Aggregated definitions

Question object

The question object is similar to what we have in the Design / routing language. You can use the shortcut directly to have access to the object: Gender.LongCaption or ^My question^.LongCaption if you have spaces in the shortcut. There are some differences: in Design the question refers to a given data point - an interview and a given loop item. Here it depends: in a sub-population type (or similar) script, the question refers to the current interview (or level) available through the Value property as in ??Gender??). In a calculation arithmetic (or similar), the object refers to all responses of the questions - available through the Data property

Properties

- **Shortcut** As String
- **LongCaption** As String
- **ShortCaption** As String
- **MinValue** As Number
- **MaxValue** As Number
- **MinDate** As Number
- **MaxDate** As Number
- **Responses** As Response Array
- **Children** as Array of Question
- **Parent** As Question

The following properties allow iterating through the structure of the survey:

- **PreviousQuestion** As Question
- **NextQuestion** As Question

Note: we cannot use Next as a property as it's a reserved keyword for For … Next

- **Index** as Number (peut etre overkill)
- **Data** As DataObject

This returns the data in the original level of the question

Note: within a tab (when using calculation arithmetic) we need to decide how this is affected by the main tab and overridden properties by calculation, question or response on Level, Sub-population, Weighting and Universe (LSWU)

- **Value** As Variant

Only available in interview type scripts

Response object

Properties

- **Caption** As String
- **Factor** As Number

Data object

Methods

- **Size()** As Number

Counts the number of interviews selected

Note: It's an easier way to get counts (in conjunction with the FilterByX(, FilterByY()... methods)

- **Counts()** As Number array

Returns the counts for all responses of that question

- **Counts(Groups as Number array)** As Number array

Returns the counts for groups of responses

Example:

q1.Data.Counts({1;2} , {4;5}) returns the top 2 and bottom 2 of a 5 scale question

- **Mean()** As Number

Returns the means

- **stdDev()** as Number

Returns the standard deviation (where the variance is divided by N)

- **StdDevEst()** As Number

Returns the standard deviation estimator (where the variance is divided by N-1)

- **Median(UseInterpolation As Number)** as Number
- **Min()** As Number
- **Max()** as Number
- **Sum()** As Number

- **Filter(NameOfSubPopulation As String)** As Data

Filters the data object according to a named sub-population

If the data is affected to a variable, the variable gets changed by the call

Note: We will need a different way of filtering responses in Tools as the sub-populations are not available

- **Weight(nameOfWeighting as String)** as Data

Weights the data according to a named weighting

- **MakeNumeric(Number* as Factor)** As Data

Transforms closed data into numeric by providing a factor for each closed response.

*If the factors are not provided, the factor defined in design are used

Examples:

gender.Data.MakeNumeric(1,2,0,0).Mean()

Income.Data.MakeNumeric().Mean()

Arithmetic table definitions

The arithmetic language is based on the aggregated language definition.

Builtins

- **CurrentCell** returns a Cell object

Returns the current cell

- **CurrentTable** returns a Table object

Returns the current table

- **GetTable (Name As string, Table As optional Number)** returns a table object

Returns a table from the current portfolio

Example:

GetTable("mysummary", 1).GetCell (4,CurrentCell.Y).GetValue

- **Calc(Index as Number)** As Number
- **Calc (Name as String)** as Number

This returns the value of the current calculation.Calc(1) replaces the old {1}

You can either use the index of the calculation or its caption

It returns DK if the index or the caption is unknown

Data Object additions

Methods

- **FilterByXY(X as Optional Number,Y as Optional Number)** As DataObject

This filters the data according to the table row and column - that is looking at the modality in row, column and edge only selecting interviews if they have given the response. If the parameter is not specified, CurrentCell.CurrentX or CurrentCell.Y is used

❖ **FilterByY(Y as Optional Number)** As DataObject

This filters the data according to the table row - that is looking at the modality in row and only selecting interviews if they have given the response. If the parameter is not specified, CurrentTable.currentY is used

❖ **FilterByX(X as Optional Number)** As DataObject

This filters the data according to the table column - that is looking at the modalities in the column AND edge and only selecting interviews if they have given the responses. If the parameter is not specified, CurrentTable.CurrentX is used

❖ **FilterByColumn(X as Optional Number)** As DataObject

This filters the data according to the column profile - that is looking at the corresponding modality in the column profile and only selecting interviews if they have given the response. If the parameter is not specified, CurrentTable.CurrentX is used

❖ **FilterByEdge(X as Optional Number)** As DataObject

This filters the data according to the edge profile - that is looking at the corresponding modalities in the edge profile and only selecting interviews if they have given the responses. If the parameter is not specified, CurrentTable.CurrentX is used

Examples:

This would give you the normalised (centered and reduced data) within the table

```
Dim dMean = Q1.Data.Mean() ‘ Overall mean
Dim dSD = Q1.Data.StdDev() ‘ Overall std dev
return (Q1.Data.FilterByXY().Mean() - dMean)/dSD
```

To get the base in row you can use:

```
Q1.Data.FilterByX().FilterSize
```

To find the highest frequency:

```
dim arrFreq = {}
Dim i
        For i=1 to CurrentTable.MaxX
            dim dCounti = CurrentCell.Row,Question.Data.FilterByXY( i ).FilterSize()
            dim dBasei = CurrentTable.RowQuestion.Data.FilterByX( i ).FilterSize()
            arrFreq.Insert( dCounti / dBasei)
        Next i
return arrFreq.Max()
```

Table Object

Properties

- **MaxX** as Number

Returns the amount of columns (in the Excel sense) in the table

- **MaxY** as Number

Returns the amount of rows (in the Excel sense) in the table

- **StartX** as Number

Returns the column (in the Excel sense) in the table where the numbers from the crosstab start (in other word the number of title columns + 1)

- **StartY** as Number

Returns the row (in the Excel sense) in the table where the numbers in the crosstab start (in other words the number of title rows +1)

Methods

- **GetCell (X as Number, Y as Number)** As Cell

Returns the content of a table using the absolute X and Y coordinates. X=1 and Y=1 indicate the top left cell in the titles

> *Note 1: Crosstabs are run, then cross stats, then flat counts and finally flat stats in that very order - this means that when the GetCell is called, Flat counts (the base of a row or a column) will not be available - maybe we need to have options to defer the run so all the tab is available*

- **FindCell(rowIndex as Number,ColIndex as Number, EdgeIndexes as Number Array)** as Cell

Finds a cell according to the indexes defined in the profile. If the cell does not exist, it will return a cell with the properties X and Y set to 0.

This is only available in the CurrentTable object (and not implemented yet)

- **GetColSigLetter(X as Number,[Strength as Number])** as String

Returns the associated col-sig letter for any given column

The strength (by default is 2) can be -3,-2,-1, 1 ,2 ,3 to indicate the strength of the significativity and the sig (1 returns a, 2 A, 3 A+, -1 -a, -2 -A, -3 -A+)

This is only available in the CurrentTable object

Cell object

Properties

- **X** as Number

Indicates the column (in the Excel sense) position of the cell. The titles are not counted so the first calculation has a X = 1

- **Y** as Number

Indicates the row (in the Excel sense) position of the cell. The titles are not counted so the first calculation has a Y = 1

- **Value** as Number

Returns the value as a number of the cell (with all significant digits)

- **Text** as String

Returns the formatted value of the cell (do not use for calculation because first it's a string then some significant digits could be hidden by the formatting)

- **Row** as ProfileItem
- **Column** As ProfileItem
- **Edges** as ProfileItems

These properties are only available in the CurrentCell / CurrentTable objects

Possible additions: (not done)

IsRowTotal as Number
IsColumnTotal
IsQuestionTotal
IsEdgeTotal
IsStat
Calc

ProfileItem object

Properties

❖ **Question** As Question

Returns the question corresponding to the profile - use with Data as in:

CurrentCell.Row.Question.Data.FilterByXY().Mean()

❖ **Index** as Number

Returns the index of the response associated before there was a row or column suppression:

Examples:

Gender
Male => 1
Female => 2

Age
Less than 24 =>3
25-45 =>4
46+ =>5

On (CurrentCell.Row.Index, 1000,2000,5000,10000)

❖ **Caption** as String

Returns the caption of the response (or group) as defined in the profile

Clean-up language definitions

The clean-up language is a superset of the arithmetics language except that CurrentCell and Calc(x) have been undefined

Cell object

Members

- **SetValue**(value As Number)

Replaces the current value by the number indicated

- **SetText**(value As Text)

Replaces the current value by the string indicated

- **SetBold**(value As optional number)

Sets the cell in bold if no parameter is specified or if the number is different than 0. Removes it if parameter is 0

- **SetItalic**(value As optional number)

Sets the cell in italic if no parameter is specified or if the number is different than 0. Removes it if parameter is 0

- **SetUnderline**(value As optional number)

Sets the cell in underline if no parameter is specified or if the number is different than 0. Removes it if parameter is 0

- **SetForeColor**(value As optional String)
- **SetForeColor**(valueRed As Number, valueGreen As Number, valueBlue As Number)

Sets the cell in the specified background color. If no parameter, restores default. When using string, you can specify “#000000” or the name of the colour like “blue”.

- **SetBackColor**(value As optional String)
- **SetBackColor**(valueRed As Number, valueGreen As Number, valueBlue As Number)

Sets the cell in the specified foreground color. If no parameter, restores default. When using string, you can specify “#000000” or the name of the colour like “blue”.