An Alternative to Alternatives

written by DaltonRuer | December 9, 2020

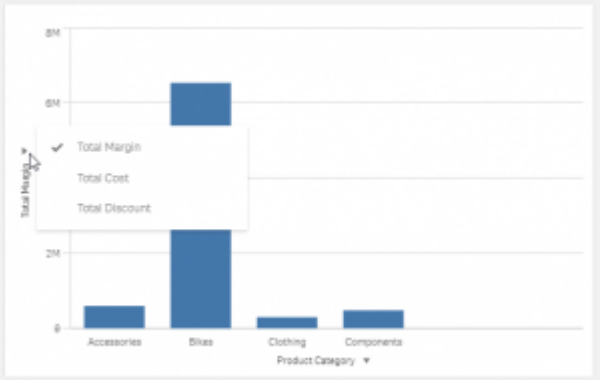
Flexibility

Question: How do you know whether you are an artist or a BI designer?

Answer: By the amount of hands on interaction from your end users.

To that end Qlik provides you the means of adding functionality to engage your users rather than just painting a pretty chart for them. QlikView provides developers the ability to create Cyclic Groups so that end users can select the Dimension they want to see in the chart. Qlik Sense extended that concept even further, allowing developers to create charts with Alternative Dimensions as well as Alternative Measures. More important than the screen space that is saved, is the fact that this flexibility encourages interaction.

After all who can resist clicking the cyclic logo in QlikView applications or the triangles for alternatives in Qlik Sense to see what the data might be saying about those alternatives?



You see, interaction leads to engagement. Instead of the end user just reading the narrative spelled out by the developer they get to create their own story and follow their intuition as insights begin unveiling themselves. In other words, they go from reading the newspaper to researching and editing the articles.

Adding more flexibility

The logical problem with both Cyclic Groups and Alternatives is that they are a fixed size or in a fixed location. The developer must construct them ahead of time. For each chart and in each application. It’s not overly cumbersome, but what if we could accomplish even more engagement for the end user, while also adding flexibility for the developer?

What do I mean by more flexibility for the developer? How about …

- The ability to add more dimensions and more measures without the need to touch the user interface.
- The ability to utilize the same approach for both QlikView and Qlik Sense.
- The ability to define things one time, but utilize them in multiple applications.
- The ability for the system to visualize more/less options based on users or their roles.

Now that I have your attention, let’s dig in.

Islands

In Data Brushing for Context I shared that the absolute wonder of Qlik is its Associate Engine. It's Wonkavator, that enables you to traverse large disparate data sets by following the field associations within the tables. The flexibility we need, takes advantage of the fact that Qlik also enables us to contain **Data Islands** which have absolutely no association to any of the other tables in our data model.

Instead of asking the end user to choose "Bikes or Clothing" as a value, we want them to choose "Product Category or Product Sub Category" as the ***dimension*** that they see in the chart. We want them to choose "Total Sales or Total Cost" as the ***measure*** they want to see in the chart.

I know, I know. That sounds just like Cyclic Groups or Alternatives. Very similar but here is the gold, if we load the options in our data model, we can expand the options at any time. We can load the data into multiple applications. We can refer to the data within many charts. We can utilize the data in both QlikView and Qlik Sense.

Loading Data Islands

Data Islands can be loaded just like any other tables, because they are in fact tables, they just don't have any associations to the rest of the data model. We can load them from a database, from a flat file or we can load them directly in an application using an ***Inline*** load. For the full flexibility described previously we would obviously want to load our dimension and measure list from an external data source. However, for the sake of your ability to understand how to implement this type of templated approach, we will just use the Inline approach so that we can change the model as we go.

