BALAKRISHNAN, LENI (Cognizant)

Created a macro to generate straight table with large number of dimension columns

To generate straight table with large number of columns using macro

Contents

[1. INTRODUCTION 2](#_Toc436916326)

[2. PREREQUISITE 2](#_Toc436916327)

[3. MACRO SCRIPT 3](#_Toc436916328)

[4. HOW IT WORKS 4](#_Toc436916329)

[4.1. Method to Generate Straight Table 4](#_Toc436916330)

[4.2. Method to Add Dimensions 4](#_Toc436916331)

[4.3. Method to Add Expressions 5](#_Toc436916332)

[4.4. Method to Set Dimension Condition 6](#_Toc436916333)

[5. ATTACHMENT 7](#_Toc436916334)

# INTRODUCTION

While working with huge volume of data we might get a requirement to show 100+ columns in a table format with Ad-hoc functionality.

In Qlikview, we can achieve those requirements using straight table with “Enable condition” option. But the challenge is to add 100+ columns into the chart manually with enable condition. This requires more time and make our work tedious.

In order to overcome this scenario, I have written a macro to add fields and its respective “Enable Condition” automatically which helps in saving our time to implement the chart.

# PREREQUISITE

Before using this macro we need to fulfil the below mentioned points,

1. Create an excel with the dimensions (or) field names list, its respective Condition to enable the field and expressions. Add names in each column in the order you need to generate the chart.
2. In your application in which the charts to be generated do the below points,
3. Create a straight table with no dimensions and expressions in Qlikview or use the existing chart itself.
4. Create list boxes to add field name, enable condition and expression one for each.
5. Create variables “vListBoxDimSelectedCount”, “vListBoxExprSelectedCount”, “vListBoxCondSelectedCount” in “Variable Overview” which holds the expression as “GetSelectedCount(DimField)”, “GetSelectedCount(ExprField)”, “GetSelectedCount(CondField)” respectively.
6. Press “Ctrl+M” and add the script in “Macro Script”.
7. In the script do the below changes,
8. *ActiveDocument.GetSheetObject("CH02")* – Change *CH02* to the chart id in your application in which the columns needs to be added.
9. *ActiveDocument.GetSheetObject("LB01")* – Change *LB01* to the Listbox Id where the dimension field names are added.
10. *ActiveDocument.GetSheetObject("LB02")* – Change *LB02* to the Listbox Id where the expressions are added.
11. *ActiveDocument.GetSheetObject("LB03") –* Change *LB03* to the Listbox Id where the enable conditions are added.
12. Select all the fields in each listbox.
13. Once chart generated, remove all the objects & variables from the qvw file.

# MACRO SCRIPT

sub GenerateStraightTable

set ChartObj = ActiveDocument.GetSheetObject("CH02")

call AddDim(ChartObj)

end sub

sub RemoveDimExpr(ChartObj,RemoveId)

if RemoveId = 1 then

for j=0 to ChartObj.GetProperties.Dimensions.Count-1

ChartObj.RemoveDimension j

next

call AddDim(ChartObj)

elseif RemoveId = 2 then

for j=0 to ChartObj.GetProperties.Expressions.Count-1

ChartObj.RemoveExpression j

next

call AddExpr(ChartObj)

end if

end sub

sub AddDim(ChartObj)

set LBDimObj = ActiveDocument.GetSheetObject("LB01")

set LBDimCount = ActiveDocument.Variables("vListBoxDimSelectedCount").GetContent

lbDimValues = LBDimObj.GetPossibleValues

if LBDimCount.String > 0 then

if ChartObj.GetProperties.Dimensions.Count > 0 then

call RemoveDimExpr(ChartObj,1)

else

for i=0 to LBDimCount.String-1

ChartObj.AddDimension lbDimValues(i)

next

call SetEnableCond(ChartObj)

end if

end if

call AddExpr(ChartObj)

end sub

sub AddExpr(ChartObj)

set LBExprObj = ActiveDocument.GetSheetObject("LB02")

set LBExprCount = ActiveDocument.Variables("vListBoxExprSelectedCount").GetContent

lbExprValues = LBExprObj.GetPossibleValues

if LBExprCount.String > 0 then

if ChartObj.GetProperties.Expressions.Count > 0 then

call RemoveDimExpr(ChartObj,2)

else

for j=0 to LBExprCount.String-1

ChartObj.AddExpression lbExprValues(j)

next

end if

end if

end sub

sub SetEnableCond(obj)

set LBCondObj = ActiveDocument.GetSheetObject("LB03")

set LBCondCount = ActiveDocument.Variables("vListBoxCondSelectedCount").GetContent

set objProp = obj.GetProperties

lbValues = LBCondObj.GetPossibleValues

for i=0 to LBCondCount.String-1

set variable = objProp.Dimensions(i).EnableCondition

variable.Type = 2

variable.Expression = lbValues(i)

obj.SetProperties objProp

next

end sub

Using the above script we could generate a straight table.

