

Power BI - DAX - p1 - Lecture 15

MEET DAX

Data Analysis Expressions (commonly known as DAX) is the formula language that drives the Power BI front-end. With DAX, you can:

- Go beyond the capabilities of traditional spreadsheet formulas, with powerful and flexible functions built specifically to work with relational data models.
- Add calculated columns (for filtering) and measures (for aggregation) to enhance data models.

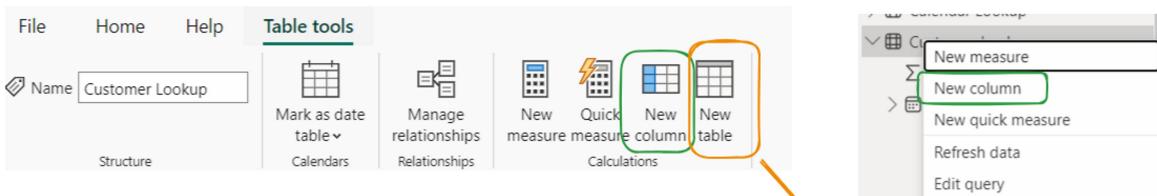


1 Parent = IF('Customer Lookup'[Total Children]>0,"Yes","No")					
Marital Status	Email Address	Annual Income	Total Children	Education Level	Parent
M	emma32@adventure-works.com	70000	5	Bachelors	Yes
M	barry20@adventure-works.com	40000	5	High School	Yes
M	martha13@adventure-works.com	70000	5	High School	Yes
S	tamarat16@adventure-works.com	40000	5	High School	Yes
S	gerald21@adventure-works.com	130000	5	Bachelors	Yes
M	alexa@adventure-works.com	40000	5	High School	Yes
M	jack53@adventure-works.com	70000	5	Graduate Degree	Yes
S	ricky1@adventure-works.com	100000	5	Bachelors	Yes
M	keith4@adventure-works.com	70000	5	Partial College	Yes
M	latoya19@adventure-works.com	70000	5	Bachelors	Yes

The screenshot shows the Power BI DAX editor interface. A search bar at the top suggests "Customer Lookup". Below it, a "Select" dropdown menu includes options like "New measure", "New column", "New quick measure", "Refresh data", "Edit query", and "Manage relationships". Several DAX code snippets are highlighted with yellow boxes:

- "Total Orders = DISTINCTCOUNT(Sales_Data[OrderNumber])"
- "Total Revenue = SUMX(Sales_Data, Sales_Data[OrderQuantity] * RELATED(Product_Lookup[ProductPrice]))"
- "Quantity Ordered = SUM(Sales_Data[OrderQuantity])"

$$\begin{aligned}
 X &= 2Y + 5; & \text{product_price} &= 10 \\
 \text{if } y &= 10: & \text{order_quantity} &= 5 \\
 X &= 25. & \text{profit} &= \text{product_price} * \text{order_quantity}
 \end{aligned}$$



Customer Lookup

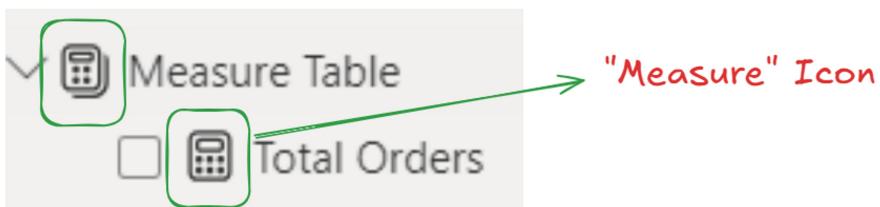
IF(LogicalTest, ResultIfTrue, [ResultIfFalse])
Checks whether a condition is met, and returns one value if TRUE, and another value if FALSE.

Parent = IF('Customer Lookup'[TotalChildren] > 0, "Yes", "No")

Let's Create a Measure Table where we can store all Explicit measures.

Click The Enter Data & Create A Table > Table Rename it to "Measure Table"

→ Now, Once Measure Table is being created. We can create unlimited Explicit Measure in it.



- CALCULATE(Expression,Filter)
"Applying some filter"

13929
Bikes Order

Calling out existing Explicit Measure

Bikes Order = CALCULATE([Total Orders], 'Product Categories Lookup'[CategoryName] = "Bikes")

Explicit Measure helps us to write complex calculation.

Mountain Bikes Orders = CALCULATE([Bikes Order], 'Product Subcategories Lookup'[SubcategoryName] = "Mountain Bikes")

CategoryName	Sum of OrderQuantity
Accessories	57809
Bikes	13929
Mountain Bikes	4706
Road Bikes	7099
Touring Bikes	2124
Clothing	12436
Total	84174

CategoryName	Sum of OrderQuantity	Total Orders	Bikes Order
Accessories	57809	57809	13929
Bikes	13929	13929	13929
Clothing	12436	12436	13929
Components			13929
Total	84174	84174	13929

M VS. DAX

M and DAX are two distinct functional languages used within Power BI Desktop:

- M is used in the Power Query editor, and is designed specifically for extracting, transforming and loading data.
- DAX is used in the Power BI front-end, and is designed specifically for analyzing relational data models