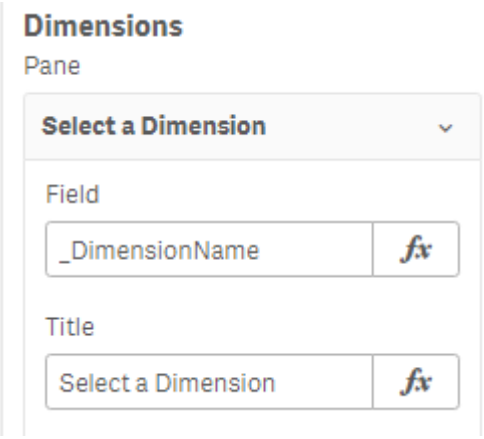
Since the goal is for the end user to have the ability to choose the **dimension** and **measure** for a chart we need an "island" for each. For dimensions, we want the end user to see a name that makes sense, while also providing our ability to use the real field name. For measures, we want the end user to see a name that makes sense to them, as well as contain the expression that should be used. Notice that in addition to the calculation functions, you can also encapsulate the formatting you wish the resulting value displayed in.

```
[Dimension Island]:
Load * Inline [
_DimensionName, _DimensionField
Product Category, Product Category Name
Product Subcategory, ProductSubcategoryName
];

[Measure Island]:
Load * Inline [
_MeasureName, _MeasureExpression
Total Sales, "Num(Sum(SalesAmount), '$#,##0.00')"
Total Cost, "Num(Sum(CostAmount), '$#,##0.00')"
Total Quantity, "Num(Sum(OrderQty), '#,##0')"
];
```

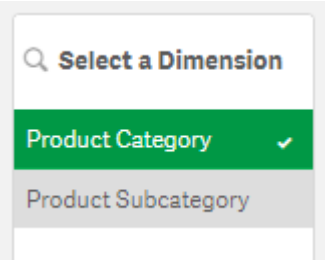
Selecting the Dimension or Measure

To create a filter panel for the dimensions we simply display the _DimensionName or _MeasureName field and provide a simple title like "Select a dimension" or "Select a Measure."



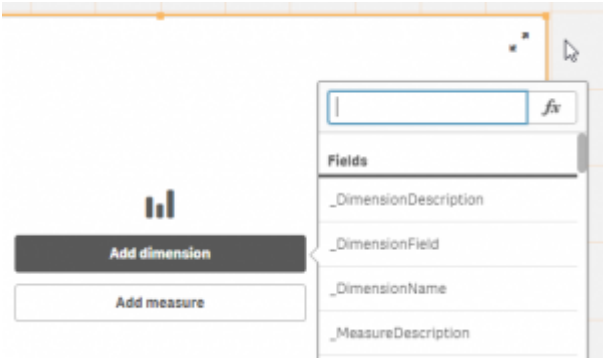
Once visualized the end user can then make their choice, and we have prepared the ingredients for the end user to start

cooking.



Charting the end user's choices

When you create any chart the first thing that Qlik will want to know is the Field name that you want to use for the Dimension.



We have that value in our data model and all we need to do now is provide the appropriate field name "_DimensionField" to Qlik, based on what the end user selected. We can do that via the expression:

= $(= '[' & _DimensionField & ']')

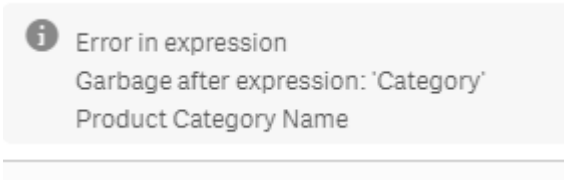
Why the crazy syntax? Why can't we just type in _DimensionField?

Good question. It's because we don't want the field _DimensionField, we want the value that it evaluates to. So we use the Dollar Sign Expansion syntax to tell Qlik to

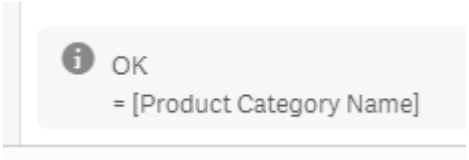
evaluate the value contained in the _DimensionField.

That makes sense, but why concatenate the '[' … ']' around the _DimensionField? Another good question. It's because if we just evaluate the field, and the field evaluates to a field that contains spaces in it, we are in trouble.

=$(=[_DimensionField])



But if we use the complete, but crazy looking expression, Qlik will evaluate the field name contained in our _DimensionField and properly wrap it with brackets, just as Qlik would do if you were hand typing the field and then picked it, and the chart knows exactly what field we are talking about.



The next step is to tell Qlik to use the _DimensionName for the label.

The good news is that the Measure field is even easier. We simply use the syntax:

$(=_MeasureExpression)

And just as with the Dimension we can use the _MeasureName for the label.

