How to Calculate Relative-time Measures in Reports and Dashboards

One of the more common requirements for analysis is the comparison of key performance indicators, or measures, over time. One common approach for report and dashboard developers is to assign some date-based filter for each measure to show, for example, last year's sales vs. sales from 2 years ago. This approach quickly deteriorates once it is realized that no single query can resolve BOTH the prior AND the two years' prior period filter criteria:

Filter 1: year(sale_date) = year(current_date) - 1

AND

Filter 2: year(sale_date) = year(current_date) – 2

There are no data for which BOTH constraints are true. If we replace the AND condition with an OR, the query returns sales for BOTH one year ago and from two years ago, and the measure in the report shows the total sales for both periods.

This is a classic condition in which multiple queries may be required. We could create a query with the first constraint, a second query with the second constraint, and join the results – most BI tools allow this level of sophistication. However, this approach is typically sub-optimal, although there are certainly cases in which modular sub-queries can provide additional agile design benefits. So, for the vast majority of applications, I highly recommend a different approach.

One of the many benefits of our recommended approach is that it is not specific to any one BI tool. Wherever you can implement simple conditional expressions, with either "if…then…else" or "CASE" constructs, you can take advantage of this solution. Another benefit is that these expressions are much more portable and lightweight. Consider the serial multi-query approach as your requirement expands to many relative-date measures. The ever-expanding number of queries, joins, and joins of joined queries will quickly become unmanageable.

OK, so let's get down to it. The solution conditionalizes the date range within a relatively simple expression for each measure. In this example, the base measure is called **sales**. The base date is an item called **sale_date**. I will also use some commonly-available date functions, although these will vary based on which BI tool, SQL standard, and underlying RDBMS you might be using. However, pretty much every environment supports the same data manipulation, even if the specific function names and/or arguments may differ.
Here is the expression for TOTAL (**not** YTD) sales from one year ago:

```
If(year(sale_date) = year(current_date)-1) then (sales) else (0)
```

Here is the expression for TOTAL (**not** YTD) sales from two years ago:

```
If(year(sale_date) = year(current_date)-2) then (sales) else (0)
```

As you can see, the expressions are designed to return values at the lowest level of detail. Both calculations can exist side-by-side in a single query. Most tools will then automatically aggregate the values in a report and/or dashboard, however you will need to be cautious about what is actually happening in your environment. It should be easy enough to troubleshoot if you are experiencing aggregation issues. You can usually resolve them by tweaking one or more object properties, or you may possibly need to explicitly calculate the aggregate within the above expressions.

Of course, if your BI application has pre-built relative measures, those are the most direct solution. However, keep in mind the potential drawbacks of these resources, such as the lack of control over the centrally-defined base date and calendar structure. Also keep in mind that this example is relatively simple. Calculating prior periods at the month, quarter, and day levels can still be achieved, however these will require more sophisticated logic. Additional complexity is also required for "to-date" calculations, but they are still addressable using this approach.

One last note: this approach is not limited to relative-time applications! It is a very powerful technique that can derive contextualized measures based on any available attributes of your data. Another popular twist is to parameterize the input values to give information consumers more control and interactivity.

Enjoy – and please reach out if you need anything!

Intelalytics
714.865.4627
albert@intelalytics.com
www.intelalytics.com