M CODE

Query Editor:

```

#"Changed Type" = Table.TransformColumnTypes(
    #"Promoted Headers", // Adding a new step
    {                      // after we promoted headers
        {"SalesTerritoryKey", Int64.Type}, // that changes column datatypes
        {"Region", type text},
        {"Country", type text},
        {"Continent", type text}
    }
)

```

DAX

Report View:

Category Name	Total Returns	Bike Returns
Accessories	1,115	
Bikes	427	427
Clothing	267	
Total	1,809	427

```

1 Bike Returns =
2 CALCULATE(
3     [Total Returns],
4     'Product Categories Lookup'[Category Name] = "Bikes" // Counting total returns
5 ) // filtered for bikes only

```

CALCULATED COLUMNS



Calculated columns allow you to add new, formula-based columns to tables in a model

- Calculated columns refer to entire tables or columns (no A1-style cell references).
- Calculated columns generate values for each row, which are visible within tables in the Data view.
- Calculated columns understand row context; they're great for defining properties based on information in each row, but generally useless for aggregation (sum, count, etc.)

HEY THIS IS IMPORTANT!

As a rule of thumb, use calculated columns to "stamp" static, fixed values to each row in a table (or go upstream and use the Query Editor!)
DO NOT use calculated columns for aggregation – this is what measures are for!

Calculated Columns - Filtering & Grouping.
- Row Context.

Measures - Filter Context.
- Aggregation.

PRO TIP:

Calculated columns are typically used for filtering & grouping data, rather than creating aggregate numerical values

EXAMPLE: CALCULATED COLUMNS



Email Address	Annual Income	Total Children	Education Level	Parent
emma32@adventure-works.com	70000	5	Bachelors	Yes
barry20@adventure-works.com	40000	5	High School	Yes
martha13@adventure-works.com	70000	5	High School	Yes
tamara16@adventure-works.com	40000	5	High School	Yes
gerald12@adventure-works.com	130000	5	Bachelors	Yes
alexa8@adventure-works.com	40000	5	High School	Yes
jack53@adventure-works.com	70000	5	Graduate Degree	Yes
ricky1@adventure-works.com	100000	5	Bachelors	Yes
keith4@adventure-works.com	70000	5	Partial College	Yes
latoya19@adventure-works.com	70000	5	Bachelors	Yes

In this case we've added a calculated column named Parent, which equals "Yes" if the [Total Children] field is greater than 0, and "No" otherwise

- Since calculated columns understand row context, a new value is calculated in each row based on the value in the [Total Children] column.
- This is a valid use of calculated columns; it creates a new row "property" that we can use to filter or segment any related data within the model.

Here we're using an aggregation function (SUM) to calculate a new column named Total Quantity

The screenshot shows the Power BI Data View interface. A yellow arrow points from the explanatory text above to the 'TotalQuantity' column in the table. The table contains 15 rows of sales data. A yellow box highlights the 'TotalQuantity' column, which contains the value 84174 for every row. The Fields pane on the right lists various lookup tables and the Sales Data table.

- Since this is an aggregation function, the same grand total is returned in every row of the table
- This is not a valid use of calculated columns; these values are statically "stamped" onto the table and can't be filtered, sliced, etc.

DAX MEASURES

Measures are DAX formulas used to generate new calculated values

- Like calculated columns, measures reference entire tables or columns (no A1-style cell references).
- Unlike calculated columns, measures aren't visible within tables; they can only be "seen" within a visualization like a chart or matrix (similar to a calculated field in a PivotTable).
- Measures evaluate based on filter context, which means they recalculate when the fields or filters around them change.

C.C - Model View
- Table View
Measures - Report View

HEY THIS IS IMPORTANT!

As a rule of thumb, use measures when a single row can't give you the answer, or when you need to aggregate values across multiple rows in a table

Rabbani to You (direct message) 10:42

1 ➕ 😊 ...

True

filtering context is like applying filter on dimension table affects the fact table ?

PRO TIP:

Use measures to create numerical, calculated values that can be analyzed in the "values" field of a report visual

IMPLICIT VS. EXPLICIT MEASURES

The screenshot shows the 'Build a visual' interface in Power BI. On the left, under 'Y-axis', there is a field named 'Sum of Order ...'. This field is highlighted with a yellow box. A yellow arrow points from this box to the 'Select data' pane on the right. The 'Select data' pane lists various fields, and the 'Sum of Order ...' field is also highlighted with a yellow box.

Example of an implicit measure

Implicit measures are created when you drag raw numerical fields into a report visual and manually select an aggregation mode (Sum, Average, Min, Max, Count, etc.)

Explicit measures are created when you actually write a DAX formula and define a new measure that can be used within the model

HEY THIS IS IMPORTANT!

Implicit measures are only accessible within the specific visualization in which they were created, and cannot be referenced elsewhere.

Explicit measures can be used anywhere in the report, and referenced by other DAX calculations to create "measure trees".

The screenshot shows the 'Build a visual' interface. In the 'Values' section, there are two fields: 'Sum of OrderQuantity' and 'Sum of ReturnQuantity', both of which have yellow boxes around them. A blue box surrounds the dropdown menu that appears when clicking on one of these fields. The menu is titled 'Sum' and contains various aggregation options: Average, Minimum, Maximum, Count (Distinct), Count, Standard deviation, Variance, Median, and Sum. An arrow points from the text 'Implicit Measures.' to this menu.

Implicit Measures.