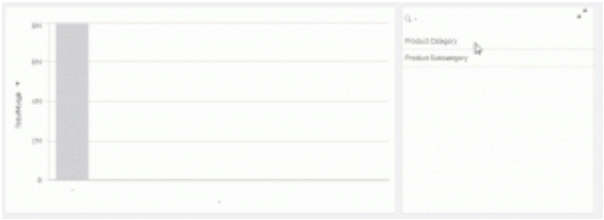
As a summary here are the four expressions you would use for the chart:

Dimension Field	**=$(='['&_DimensionField&']')**

Dimension Label	**=_DimensionName**
Measure Field	**=$(=_MeasureExpression)**
Measure Label	**=_MeasureName**

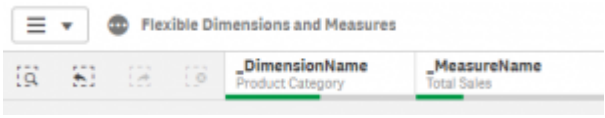
Problems

If the user makes a single selection it all works great. But if they don't, we have a logical problem. Qlik doesn't have a single *expression* to evaluate It has an entire table of expressions. Thus, it has no idea what to do, and the user isn't going to be happy.



The solution is to ensure that they always pick one, and that they can only pick one. In QlikView you would simply set the "Always one selected value" property for the Dimension and Measure filter ***after selecting a value***. In Qlik Sense you would simply find the field in the Fields list and set the property for the _DimensionName and _MeasureName fields there to ensure that there is always one, and only one value selected.

The second issue is more a nuisance than a problem. You see Qlik automatically displays selected values for the end users to provide the context for what they are seeing.



Which is great for "real field" values, but in our case, this can be very confusing for the end user. I hear the user screaming now "Why can't I clear all my selections? {insert your name here} told me that all I had to do was press the X and it would clear all my selections. The X is not even enabled!!!!!!"

The solution is in the fact that Qlik provides a system variable that allows you to hide certain selections. Specifically, those that start with a character set you define. The system variable is conveniently named "HidePrefix" and is used as it's name suggests. To hide fields that start with a given prefix.

If you want to hide the _DimensionaName field you could use the following in your load script:

Set HidePrefix = '_D';

If you want to hide all fields that begin with "_" you would use the following in your load script:

Set HidePrefix = '_';

Notice in this image that values were selected, the chart reflects the selections, but neither field is displayed so the user doesn't have to be concerned about them.

 Be aware that once you set the HidePrefix system variable you not only hide the field in the selection bar, you hide the field in the field list. So be sure you set the "Always one selected value" property for your fields prior to setting it.

Once you have established this *template* if you will, the sky is the limit. You want more dimensions or measures? Simply add them to the data source, and every application will begin displaying them as soon as each is reloaded. Your end users will automatically have more ways to engage with their data.

If you want to limit the available Dimensions or Measures based on a user or their role, simply apply Section Access to the Dimension/Measure islands just like any other table. Your brain is racing now isn't it. But wait, if you continue reading there is even more.

More engagement for the end user?

At the onset of this post, I snuck in the phrase "accomplish even more engagement for the end user" and then proceeded to talk about how to add more flexibility for the developer. None of which made things any "more engaging" from the end user perspective. So what would "more engaging" look like for an end user?

Well, many believe that the goal of analytics is "Actionable Intelligence." But it is not intelligent to act on data you don't trust. Thus, if we are going to create a "more

engaging” user experience it is one that will have to increase trust. Believe it or not you already have the framework to help build the end user’s trust.

All we need to do is add one more column to our data islands for Dimensions and Measures. A simple Description column.

Why would that be a big deal? Because we can then visualize the Description for the end user. We can let them understand exactly what they are looking at. If they agree that the *metadata* is correct, they can not only see the number, they can act on it. If it isn’t they simply choose a different dimension, measure until they find the one their boss, department, region etc. actually care about.

In the screenshot above you will see that there the columns _DimensionDescription and _MeasureDescription have been added. I’ve also added an additional measure to demonstrate the flexibility. You may be scratching your head looking at the values in the _MeasureName column, unless you follow posts online from fellow DataOnThe.Rocks bloggers Christophe Brault or Mária Šándorová. You see, emojis can also add to the engagement for users. Contrast the image below with the one above. The □ and #□□ emojis help the end user quickly and clearly find the measures that involve monetary values versus numeric amounts. Small splash of color and yet it’s dramatic how quickly the brain reacts to it. Isn’t it?

Now imagine a longer list of measures, yet the end users can quickly find the percentages, totals, monetary values.