# HOW IT WORKS

Below is the explanation of the code,

* 1. **Method to Generate Straight Table**

sub GenerateStraightTable

set ChartObj = ActiveDocument.GetSheetObject("CH02")

call AddDim(ChartObj)

end sub

The above procedure will just create an object for the chart and in turn call the procedure to add dimensions.

* 1. **Method to Add Dimensions**

**sub AddDim(ChartObj)**

**set LBDimObj = ActiveDocument.GetSheetObject("LB01")**

*This creates an object for the listbox with fieldnames which in turn used to access its functions.*

**set LBDimCount = ActiveDocument.Variables("vListBoxDimSelectedCount").GetContent**

*This gets the value from the variable vListBoxDimSelectedCount which is used to loop in the fields to add to chart.*

**lbDimValues = LBDimObj.GetPossibleValues**

*This holds the array of field names selected in the listbox which is added in the chart.*

**if LBDimCount.String > 0 then**

*Checks whether any value selected in the listbox.*

**if ChartObj.GetProperties.Dimensions.Count > 0 then**

*Here it checks whether any dimensions are already existing in the chart. If so, it will remove all the columns from the chart using the below method.*

**call RemoveDimExpr(ChartObj,1)**

*This method removes all columns from the chart. The chart ID and an indicator is passed in this method. 1 represents dimensions to be removed.*

**else**

**for i=0 to LBDimCount.String-1**

**ChartObj.AddDimension lbDimValues(i)**

*If no existing columns in straight table, this script will start add the field names from the list box to chart. Note, the name should be same as field name in application else it will fail to load.*

**next**

**call SetEnableCond(ChartObj)**

*Once all the dimensions are added, the Enable Condition method will be called which add condition to each dimension.*

**end if**

**end if**

**call AddExpr(ChartObj)**

*Once all dimensions are added, expression method is called.*

**end sub**

* 1. **Method to Add Expressions**

**sub AddExpr(ChartObj)**

**set LBExprObj = ActiveDocument.GetSheetObject("LB02")**

*This creates an object for the listbox with expressions which in turn used to access its functions.*

**set LBExprCount = ActiveDocument.Variables("vListBoxExprSelectedCount").GetContent**

*This gets the value from the variable vListBoxExprSelectedCount which is used to loop in the values to add to chart.*

**lbExprValues = LBExprObj.GetPossibleValues**

*This holds the array of expressions selected in the listbox which is added in the chart.*

**if LBExprCount.String > 0 then**

*Checks whether any value selected in the listbox.*

**if ChartObj.GetProperties.Expressions.Count > 0 then**

*Here it checks whether any expressions are already existing in the chart. If so, it will remove all the expressions from the chart using the below method.*

**call RemoveDimExpr(ChartObj,2)**

*This method removes all expressions from the chart. The chart ID and an indicator is passed in this method. 2 represents expressions to be removed.*

**else**

**for j=0 to LBExprCount.String-1**

**ChartObj.AddExpression lbExprValues(j)**

*If no existing expressions in straight table, this script will start add the expression from the list box to chart.*

**next**

**end if**

**end if**

**end sub**

## Method to Set Dimension Condition

**sub SetEnableCond(obj)**

**set LBCondObj = ActiveDocument.GetSheetObject("LB03")**

*This creates an object for the listbox with expressions which in turn used to access its functions.*

**set LBCondCount = ActiveDocument.Variables("vListBoxCondSelectedCount").GetContent**

*This gets the value from the variable vListBoxCondSelectedCount which is used to loop in the values to add to chart.*

**set objProp = obj.GetProperties**

*Here we get the properties of the chart*

**lbValues = LBCondObj.GetPossibleValues**

*This holds the array of expressions selected in the listbox which is added in the chart.*

**for i=0 to LBCondCount.String-1**

**set variable = objProp.Dimensions(i).EnableCondition**

*Creating object for each dimension column to get its properties*

**variable.Type = 2**

**variable.Expression = lbValues(i)**

*Here we are setting the condition to each column in the straight table.*

**obj.SetProperties objProp**

*This sets the properties to each column in the chart*

next

end sub

# ATTACHMENT

Here I have attached a sample qvw file which has the macro in it. Users can make use of this file to test how the macro works.



