QlikView QlikSmart

Controlling NULL Values in QlikView Script

Introduction

Nulls in QlikView are representatives of a missing value. An example of this can be illustrated when loading data from an xls file whereby an empty cell would be represented by a null within the QlikView data model. Dependent on the use case, this scenario is beneficial in enabling us to limit a chart when designing a user interface to supress where null is implicated and thus all such values are ignored. Consequently, this may also illustrate a limitation whereby null values which are forced to be visible within a chart object are not selectable. Additionally, a common data filtering object such as a list box do not show nulls - a user who wishes to calculate a specific metric value and thus visualise the results for nulls is unable to filter accordingly – a common being to monitor the quality and integrity of their data.

The remainder of the article outlines below illustrates several techniques to add value to null values within the data model which ultimately help to assist the data owner as well as identifying data quality issues. The examples used throughout this white paper are constructed using QlikView 12.0 SR5.

IsNull()

The IsNull function will test each value within a field and return 0 where the result is false and -1 for true. Using a combination of IsNull() with the if() function is a method of converting nulls to a specific value.

```
if(IsNull(Country),'n/a', Country) as Country
```

The above script will identify any null values in the field Country and replace them with the text ‘n/a’ – in the case of the result being false, the value of Country in that record will be used.

Trim() & Len()

Using the two functions Trim() & Len() force QlikView to identify blank values where the result is equal to zero – these are not null values however can be treated the same with respect to data

quality. Trim() removes any blank spaces for each record for the specified field and the number of characters which remain from the result of this will be counted using Len().

```
if(
	Len(
		Trim(Country)
		) = 0, 'n/a', Country
	) as Country
```

This script identified records of zero characters once spaces removed which are converted to ‘n/a’ – the rest will hold its value within the Country field.

NullAsValue

NullAsValue is a statement within the script used to turn on for all tables loaded after that point which replaces null values with what is specified. It can be turned off by using the NullAsNull statement. This is to be used in conjunction with a SET NullValue variable defined previous to the statement – this holds the value which will replace nulls as shown below.

```
SET NullValue = 'n/a';

NULLASVALUE Country;
```

Any nulls found beyond the above statement within the field ‘Country’ are replaced with ‘n/a’. Multiple fields can be stated within the NullValue variable and if the above script is replicated further along in the script, the value to replace nulls can be replaced as shown below.

```
SET NullValue = 'no value';

NULLASVALUE Country;
```

If a field named ‘Country’ is now found, nulls will be replaced with ‘no value’ – this is an efficient way to replace nulls with a value across multiple fields.

NullMap

A mapping table can be used as a look table for where a field holds a null value. The mapping table holds one record with two columns. The first is used to search for matching content and the second as the resulting field we replace nulls with as shown below.

```
NullMap:
Mapping LOAD
		Null() as MatchingContent,
		'n/a'  as ResultingValue
AutoGenerate 1;
```

```
LOAD
      ApplyMap('NullMap',Country,Country) as Country
From TableABC;
```

The script here loads a mapping table with one row using antogenerate. ApplyMap is then used to search for null values in the 'MatchingContent' field within the NullMap table – if a match is found within a record in the 'Country' field, it is replaced with the contents of the 'ResultingValue' field, 'n/a' – if a null is not found, the value of the 'Country' field will be used.

TIP: When loading data from a QVD, the NullAsValue method will not allow you to replace nulls – consult to using one of the other methodologies detailed in this document. If you wish to use the NullAsValue method, perform this on data that is loaded from source other than QVD.

Written by Ricky Tanna