2021

SCRIPTING WITH QLIK SENSE

Qlik Sense command line syntax and script syntax are explained in a notation called Backus-Naur Formalism (or BNF). Backus-Naur formalism, named after the pioneers of computer programming, John Backus and Peter Naur, is a set of spelling rules used to describe the spelling rules of formal languages. Before you can start visualizing data with Qlik Sense, data must be loaded into the application. This can be done using the data manager or the data load editor. The data load editor is used to create, edit and run a data load script. This report covers loading data using scripts, editing scripts, and transforming data.

WHEN TO USE SCRİPTING ON QLIK SENSE?

The fields and tables to load are specified in the script. Scripting is often used to specify which data to load from data sources. Script statements are used to reducing data , transforming data , and manipulating the data structure.

LOADING DATA USING SCRIPTING

The LOAD and SELECT statements are used to create an internal table using the script editor. The SELECT statement is used to load data from databases, while LOAD is used to load data from files. The LOAD statement must be used to rename the preceding fields. No changes can be made to the field names with the SELECT statement.

There is no distinction between tables created using the LOAD or SELECT statements. If multiple tables are being loaded, it doesn't matter whether the tables are loaded with LOAD or SELECT statements or a mix of the two.

Field names are case sensitive and are used to establish associations between tables. Sometimes it is necessary to rename fields in the load script to achieve a desired data model.

RENAMING FIELDS USING SCRIPTING

Loaded table field names can be changed directly using script. Renaming of fields is done using the AS keyword.

Change GrossSales, to:

GrossSales AS "Gross Sales"

REDUCING DATA USING SCRIPTING

There are several different ways to reduce the amount of data uploaded. For example, data can be loaded by filtering from files or data connectors. Data can also be reduced directly in the load script.

Where Sales > 100.00 and Date("Invoice Date") >= '01/01/2013';

This only loads data when sales are greater than $100.00. It also uses the Date function to load data where the date is greater than or equal to January 1, 2013. This ensures that only the data of interest is loaded, not all of the data.

TRANSFORMING DATA USING SCRIPTING

There are many ways to transform and manipulate data using the data load editor.

Choosing to load only a subset of the data in the file, such as a column selected from a table, can make data manipulation more efficient.

This report will cover some basic data transformations using Resident load and Previous load.

Transforming Data with Resident LOAD

Data can be loaded from a preloaded table using the Resident source qualifier in the LOAD statement. This is also useful when you want to perform calculations on data loaded with a SELECT statement without the option to use functions such as date or numeric value manipulation.

Transforming Data with Preceding LOAD

This allows you to perform transformations and apply filters so that you can load the data all at once. Basically, it's a LOAD statement loaded from a LOAD or SELECT statement without specifying a source qualifier to do normally. This way, any number of LOAD statements can be stacked. This continues until the bottom expression is evaluated first, then the expression above it, and the top expression first.

CONCATENATION USING SCRIPTING

Concatenation of two tables into a single table is called Concatenation.

As a result of the concatenate, the data is not changed and the resulting table contains the same number of records as the two original tables. Two or more concatenate operations can be performed. Thus, a table that is concatenated from more than two tables is obtained.

Automatic Concatenation

If the field names and number of fields of two or more loaded tables are exactly the same, the application automatically combines the contents of different expressions into a single table. The number of fields and their names must be exactly the same for automatic concatenation to occur. The order of the two LOAD statements is optional, but the table will be named after the first loaded table.

Forced Concatenation

Even if two or more tables don't have the same set of fields, it's still possible to force two tables to concatenate. This is done with the Concatenate prefix in the script that concatenates a table with another named table or the most recently created table.

Preventing Concatenation

If the field names and the number of fields of the loaded tables are exactly the same, the contents of the different expressions are automatically combined into a single table. This can prevent the table loaded with the associated LOAD or SELECT statement from being merged with the existing table using a NoConcatenate statement.

Table contents can be completely separated by changing the names of the fields to avoid creating a synthetic key with matching field names.

ABOUT CIRCULAR REFERENCES

There can be circular references (loops) associated with multiple paths of association between two existing fields in the data structure.

An error message will be displayed in the data load editor stating that a circular reference was found during the loading of the data. However, this error does not prevent the data from being loaded.

Red dotted lines between tables in the data model viewer after the data load is performed indicate that circular reference has been established.

While circular references are not a problem when loading data, they are something to be avoided. Because it can cause ambiguities during the interpretation of the data.

Resolving Circular References

It can be seen that circular reference occurs between fields with the same field names. The easiest way to solve this problem is to change one of the fields names or completely remove the fields that are not needed.

ABOUT SYNTHETIC KEYS

A composite key relationship exists when two or more internal tables have two or more common fields. This relationship is realized through synthetic keys. All occurring combinations of the composite key are represented by these anonymous fields.

When the number of composite keys increases, they can affect performance and increase memory usage. When there are multiple dependent synthetic keys, it is good practice to remove them.

After the data load is completed in the data load editor, the data load process window displays a warning that synthetic keys have been created. In the data model viewer it can be seen that a new table starting with the field name $Syn and the synthetic key have been created. Such occurrences are undesirable to keep, as they make connections and data structure confusing.

Resolving Synthetic Keys

The easiest way to remove synthetic keys is to rename one or more fields in the tables. Or the field that caused the synthetic key to form can be completely removed.

SOURCES

1. Scripting for Beginners- www.help.qlik.com
2. Script syntax and chart functions-www.help.qlik.com
3. www.wikipedia.org
4. Qlikview Tutorial-www.tutorialspoint.com