

THE SET ANALYSIS

Summary

1	Why use the sets	3
2	The identifier.....	4
3	The operators	4
4	The modifiers.....	5
4.1	All members	5
4.2	Known members	6
4.3	Search string	7
4.4	Using a boundary for an integer dimension	8
4.5	Using a variable.....	9
4.5.1	Using a variable storing one or several members	9
4.5.2	Using a variable storing a search string	9
4.5.3	Using a variable storing integer members.....	10
4.5.4	Using a variable storing a dimension	10
4.5.5	Using a variable storing the whole set	10
4.6	Using a function.....	10
4.6.1	Numeric function	10
4.6.2	Function returning members.....	12
4.7	Using p() and e()	13
4.8	Using two fields	14
4.9	The extreme results.....	15
4.9.1	The whole application.....	15
4.9.2	The empty sets.....	15

Foreword

This document is the translation of the same document I have written in French some time ago:

<http://community.qlikview.com/docs/DOC-4889>

The 4th paragraph answers to one specific question: how do you write the set analysis when you have a variable, a search string, when you have boundaries or 2 fields.

The aim is to clarify some syntax of the Set Analysis, it is not a complete doc: a real book would be needed to cover the Set Analysis part of QlikView. But I hope it will be useful.

Enjoy ...

The version of this document is 1.0. QlikView 11.20 SR2 has been used.

Short naming convention in this doc:

XXX_KEY: dimension key

XXX_LDESC: dimension long description

XXX_SDESC: dimension short description

The TIME axis contains:

- Year, Month
- TIME_KEY, TIME_SNAME et TIME_NAME that are periods from 2010 to 2013, current month. A concatenation of years and months.

1 Why use the sets

The sets let you create a different selection than the active one being used into the chart or table. The created group let you compare the aggregations of this group and the one of the current selection.

These different aggregations let you compare for example the sales of the current month and those of the previous year, create YTD or moving totals. You may also create market shares, a percentage of the products, of the year

A set modifies the context only during the expression that uses it. Another expression without any set will get the default context, standard selection or the group of the Alternate State.

A set is always written between curly braces `{ }`. A set can modify the selection of one or several dimensions. It can be composed of a single set but can include also an inner set.

A set may be composed of:

- An identifier
- An operator
- A modifier

These 3 fields are optional. Of course, to write a set, you will need to incorporate one or several of these fields.

2 The identifier

0 – empty set

1 – complete application

\$ - current selection (take care that sometimes your chart or table does not use the current selection but an Alternate States)

\$1 – previous selection (\$2: the second previous selection, etc.)

\$_1 – next selection

Bookmark01 – name or ID of a bookmark

Group – group name (Alternate State)

Sum({Book1} [Sales]) : sales of the selection of Book1

Sum({Group1} [Sales]) : sales of the Alternate State Group1 (*the syntax is identical*)

Sum({1} [Sales]) : sum of everything (All dimensions are completely reset to All)

Sum({\$} [Sales]) : sum of the current selection (= sum([Sales])

3 The operators

+ : union of sets

*: intersection of sets

- : Exclusion of two sets

/ : members belonging to only one of the two sets

If you use the operators with the equal sign:

<Dimension += another set>: add the set to the current selection

<Dimension -= another set>: remove the set from the current selection

Sum({1-\$} [Sales]) : sum of the sales of the « database » except the current selection

Sum({GROUP1 * Book1} [Sales]) : sum if the sales for the intersection of Group1 and Book1 (the members belonging to both)

The sets that we will study are between angle brackets : <>. They are quite more complex also. But you may use also the operators:

{<set 1> -<set 2>} or **{set 1}-{set 2}**: members in the set1 that are not in the set2 (because we remove them)

4 The modifiers

Yes, that is the tricky part of the Sets Analysis ! With them, you can reduce or enlarge the scope of the selection used in an aggregation. You may modify the selection for one or several dimensions.

A set analysis is not like clicking in the ListBoxes. If you decide to set a dimension to a specific value currently not selected:

- In the ListBoxes, you will get a new selection
- In the set analysis, you will NOT: you will just set this dimension to this member (and the intersection with the current selection may be empty)

A set analysis is calculated before the chart is computed and redrawn. As you may notice, it is NOT calculated per each row/column in the chart. So, you may not guess a scope of product depending on the rows that are products. You may not guess another set of periods depending of the columns if the Period dimension is in column : the YTD or moving totals you may encounter are often valid only if the period is not a dimension of the chart.

