

Introduction To DAX

What is DAX?

- Data Analysis Expressions
- Formula Expression Language for Power BI, Power Pivot and SSAS Tabular.
- Formula includes functions, operators and values.
- Dax is used to create Calculated Columns, Calculated Measures and Calculated Tables

Calculated Columns:

- Create New Columns on a table.
- Method for Connecting Disparate Data Sources with Multiple Key Columns.
- Row by row calculation called **Row Context** in DAX terminologies.
- Stores values in the memory.
- When you want to Slice and Dice data based on fields that do not exists in data model you need to create **calculated columns(Columns created using DAX)**
- Come with cost as they evaluated per row, consume memory and increase size of data model.
- Data refresh take longer time as column values are evaluated at refresh time and not on demand.

Example:

Create a calculated Column Full Name in Customer Table

Navigation:Table View→Table Tools→New Column

Full name = Customer[FirstName] & " " & Customer[LastName]

Calculated Measures

- Create Dynamic Calculations for Reporting.
- Usually, calculation works on an aggregated level basis.
- Not pre calculated, Evaluated dynamically on the Fly
- Calculated based on subset of data *selected by filters, slicers*
- *This filtered dataset, called Filter Context*
- Measures do not consume RAM; they consume CPU
- Used for buildings aggregates, ratios, percentages, Time intelligence calculations.

Example:

Calculate Total Sales

Navigation:Table View→Table Tools→New Measure

Total Sales=sum(InternetSales[SalesAmount])

Calculated Tables

- Create a new table derived from another table.
- These are in-memory tables.

- Can be used to create a date table when one doesn't exist already.

Example:

Create FemaleCustomer Table ,From Customer Table

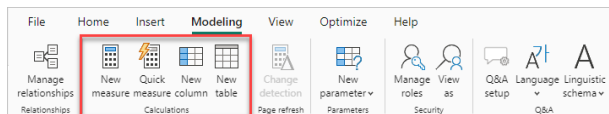
Female Customer = `FILTER(Customer, Customer[Gender]="F")`

Create Date Table

CalenderDataTable = `CALENDAR("1/1/2018", "12/31/2022")`

Navigation for Creating Calculations:

1. Report View → Modeling Tab → Calculations



2. Table View → Table Tools → Calculations
3. ModelView → Home → Calculations

Working With Calculated Columns

Custom column can be created by following ways

- Create Column in source query
- Create Custom column in Power Query
- Create Calculated column using DAX.

Note: The compression of a DAX calculated column might be lower than that of a Power Query computed column. Comparatively it makes so *pbix file is bigger and the performance usually lower..*

Lab 1: Create Calculated Columns on Customer Table

1. For Lab open file in following Location:
C:\PowerBI Learn\ClassLabs\Dax\Model-Dax-starter
2. Save the File in same folder as Model-Dax-Final.pbix and start working with Model-Dax-Final.pbix file.
3. Table View → Select Customer Table → Table tools Ribbon → Click on **New Column** → Observe the formula bar → Create Following Calculations

Note: For Each of the Calculations you need to click on New Column and create it.

Create a calculated Column Full Name

Full Name = `Customer[FirstName] & " " & Customer[LastName]`
//Press Enter Key to commit the calculation.

Create a calculated Column Age

Age =

```
IF(
    FORMAT('Customer'[BirthDate], "mmdd") <= FORMAT(TODAY(), "mmdd"),
    DATEDIFF('Customer'[BirthDate], TODAY(), YEAR),
    DATEDIFF('Customer'[BirthDate], TODAY(), YEAR) - 1)
```

//Simple IF Age type = IF(Customer[Age] >55,"Middle Age","Young")
//Nested IF //Age Breakdown Calculation Age breakdown = IF(Customer[Age] >= 55,"55+", IF(Customer[Age] >=45,"45-54", IF('Customer'[Age] >= 35, "35-44", "18-34"))) //Use SHIFT+ENTER to go to next line and auto format
//Using SWITCH -- Age Breakdown Calculation (Column on Customer Table) Age Breakdown1 = SWITCH (TRUE(), 'Customer'[Age] >= 55, "55+", 'Customer'[Age] >= 45, "45-54", 'Customer'[Age] >= 35, "35-44", "18-34")

Navigation Function:

- They are Like VLOOKUP or JOIN
- **RELATED:** Returns value from a connected table
- **RELATEDTABLE:** Returns a table from a connected table
- For navigation functions to work, relationship must exists between tables in data model.