Hopefully, in addition to the visualization of the emojis, you also recognized that the chart has been updated to include the _DimensionName/_MeasureName fields as well as the new Description. Look how clear everything becomes for the end user. They know what they are filtering, and they know exactly what is being visualized. Hopefully you agree that's pretty powerful, yet so easy to accomplish. It's what people love about adding descriptions to Master Items, but without requiring end users to find it.

You need even more measures, don't you?

You already know you can add as many rows to your Measure Island as you wish. So what do I mean?

I am talking about the fact that not all chart types are easy as a pie or bar chart. They require more than 1 measure. Can we use the same framework for something like a Scatter Plot where we need X, Y, Size and Color?

Of course, we can or I wouldn't have brought it up. Simply load 3 more measures islands from our now *resident* data island:

```
[Measure Island 2]:
Load
    _MeasureName as _MeasureName2,
    _MeasureExpression as _MeasureExpression2,
    _MeasureDescription as _MeasureDescription2
Resident [Measure Island];

[Measure Island 3]:
Load
    _MeasureName as _MeasureName3,
    _MeasureExpression as _MeasureExpression3,
    _MeasureDescription as _MeasureDescription3
Resident [Measure Island];

[Measure Island 4]:
Load
    _MeasureName as _MeasureName4,
    _MeasureExpression as _MeasureExpression4,
    _MeasureDescription as _MeasureDescription4
Resident [Measure Island];
```

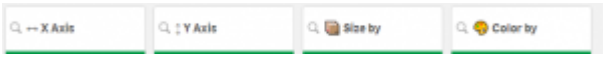
Your expression for the measures simply change from

=$(=_MeasureExpression)

Into

=$(=_MeasureExpression2), =$(=_MeasureExpression3) or =$(=_MeasureExpression4)

But what about the Measure label? Should we just change to “Select Measure 1” “Select Measure 2” “Select Measure 3” and “Select Measure 4?” No way, we aren’t rookies. We are experienced emoji veterans, and there is no better way to use them, than include them in the title so the user immediately recognizes what they would be selecting the measure for.



This has been a lot of mind blowing information in one post. So now it’s time to relax and simply interact with the application below to see all of the concepts in action. **Feel free to change the dimension and change any of the 4 measures.**

Bonus Material for Advanced

Developers

If you are new to Qlik, and what was shared to this point really leaves you scratching your head feel free to step away from the monitor. But if you're a 20 year Qlik veteran, and you need more here are a few more concepts for your to consider to really take this "concept" to extremes.

Bonus 1: Instead of embedding expressions in your Measure Island utilize variables instead, and change your expression to visualize the result to a double-dollar sign expansion since the first expansion will return the variable name, we have to then expand the variable itself.

Set vTotalNetValueOfOrderedItems = Sum([Sales Items Net Value NETWR]);

In your charts use: **$(=$(=_MeasureVariable))**

Bonus 2: If you have been designing Qlik applications for any length of time, you will know that end users are fussy. As soon as you do something like this, they will immediately say "That's great, but I need to see all of these measures but just for the Current Year To Date. Now I'm super glad we switched to using variables. All we need to do is add a variable that would calculate the Current Year to Date version of the variable and then add it to our data model, not just replace the existing one:

In your user interface you would have the option to refer to either of the variables. If you need some KPI's or charts that focus on Current Year to Date simple change your double dollar sign expansion expression to:

$(=$(=_MeasureVariableCY))

Bonus 3: I can almost hear the gears in your head turning now. "What if I wanted the end user to be able to choose if they

want to see the screen for “all time” or “just current year to date?” Great creativity on your part. Let’s take it further. What if you wanted the end user to choose All time, Current Year to Date, Current Month to Date, Previous Month to Date or a slew of other choices? The answer of course is to simply create another data island for those options. One where you present the name like “Current Year to Date” and the other column is a suffix value like “CY”. Then your on-screen expressions would include an evaluation to add the suffix. Now that’s kind of cool. Or in a table you might have 4 different columns to show all 4 (or more) of the measure values easily providing you follow a defined a pattern.

Bonus 4: If you were paying attention to the screen shots of the code you would have noticed that I included _MeasureNumber and _MeasureType as fields. Take a few minutes to guess why?

If you guessed that I did that in order to provide a way to use Section Access and limit which are available based on the end user you are one hundred percent correct.