4.1 All members

`{Starting selection <Dimension1 = {*} >} ({*} for numeric, {"*"} for text)`

Or:

`{Starting selection <Dimension1 = >}`

The starting selection may be the current one (\$ by default) or the complete application 1 or the name of a group or a bookmark (see Chapter 2 about identifiers)

If you want all members of one dimension, you can write at the right of the equal sign:

- `{*}` or `{"*"}` if the dimension is text
- nothing

As you can see below, the `{*}` can be omitted.

Ex: All MANUFACTURERS but one CATEGORY

`{<MANUFACTURER_LDESC = {"*"}, CATEGORY_LDESC={"ACC"} >}`

`{<MANUFACTURER_LDESC =, CATEGORY_LDESC={"ACC"} >}`



Take care if you have created several ListBoxes on the same logical dimension, like TIME. If you have a Listbox for Month and one for Year, please keep in mind that the user can select on both fields, and therefore limit the scope of Time by these two fields.

Check the difference between these 2 sets:

{<YEAR = {2012}>} will get the year 2012 but only for the months that have been selected in the ListBox (that is not the Total of the Year 2012)

{<MONTH=, YEAR = {2012}>} will get ALL months of the year 2012

Each time you have a hierarchical dimension and display several Listboxes, take care to reset to all every dimension except the one you want to select.

4.2 Known members

Syntax :

```
{Selection <Dimension1 = {member 1, member 2, ...} [, Dimension2 = {member 1, member 2, ...}] >}
```

Ex : {<MANUFACTURER_LDESC = {AMBOISE, BELLE, AUTREMENT}>}

Ex : {<MANUFACTURER_LDESC = {AMBOISE, BELLE, AUTREMENT}, CATEGORY_LDESC={ACC}>}

In the global syntax: `sum({<MANUFACTURER_LDESC = {AMBOISE, BELLE, AUTREMENT}>} [Value Sales])`

Please note that:

- The selection may be the current one (\$ sign), the database (1), a bookmark or a group. The current selection is the default selection, the \$ may be omitted.
- There is no comma (,) between the set and the measure name
- The dimension or the measure name must be enclosed into brackets [] if the name contains specific characters like a space, a hyphen (-) ...
- Members are always enclosed into curly brackets {} : whatever the way to find them (by name, by search string, by a function)
- Named members are separated with a comma (,)
- Numeric member names are not enclosed into single or double quotes
- Text members are enclosed into single or double quotes but not necessarily if they do not contain specific characters like spaces

Ex: {<MANUFACTURER_LDESC = {"AMBOISE", "BELLE"}, CATEGORY_LDESC={"ACC", "COLD PLATES"}>}

Ex: {<MANUFACTURER_KEY = {253, 2789, 1200}, CATEGORY_LDESC={"ACC", "COLD PLATES"}>}

If we do not write anything between the curly brackets, the set is empty:

Ex: {<MANUFACTURER_LDESC = {}, CATEGORY_LDESC={ACC} >}

Single or double quotes, square brackets

QlikView accepts the three syntaxes for text. You may write:

```
{<MANUFACTURER_LDESC = {"AMBOISE"}>}  
{<MANUFACTURER_LDESC = {'AMBOISE'}>}  
{<MANUFACTURER_LDESC = [AMBOISE] >}
```

And because there is no specific character in this member name, I may write it a 4th way:

```
{<MANUFACTURER_LDESC = {AMBOISE}>}
```

This technique is very useful when we include a function including a set into a more global set.

4.3 Search string

Sometimes, we want to get all MANUFACTURERS with a specific name or which contain specific letters in their name:

Syntax: {<Dimension = {"*search string"}>}

- * = 0 to several characters (P1* will return P1, P10, P11 ...)
- ? = 1 character (P1? will return P10, P11 ... but not P1)

We can do several search strings in the same set:

Syntaxe : {<Dimension = {"*search1*", "fixed text", " ?earch2*"}>}

Ex : {<MANUFACTURER_LDESC = {"ENT*", "?LIS*"}, CATEGORY_LDESC={"A?? "} >}

It is why we get all members of a dimension with the star: <Dimension = {"*"}> or as explained earlier by placing nothing at the right of the equal sign <Dimension =>. The two following sets are identical:

Ex: Tous les MANUFACTURERS mais sur la catégorie ACC
{<MANUFACTURER_LDESC = {"*"}, CATEGORY_LDESC={"ACC"} >}

```
{<MANUFACTURER_LDESC = , CATEGORY_LDESC={"ACC"} >}
```

Ex: All MANUFACTURERS except those whose name begins with ENT

```
{<MANUFACTURER_LDESC = {"*"} - {"ENT*"}>}
```

```
{<MANUFACTURER_LDESC = - {"ENT*"}>}
```

Ex: All MANUFACTURERS of the selection + those whose name begins with ENT

```
{<MANUFACTURER_LDESC += {"ENT*"}>}
```

Ex: All MANUFACTURERS of the selection except those whose name begins with ENT

```
{<MANUFACTURER_LDESC -= {"ENT*"}>}
```



Take care: {<MANUFACTURER_LDESC -= {"ENT*"}>} and {<MANUFACTURER_LDESC = -{"ENT*"}>}

are very different. The first set removes from the current selection all MANUFACTURERS whose name begins with ENT, while the second set selects all MANUFACTURERS except those who begin ENT : the {"*"} is by default.

Ex: All MANUFACTURERS, Alls categories whatever the selections

```
{<MANUFACTURER_LDESC = {"*"}, CATEGORY_LDESC={"*"} >}
```

Ex: All years (because it is numeric, no quote), all categories

```
{<Year = {*}, CATEGORY_LDESC={"*"} >}
```

```
%Categorie: sum([Sales])/sum({$ <CATEGORY_LDESC={"*"}>} [Sales])
```

```
%Fabricant: sum([Sales])/sum(TOTAL [Sales])
```

Fabricant = Manufacturer. Fabricant is a dimension of the tableau. Category is not a dimension: it has been set through a ListBox.

Tableau par fabricant (Ventes Volume)				
FABRICANT_LName	Somme de la sé...	Total catégories	% Catégorie	% Fabricant
AUTRES FABRICANTS	927 121	1 977 973	46,87%	53,46%
BO	10 920	71 546	15,26%	0,63%
EN	381 271	1 406 685	27,10%	21,99%
ER	409 735	426 648	96,04%	23,63%
EUR	141	141	100,00%	0,01%
FR	663	663	100,00%	0,04%
FR	451	2 758	16,35%	0,03%
GI	1 360	1 360	100,00%	0,08%
IS	2 447	2 447	100,00%	0,14%
Totals	1 734 109	3 890 221	44,58%	100,00%

QlikView accepts the single or double quotes, the square brackets also for the search string:

Ex:

```
{<MANUFACTURER_LDESC = {"E*"}, CATEGORY_LDESC={"A??"} >}
```

```
{<MANUFACTURER_LDESC = {'E*'}, CATEGORY_LDESC={'A??'} >}
```

```
{<MANUFACTURER_LDESC = {[E*]}, CATEGORY_LDESC=[A??] >}
```

4.4 Using a boundary for an integer dimension

Most of the time, Years are numeric. That is also the case for many keys because QlikView (as SQL) is faster when joining numeric keys than string keys.

The syntax is quite strange because you need to insert the code between double quotes:

```
{<TIME_KEY = {"<12"}>}: All the TIME_KEY members less than 12
```

```
{<TIME_KEY = {">=12 <=18"}>}: key members between 12 and 18, both included
```


Ex: keys 4, 5, 6 and those between 12 and 20 (excluded)
{<TIME_KEY = {4, 5, 6} + {">=12 <20"}}

4.5 Using a variable

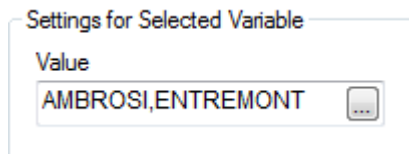
4.5.1 Using a variable storing one or several members

Syntax: <Dimension = {\$(NameVariable)} >
Or <Dimension = {\$(=NameVariable)} >

(with or without the = sign in front of the variable)

Please note that \$(=NameVariable) will interpret the variable and therefore QlikView rewrites the command before executing it.

The content of the variable may contain several members, as if they were in the real syntax. Enclose them between quotes or bracket according to their name.

