Featherman’s Date and Time DAX Functions

We will make some measures here. Measures are not stored in columns for every row of a table. Rather they are expressions (formulas) that can be placed into charts, maps, tables, and matrices and depending on the evaluation context (the filtering in the object.) The amazing innovation with measures is that they can perform different calculations depening on where they are used, so they are portable logic.

These formulas depend on a date table being added to the data model. A date table is a custom made table of dates and there is a row for each day in the year whether there is a sale on that day or not.

|  |  |  |
| --- | --- | --- |
| 1 | TotalUnits =  SUM(Sales[SalesQuantity]) | This will provide a sum of the units sold on sales invoices. Depending on the slicers and row and column headings the returned result will compute. |
| 2 | PrevQtrUnits =  CALCULATE(SUM(Sales[SalesQuantity]), PREVIOUSQUARTER(Calendar[DateKey])) | This measure has a formula will calculate the number of units sold for the previous quarter. This is made possible by a connection to a date table both tables have a datekey which is a date field datatype. |
| 3 | DeltaUnitsQtr =  ([TotalUnits]-[PrevQtrUnits])/[PrevQtrUnits] | Be sure to format this measure to show a percentage. The structure is  (this quarter – last quarter)/last quarter |
| 4 | YTDUnits = TOTALYTD(SUM(Sales[SalesQuantity]),'Calendar'[DateKey]) |  |
| 5 | YOYQTR\_Units =  CALCULATE(sum(Sales[SalesQuantity]), SAMEPERIODLASTYEAR('Calendar'[DateKey])) | The calculate function allows you to provide several layers of filters, one of which is called Sameperiod last year. Here you can see the quarter was not specified, but the formula figured out the evaluation context.  The SUM part of the CALCULATE function is the same as formula in #1 above. The filter here figures out the same time period last year. When we practice all of these formulas again using Month data you can see the magic of this function – SAMEPERIODLASTYEAR  The formula works because there is a link between the fact table and the calendar table. |
| 6 | YOY%DeltaUnits =  ([TotalUnits] - [YOYQTR\_Units])  /[YOYQTR\_Units] | We can see the change in units sold as compared to the prior year. Again the formula is (the new period – the old period)/Old period |

|  |  |  |
| --- | --- | --- |
|  | In-class student challenge #1 |  |
| 7 | PrevMonthUnits =  CALCULATE(SUM(Sales[SalesQuantity]), PREVIOUSMONTH(Calendar[DateKey])) | This formula is essentially the same as # 2 above. Please go ahead and create this formula and recreate formulas 3, 4, 5,6 using a month time frame. |
|  | In-Class student challenge #2 |  |
| 8 | MTDUnits = TOTALMTD(SUM(Sales[SalesQuantity]),'Calendar'[DateKey]) | Make a new tab on your dashboard and display unit sales by the day. Then add a mothtodate and a year to date colulmn and % change formulas. These columns will keep a running total of sales. |
|  | Fixing some problems with comparison functions |  |
| 9 | YOYDeltaUnitsFixed =  IF([YOYQTR\_Units]=0," ", ([TotalUnits] - [YOYQTR\_Units])/[YOYQTR\_Units]) | The formatting is = IF(test condition, action if test condition is true, action if test condition is false)  This formula reads if the prior year # units is 0 then place a blank in the column, otherwise do the formula. |