Get Region column in FactInternetSales Region = RELATED(SalesTerritory[Region])
Get Total Sales in Region in Sales Territory Region Table Total Sales = SUM('InternetSales'[SalesAmount]) //Filter is disabled for row context Total Sales = SUMX(RELATEDTABLE(InternetSales),InternetSales[SalesAmount]) // X function :x works on one row at a time performs expression and returns single column table.Sum calculates the Sum on those column values.
Get Last order Date for Customers in Customer Table Last Order Date =MAX('InternetSales'[OrderDate]) //Filter is disabled for row context Last Order Date = MAXX(RELATEDTABLE('InternetSales'), 'InternetSales'[OrderDate])
//Total Transactions (Column on Sales Territory table) Total Transactions = COUNTROWS('InternetSales') //Wrong

```
Total Transactions = COUNTROWS(RELATEDTABLE('InternetSales'))
```

```
// Region Volume
```

```
Region Volume =
```

```
SWITCH(TRUE(),
    [Total Transactions] >= 7000, "High Volume",
    [Total Transactions] >= 4000, "Medium Volume",
    [Total Transactions] >= 1, "Low Volume",
    "N/A" )
```

Working With Measures

- Measure names must be unique as they are not columns of table.
- You can create separate table for managing measures
- You can organize all columns of table in separate folder so remaining things can be measures.

What is FILTER Context?

Filter context describes the filters that are applied during the evaluation of a measure or measure expression.

What makes Filter Context?

- Rows and Columns Headers
- Slicers
- Filters in filters pane
- DAX Formula Filters

Category

☐ Accessories
 ☒ Bikes
 ☐ Clothing
 ☐ Components

Reseller Revenue

Group	Country	Region	Specialty Bike Shop	Value Added Reseller	Warehouse	Total
Europe	France	France	\$218,497.65	\$711,788.4	\$5,522,891.18	\$2,432,178.76
		Total	\$218,497.65	\$711,788.4	\$5,522,891.18	\$2,432,178.76
	Germany	Germany	\$142,167.41	\$327,816.0	\$872,891.20	\$1,342,015.61
		Total	\$142,167.41	\$327,816.0	\$872,891.20	\$1,342,015.61
	United Kingdom	United Kingdom	\$191,318.02	\$74,918.0	\$1,125,341.11	\$2,291,570.25
Total	Total	\$191,318.02	\$74,918.0	\$1,125,341.11	\$2,291,570.25	
	North America	Canada	Canada	\$331,396.15	\$45,923.4	\$2,393,818.71
Total			\$331,396.15	\$45,923.4	\$2,393,818.71	\$4,370,334.91
United States		Central	\$183,178.00	\$143,462.0	\$760,100.00	\$2,014,958.27
		Northeast	\$200,763.00	\$79,124.0	\$749,000.00	\$1,880,546.82
		Northwest	\$1,141,154.00	\$80,307.0	\$6,622,400.00	\$4,641,677.36
	Southeast	\$1,141,154.00	\$80,307.0	\$6,622,400.00	\$4,641,677.36	
Total	Total	\$1,306,994.18	\$872,891.20	\$8,534,345.19	\$16,884,499.45	
	Pacific	Australia	Australia	\$298,723.55	\$672,588.82	\$232,110.86
Total		\$298,723.55	\$672,588.82	\$232,110.86	\$1,223,423.71	
Total		Total	\$2,850,540.82			

Filters

Search

Filters on this visual

Business Type is (All)

Channel is Reseller

Country is (All)

Group is (All)

Region is (All)

Revenue is (All)

Add data fields here

Filters on this page

Fiscal Year is FY2020

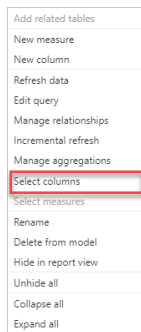
Add data fields here

Filters on all pages

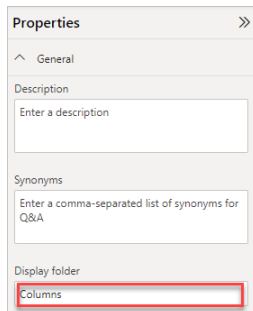
Pause

Lab 2: Create Measures:

1. Before creating measure create arrange all columns from InternetSales in Folder called Columns.
2. Model View → Select InternetSales Table → Observe ... (more options) → click on it → Select **Select columns** → All the columns are selected



3. In properties pane specify Display folder as Columns



4. Start Creating Measures given below:

Navigation:

Report View → Select InternetSales Table → Table tools → New measure →

//Total Sales
Total Sales = <code>sum(InternetSales[SalesAmount])</code>
// Total Transactions
Total Transactions = <code>COUNTROWS('InternetSales')</code>
//Total Cost
Total Cost = <code>SUM('InternetSales'[TotalProductCost])</code>
//Profit(Using already created measures in expression)
Profit = <code>[Total Sales]-[Total Cost]</code>
// Profit Margin
Profit Margin = <code>DIVIDE([Profit], [Total Sales])</code>

Note: You can change Home Table for measure as it does not belong to any table.

Working with CALCULATE

- Base measure works all the time with different filter contexts.
- If you want to have derivative or variation of automatic filter context, use CALCULATE to evaluate that expression on different filter context.
- CALCULATE allows you evaluate expression based on Filter Expression
- Example: Total Sales of Last Year, Total Sales for US

Syntax:

CALCULATE(<expression>, [[<filter1>], <filter2>]...)