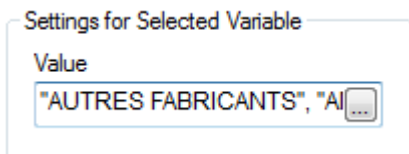


Ex :
<MANUFACTURER_LDESC = {\$(vChoiceMANUFACTURER)}, CATEGORY_LDESC={\$(=ChoiceCategory)}>



There is another \$ sign in the set analysis that means the current selection

\$(NameVariable): it can be enclosed between double quotes or not like in the example above. If there is no quote, you may add some in the text of the variable because the member names contain spaces.



4.5.2 Using a variable storing a search string

A normal set: {<MANUFACTURER_LDESC = {"E*"}, CATEGORY_LDESC={"A??"} >}

The variable vChoiceProduit stores a search string and also the double quotes "E*":
{<MANUFACTURER_LDESC = {\$(vChoixProduit)}, CATEGORY_LDESC={["A??"]} >}

4.5.3 Using a variable storing integer members

With integer keys, you need to enclose the variable name between double quotes:

```
{<TIME_KEY = {"<$(vFirstPeriod)">}}  
Or {<TIME_KEY = {"<$(=vFirstPeriod)">}}
```

The periods between two dates:

```
{<TIME_KEY = {">$(vFirstPeriod) <$(=vLastPeriod)"> >}}
```

4.5.4 Using a variable storing a dimension

We have already seen that we can use variables with the \$ sign. This variable may contain members but also a dimension:

Ex: `sum({$ <$(vDim)={"*"} >} [Volume Sales])`

Because the dimension may contain spaces or specific characters, it would be safer to use the square brackets (remember that QV rewrites the syntax, so once your dimension name with a space replace the variable, you would add brackets): `sum({$ <[$(vDim)]={"*"} >} [Volume Sales])`

4.5.5 Using a variable storing the whole set

We can store members or dimension into a variable. Why not put the whole syntax?

Ex: `sum({$ <$(vSet) >} [Volume Sales])`

Ex: `sum({$ <$(vSet), CATEGORY_LDESC= "ACC" >} [Ventes Volume])`

As you see, we can store the complete syntax or just part of it!

The **vSet** variable must contain a valid syntax except <> like :

- `MANUFACTURER_LDESC ={"*"}, TIME_SDESC={"P 01/13"}`
- `Year={$ (=max({1} Year))}, TIME_LDESC=`
- ...

4.6 Using a function

So far, we have selected members that we knew in advance they will be or not in the set. But we can also use functions that will return members according to a test, a rank

4.6.1 Numeric function

Syntax: `<Dimension = {" $(=f(args)) "}`



the \$ sign is used to expand or interpret the function

If you want to get the last year of the application, use the function max([set]) Champ)

<Year = {"\$ (=max({1} Year))"}>

Please note :

- The max function, like any aggregation function, take into account the current selection
- We have used an inner set. {1} inside the max function means the whole application. If we do not use it, the max function will return the last year among those chosen by the user
- We use **two** equal signs, 1 before the curly braces, 1 between the quotes

If you are looking for the previous year of the last year of the selection, you may write one of the two syntaxes that are equivalent:

- <Year = {"\$ (=max({\$} Year)-1)}>
- <Year = {"\$ (=max(Year)-1)}> , {\$} = current selection, it is the default

Most of the time, the set analysis become tedious, especially when writing YTD, Moving Total, Comparisons vs Year ago. One way to simplify the syntax is to populate some fields directly into the model.

Herebelow, the field A-1 stores the key for the Yr Ago period, P-1 stores the key for the previous period, YTD stores a flag (1 or 0) to indicate if the period should be taken to compute the YTD data:

TIME_KEY	MONTH	YEAR	A-1	P-1	YTD	
13	Janvier	2013	1	12	1	
14	Février	2013	2	13	1	
15	Mars	2013	3	14	1	
16	Avril	2013	4	15	1	
17	Mai	2013	5	16	1	
18	Juin	2013	6	17	1	
19	Juillet	2013	7	18	0	
20	Août	2013	8	19	0	
21	Septembre	2013	9	20	0	
22	Octobre	2013	10	21	0	
23	Novembre	2013	11	22	0	
24	Décembre	2013	12	23	0	
25	Janvier	2014	13	24	1	
26	Février	2014	14	25	1	
27	Mars	2014	15	26	1	

