

Case Study

Power BI Top 10 DAX Functions

Data Analysis Expressions (DAX) is a specialized formula and query language used in Power BI to create custom calculations and perform advanced data analysis. Some common DAX concepts are:

- **Measures:** Dynamic calculations where results change based on filters, slicers, and report context. They are calculated at query time and do not store data.
- **Calculated Columns:** New columns added directly to a table using DAX formulas. They are calculated row-by-row and stored in the data model.
- **Calculated Tables:** Entire tables generated by a DAX expression, often used for date tables or filtered subsets.

Some of the most common categories of DAX functions are:

- Aggregation functions (SUM, AVERAGE, MIN/MAX, COUNT/DISTINCTCOUNT)
- Filter functions (CALCULATE, FILTER, ALL)
- Logical functions (IF/SWITCH)
- Text functions (CONCATENATE)
- Time Intelligence functions (SAMEPERIODLASTYEAR/DATEADD)

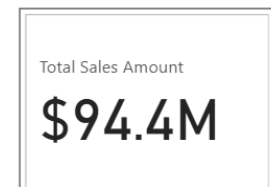
To create a Measure: (In Power BI Report View)

1. Right click on the field that you want to create a measure for and select **New measure...**
2. Type the DAX formula in the formula bar at the top
3. Press the Enter key to complete the formula
4. Click somewhere in the report screen to hide the formula bar
5. The measure will appear in the Data pane on the right
6. Add the measure to a visual to see the results

1. SUM:

The **SUM** function calculates the sum of a specified column or expression. It is commonly used to aggregate numerical data. Example: Suppose you have a field called GROSSAMOUNT that you want to total. You can use the SUM function to calculate the total sales amount:

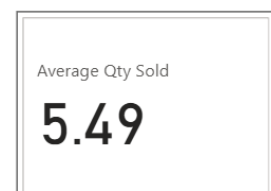
Total Sales Amount = SUM(Sales[GROSSAMOUNT])



2. AVERAGE:

The **AVERAGE** function calculates the average of a specified column or expression. It is useful for finding the average value of numerical data. Example: You have a field called QUANTITY. You can use the AVERAGE function to calculate the average quantity sold:

Average Qty Sold = AVERAGE(Sales[QUANTITY])



3. MAX/MIN:

The **MIN** and **MAX** functions return the minimum and maximum values, respectively, from a specified column or expression. Example: Suppose you have a field called PRICE. You can use the MAX and MIN functions to find the highest and lowest prices, respectively:

Highest Price	Lowest Price
\$7,900	\$249

Highest Price = MAX(Products[PRICE])

Lowest Price = MIN(Products[PRICE])

4. COUNT/DISTINCTCOUNT:

The **COUNT** function counts the number of rows in a table or column that contains a value. It is helpful for determining the number of occurrences. The **DISTINCTCOUNT** function does not count duplicates. Example: Suppose you have a field called SALESORDERID. You can use the COUNT function to count the number of transactions in the Sales file. You can also use the DISTINCTCOUNT function to count the number of orders in the Sales file since each order may contain multiple transactions:

# of Transactions	# of Orders
1323	232

of Transactions = COUNT(Sales[SALESORDERID])

of Orders = DISTINCTCOUNT(Sales[SALESORDERID])

5. CALCULATE:

The **CALCULATE** function is one of the most important functions in DAX. It allows you to modify the context in which a calculation is performed, enabling you to filter, modify, or override the default behavior of calculations. Example: Suppose you have a field called GROSSAMOUNT and PRODCATEGORYID. You can use the CALCULATE function to filter the total sales amount by a specific product category:

BX Sales
\$15M

BX Sales = CALCULATE(SUM(Sales[GROSSAMOUNT]),Products[PRODCATEGORYID]="BX")

6. FILTER:

The **FILTER** function returns a table representing a subset of another table based on a specific condition. It is often used as a nested argument within CALCULATE to apply more complex logic. Example: Suppose you have a field called GROSSAMOUNT. You can use CALCULATE and FILTER functions to calculate total sales but only for those amounts greater than \$25,000:

High Sales
\$93M

High Sales = CALCULATE(SUM(Sales[GROSSAMOUNT]),
FILTER(Sales,Sales[GROSSAMOUNT]>25000))

7. ALL:

The **ALL** function removes filters from a table or column. It's essential for calculating totals or percentages. Example: Suppose you have a field called GROSSAMOUNT. We already know that most of the visuals get automatically filtered when using selecting criteria on a report. You can use the ALL function within the CALCULATE function to always show the total sales amount and never filter the answer based on how other visuals change:

Grand Total
\$94M

Grand Total = CALCULATE(SUM(Sales[GROSSAMOUNT]), ALL(Sales))

8. IF/SWITCH:

The **IF** function performs conditional evaluations. It returns one value if a condition is true and another value if the condition is false. Example: Suppose you have a field called GROSSAMOUNT that you want to test to see if the total sales amount is on target. You can use the IF function to categorize sales as "On Target" or "Not on Target" based on a threshold:

Sales Comment = IF(SUM(Sales[GROSSAMOUNT])>85000000, "On Target", "Not on Target")

Total Sales Amount \$94M	Sales Comment On Target
------------------------------------	-----------------------------------

You can nest IF statements if you have multiple conditional evaluations:

Sales Comment2 = IF(SUM(Sales[GROSSAMOUNT])>85000000, "Above Target",IF(SUM(Sales[GROSSAMOUNT])>50000000,"On Target","Below Target"))

Total Sales Amount \$94M	Sales Comment2 Above Target
------------------------------------	---------------------------------------

The **SWITCH** function is like the IF function but allows you to evaluate multiple conditions and return different values based on each condition. By default, the SWITCH function only works on numeric fields. If you want to apply it to text fields you will have to use the CONTAINSSTRING function and the TRUE statement within the SWITCH function. Example: Suppose you have a field called TYPECODE that has the following values: PR, CA, and AR. You can use the SWITCH function to change the values:

New Rating=SWITCH(Products[Rating], 1, "Poor", 2, "Average", 3, "Good", 4, "Excellent")

9. CONCATENATE:

The **CONCATENATE** function combines multiple text strings into a single text string. It is handy for creating concatenated fields. Example: Suppose you have a table with columns for first and last names. You can use the CONCATENATE function to create a full name field:

Full Name=CONCATENATE(Customers[First Name], " ", Customers[Last Name])

10. DATEADD:

The **DATEADD** function in DAX is a time intelligence tool used to shift a set of dates forward or backward by a specified interval (day, month, quarter, or year). Example: Suppose you have a sales order date field (CREATEDAT) and you want to calculate what the total sales were one year ago:

Prev Sales = CALCULATE(SUM(Orders[GROSSAMOUNT]),DATEADD(Orders[CREATEDAT].[Date],-1,YEAR))

Product Code	Sales	Prev Sales
AR	\$91,909,917	\$62,969,185
CA	\$41,440,814	\$28,620,102
PR	\$223,391,600	\$153,666,866
Total	\$356,742,331	\$245,256,153