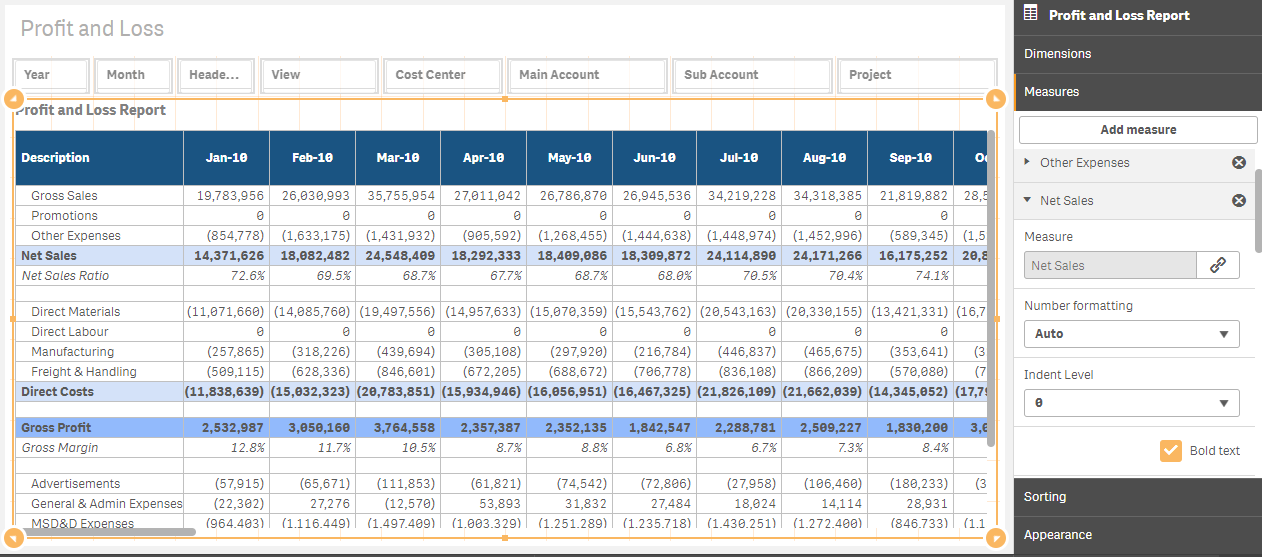
Often there is a requirement for highly structured, financial type reports to be created in Qlik. These reports are tricky to create in most BI software as they usually require different calculations on various rows. BI software (like Qlik) usually generate columns for each calculation and a dimension determines the split by rows.

There are already a couple of great extensions available on Branch which can produce these reports. These solutions generally rely on the layout of the report being represented in the data model. So row based formatting and the many calculations must be built into the data model.

This extension takes a different approach. Columns are transposed as rows and vice versa - each new calculation you add will add a new row to the report. This allows the report layout to be defined entirely through the UI by the user.

This is not meant to be a replacement for other extensions that exist. Rather it is designed to complement them, as each approach has its benefits and in combination you will hopefully have all your financial reporting needs covered.

*Most of the credit for this extension needs to go to Michael Clemens, who wrote the transposition logic. I merely adapted his work to allow for the formatting options.*



**Report Level Configuration Settings**

1. Label Column Header

This is the text that will appear in the first field of the first column, above the row descriptions.

1. Header Background color

The background color for the header row. Enter it as a hexadecimal value.

1. Header text color

The text color for the header row. Enter it as a hexadecimal value.

1. Format all values

To save you from having to format the number for every single row you can set the global format here. Row level formats will override this value for the specific row.

1. Hide decimals

To save you from having to format hide decimals for every single row you can set it globally here. This will cause each number to be rounded. Row level formats will override this value for the specific row.

1. Add totals column

Check this box to create a totals column on the end of the report. Rows with a percentage in the text (ie. ratios) will not be totaled.

**Report Dimension Settings**

The report dimension will determine the number of columns. Only one dimension can be used at this stage. There are no other special dimensional settings.

Note that when a % character is detected in one of the report dimension columns then the data in that column will be formatted as percentage regardless of other number format settings for that row. This is to allow for ratio columns like those illustrated below.

**Row Level Configuration Settings**

1. Indent Level

Select a value to indent the text. Each incremental number will indent the row text by another 10 pixels. This will help visually define sections in the report.

1. Bold Text

Tick this option on to display the text as Bold for this row.

1. Italic Text

Tick this option on to display the text as Italic for this row.

1. Background color

Set a background color for the row. Enter it as a hexadecimal value.

**How To Create Comparative Reporting Columns**

The transposed table approach works very well when you wish to create a financial report that has columns created via a dimension, such as fiscal month or division. When you wish to have columns that each are a separate calculation, then you need to represent those calculations in the data model via a dimension. Take the simple report below for example. This can still be achieved, but it requires a bit more work.

**Option 1. Nested Expression Solution**

One option is to use an artificial dimension to create your columns. Then the expression used for each row would need a nested If() statement or combined Pick() Match() statement to ensure that a different expression was used for each column based on the column header. As a simple example, you might load a dimension (unlinked to the rest of the data model as follows:

reportColumn:

LOAD \* INLINE [

rptColumn, rptSort

Actuals YTD, 1

Prior YTD, 2

Variance $, 3

Variance %, 4

];

Then your revenue row calculation in the report could be defined like the below. The calculation for every other row would follow the same approach but changing the set analysis filter for Account:

if(rptColumn ='Actuals YTD', Sum({<Account={‘Revenue’}, Year={2013}>} Amount),

if(rptColumn ='Prior YTD', Sum({<Account={‘Revenue’}, Year={2012}>} Amount),

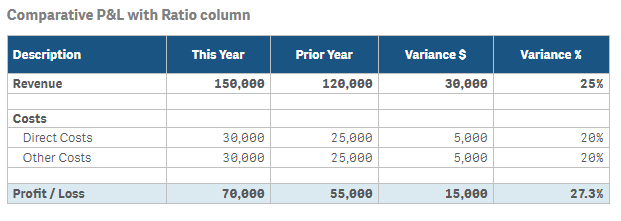
if(rptColumn ='Variance $', Sum({<Account={‘Revenue’}, Year={2012}>} Amount) - Sum({<Account={‘Revenue’}, Year={2012}>} Amount),

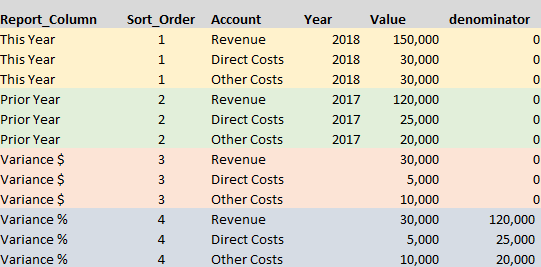
if(rptColumn ='Variance %', (Sum({<Account={‘Revenue’}, Year={2012}>} Amount) - Sum({<Account={‘Revenue’}, Year={2012}>} Amount)) / Sum({<Account={‘Revenue’}, Year={2012}>} Amount)

))))

**Option 2. Data Modelling Solution**

Another option is to use data modelling to give the correct outcome. The data structure that I found works is outlined in the second image below. A sample app is included to demonstrate this approach.





Note the use of the denominator column and how only the ratio column leverages this. In the expressions that are used for the report you need to default a denominator of 1 when the sum is equal to zero. The expression I used for the Revenue row for example is as follows:

sum({<Account={'Revenue'}>} Value)

/

if(sum({<Account={'Revenue'}>} denominator) <> 0, sum({<Account={'Revenue'}>} denominator), 1)