If you want to get the periods of the Year Ago, you just need to use the appropriate field and the **concat** function:

sum({<MONTH=, YEAR=, TIME_KEY = {\$ (=concat([A-1], ',')) } >} VALEUR)

YTD:

sum({<MONTH=, YTD = {1}, YEAR = >} VALEUR)

YTD from Year Ago:

sum({<MONTH=, YTD = {1}, YEAR = {\$ (=YEAR-1)} >} VALEUR)

MOVING TOTAL 12 periods:

sum({<MONTH=, YEAR=, TIME_KEY={ ">\$ (=Max (TIME_KEY) -12) <=\$ (=Max (TIME_KEY))" >} >} VALEUR)

MOVING TOTAL 12 periods, Year Ago:

sum({<MONTH=, YEAR=, TIME_KEY={ ">\$ (=Max (TIME_KEY) -24) <=\$ (=Max (TIME_KEY) -12)" >} >} VALEUR)



Keep in mind that the set analysis is “computed” once before the chart is computed. So, if you are looking for a set that returns different members according to the row or column of the chart, you may wait for a long while ...

Therefore, you can also modify deeper the model so that the set analysis for all the YTD, Moving total, Year Ago comparisons become very simple. See a doc I have written about this possibility:

<http://community.qlikview.com/docs/DOC-4821>

Some people populate flags (with 1 and 0) so that they can write : `Sum(VALUE*Flag)`

This is simple and works well. But it is very time consuming, much more than `sum(<Flag = {1}> VALUE)` because QlikView in the second case reduces the scope before the computation. With `sum(VALUE*Flag)`, QlikView does not scope the data and computes everything. See the excellent answer by John Witherspoon in the following discussion: <http://community.qlikview.com/message/1434>

4.6.2 Function returning members

The text function is between quotes. No \$ sign.

Syntax: `<Dimension = {"=f(parameters)"} >`

We want to aggregate the volume sales for the MANUFACTURERS whose value sales is greater than 100,000 dollars based on the current selection:

```
sum( {$ <MANUFACTURER_LDESC={"=sum([Value Sales])>100000"} >} [Volume Sales])
```

But we can also decide to search the MANUFACTURERS on a specific period, or for specific products. We will create an inner set:

```
sum( {$ <MANUFACTURER_LDESC={"=sum({1<TIME_SDESC={'P 01/13'},
CATEGORY_LDESC={'ACC','CHEESE CAKE'}>}) [Volume Sales])>100000"} >} [Volume Sales])
```

Remember that the members could be enclosed between single or double quotes. Because the function uses double quotes, we will use for the members either the single quotes or the square brackets [].

I want to find the Manufacturers whose sales are over 100,000 for the 2 categories CHEESE CAKE and ACC in January 2013 (period P01/13). But I want to remove from that list those whose sales are lower than 50,000 for all categories in January 2012:

I need two sets : {<set 1> - <set 2>}, each of them will use the function sum():

```
Set 1 = 1 <MANUFACTURER_LDESC={"=sum({1< TIME_SDESC={'P
01/13'},CATEGORY_LDESC={'ACC','CHEESE CAKE'}>}) [Value Sales])>50000"} >
```

```
Set 2 = <MANUFACTURER_LDESC={"=sum({1< TIME_SDESC={'P 01/12'},CATEGORY_LDESC={'*'}>})
[Value Sales])>100000"} >}
```

In a global syntax :

```
sum( {1 <MANUFACTURER_LDESC={"=sum({1< TIME_SDESC={'P
01/13'},CATEGORY_LDESC={'ACC','CHEESE CAKE'}>}) [Value Sales])>50000"} >
- <MANUFACTURER_LDESC={"=sum({1< TIME_SDESC={'P 01/12'},CATEGORY_LDESC={'*'}>}) [Value
Sales])<50000"} >}
[Value Sales])
```

this could be reduced a little :

```
sum( {1 <MANUFACTURER_LDESC={"=sum({1< TIME_SDESC={'P 01/13'},CATEGORY_LDESC={'ACC',
'CHEESE CAKE'}>}) [Value Sales])>50000"} -{"=sum({1< TIME_SDESC={'P
```

```
01/12'},CATEGORY_LDESC={'*'}>} [Value Sales]<50000"> } >}  
[Value Sales])
```

Please note that the function is between quotes: you will have no help to write the syntax, for the function, nor for the fields. QlikView is case sensitive for the field names: if you write TimeKey instead of TimeKEY, you will get 0. So test the function before including it in the set analysis.

If you want to get the Top 20 products:

```
Sum({<Product = {"=rank(sum(Sales), 4)<= 20">} Sales)
```

And now the Top 20 products of the brand 10 (we need to use an inner set):

```
Sum({<Product = {"=rank(sum({<Brand = {10}>} Sales), 4)<= 20">} Sales)
```



Take care NOT to write:

```
Sum({<Brand = {10}, Product = {"=rank(sum(Sales), 4)<= 20">} Sales)
```

You would get the intersection of Brand 10 and the top 20 products. And perhaps, you will get nothing because there is no product of the brand 10 among the top 20 products.



Keep also in mind that the set analysis is “computed” once, before the chart is computed. Not once per row: you will not get different products by region if you have a region dimension in your chart.

4.7 Using p() and e()

These functions returns the members that have (or have not) data. They are based on the associative model.

Syntax: {<Dimension1 = P({< {Set Dimension to select} Dimension1 }>) [, Other dimensions of the Set]>}

Please note that:

- Both functions **p()** et **e()** have the same syntax : one p() returns the possible values, the other e() returns the excluded values
- They will be inserted into a greater set
- These functions do let you search the Top N products or those whose sales are greater than X : they just return the members (ie the products) that have data, or those that have not.

Ex: we are looking for the manufacturers who sell CROISSANT in the first period of 2013. But we want to display all categories:

```
sum({$ <CATEGORY_LDESC={"*"}, MANUFACTURER_LDESC=p({1<CATEGORY_LDESC={"CROISSANT"},  
TIME_SDESC= "P 01/13">} MANUFACTURER_LDESC) >} [Sales Value])
```

4.8 Using two fields

If we want to select a dimension through two fields, for example:

- Billed client = Delivered Client
- Ordering Date = Delivery Date
- ...

The syntax is more or less the one seen 4.6.2.

Syntax: {<Dimension = {"=boolean condition">}}

The boolean condition will be created by comparing the two fields.



Attention: the searched dimension cannot be also in the boolean condition. If needed, create an integer key with **Autonumber()**.

We want to get the sales that have been delivered the same day. We have two fields : DayDelivery and DayOrder

```
Sum({<KeyAutoNumber = {"=(DayDelivery=DayOrder)" } >} Sales)
```

And we can easily get the sales with a late delivery, for example 7 days or whatever stored into a variable:

```
Sum({<KeyAutoNumber = {"=(DayDelivery<DayOrder -7)" } >} Sales)
```

```
Sum({<KeyAutoNumber = {"=( DayDelivery<DayOrder -$(vDelai))" } >} Sales)
```

You can get an equivalent to the above syntax but this is slower because QlikView needs to compute all the lines of the scope:

```
Sum(if(DayDelivery=DayOrder, Sales))
```

`if()` returns `Null()` if the third parameter has been omitted and the boolean condition is false. This syntax does not need an additional key (or field).

4.9 The extreme results

4.9.1 The whole application

You can use 1 between the braces {} : {1}

```
sum({1} [Sales Value])
```

Last year (we already studied it):

```
$(=Max({1} YEAR))
```

4.9.2 The empty sets

It can happen that the set is empty. The result will be 0. It is either something wanted, or something more unwanted:

- You use the braces without anything in it: {}
- You typed the fields or the members incorrectly: Qlikview is case sensitive
- You typed a wrong expression
- You typed a wrong intersection: a product, a client but this client does not buy the product (0 is therefore the right answer)
- You forgot to reset a field that is limited through a ListBox. For example, you have brands and products in Listbox, and the user select a product. By setting a brand in the set Analysis, you may encounter an emptiness because the chosen product does not belong to the brand. You need to reset to all the products: {<Product= , Brand = {1, 2} >}

For time periods, when we want a total year, it would safer to write in the set analysis all the dimensions that could be set through ListBoxes:

```
sum({$ <Year = {2012}, QUARTER =, MONTH=, DAY= > [Volume])
```

This time, we get the whole year, whatever the selection the user has made.

Some links:

A GUI that generates some code:

http://tools.qlikblog.at/SetAnalysisWizard/QlikView-SetAnalysis_Wizard_and_Generator.aspx?sa=

Simply create YTD, moving totals and comparison vs Year ago:

<http://community.qlikview.com/docs/DOC-4821>