- Expression is Aggregate.
- Filters add to or override Filter Context.

Examples: Get Percentage sales for different countries with respect to Total Sales.

1. Calculate Total Sales for ALL Countries ignoring the filter context
2. Divide Total sales in filter context by the calculation created previously.

ALL: Returns all the rows in table or all values in a column, **ignoring** any filter applied.

-- Total Sales (All Countries)

Total Sales (All Countries) =

```
CALCULATE(  
    [Total Sales], ALL(SalesTerritory[Country])  
)
```

//Ignore filter for Country while calculating Total Sales.

//You can add multiple filters

//Remove blanks

// Total Sales (All Countries)

Total Sales (All Countries) =

```
IF(  
    [Total Sales] = BLANK(),  
    BLANK(),  
    CALCULATE(  
        [Total Sales],  
        ALL('Sales Territory'[Sales Territory Country]) ) )
```

REMOVEFILTERS:

- Instead of ALL it can be used.
- Clear filters from specified tables or columns
- *You can add multiple filters to be removed, by adding more conditions*

Total Sales (All Countries) =

```
CALCULATE(  
    [Total Sales],  
    REMOVEFILTERS(SalesTerritory[Country]) )
```

Total Sales (All CountriesYear) =

```
CALCULATE(  
    [Total Sales],  
    REMOVEFILTERS(SalesTerritory[Country]) , REMOVEFILTERS('Date'[CalendarYear]) )
```

--Percent of Total ,of countries with respect to total sales.

Country Percent of Total Sales =
DIVIDE(
 [Total Sales],
 [Total Sales (All Countries)])

// You can see percentage total of countries in PowerBI using

show value as → Percent of grand total

Values → Total Sales → drop down → show value as → Percent of grand total

This shows percentage with respect to Grand total in Visual.

But DAX Expression will not be affected by **filter by country** from anywhere.

Total Sales (United States) =
CALCULATE(
 [Total Sales],
 SalesTerritory[Country] = "United States")

Total Sales (US & Canada) =
CALCULATE(
 [Total Sales],
 SalesTerritory[Country] = "United States" || SalesTerritory[Country] = "Canada")

Total Sales (US & Canada) =
CALCULATE(
 [Total Sales],
 OR(SalesTerritory[Country] = "United States" ,SalesTerritory[Country] = "Canada"))

Total Sales (US & Canada) =
CALCULATE(
 [Total Sales],
 (SalesTerritory[Country] in{ "United States" , "Canada"}))

Total Sales (2019) =
CALCULATE(
 [Total Sales],
 'Date'[CalendarYear]=2019)

ALLSELECTED

It lets the measure access filter context as it is visible outside the current visual.

Total Sales(selected) =
CALCULATE(
 [Total Sales],
 ALLSELECTED('Sales Territory'[Sales Territory Country]))

```
Country Percent of selected Total Sales =
DIVIDE(
    [Total Sales],
    [Total Sales(selected)] )
```

Calculated Columns Vs Measure

Calculated Columns	Measures
Creates a value for each row. So calculated on model.	Calculated on demand dynamically based on filters in Visualization
Stored in .pbix. Adds to disk space and potentially increase refresh time	Utilize CPU for calculations. No impact on memory
Table Name might be required	Table name not required in expression

Lab3: Get Percentage sales for different countries with respect to Total Sales.

1. Calculate Total Sales for ALL Countries ignoring the filter context

```
//Remove blanks
// Total Sales (All Countries)
Total Sales (All Countries) =
IF(
    [Total Sales] = BLANK(),
    BLANK(),
    CALCULATE(
        [Total Sales],
        ALL('Sales Territory'[Sales Territory Country]) ) )
```

2. Divide Total sales in filter context by the calculation created previously.

```
--Percent of Total ,of countries with respect to total sales.
Country Percent of Total Sales =
DIVIDE(
    [Total Sales],
    [Total Sales (All Countries)] )
```

3. Create Table Visual with Country, Total Sales, Country Percent of Total Sales and observe.

Role Playing dimensions:

- A **role-playing dimension** is a dimension that can filter related facts differently. For example, at Adventure Works, the date dimension table has three relationships to the InternetSales facts. The same dimension table can be used to filter the facts by order date, ship date, or delivery date.
- In a Power BI model, this design can be imitated by creating multiple relationships between two tables.

- There can only be one active relationship and other Inactive relationship between two Power BI model tables.
- Filter propagation happens through default relationship.
- To use inactive relationship as Filter define a DAX expression that uses the [USERELATIONSHIP function](#)
- USERELATIONSHIP function allows us to specify the columns of relationship which we want to activate.

Lab4: Create Measures to show

- **Sales based on shipdate**
 - **Sales based on Due Date**
1. Create Inactive Relationship between InternetSales[DueDate] and Date[FullDateKey]
 2. Create Inactive Relationship between InternetSales[ShipDate] and Date[FullDateKey]
 3. Create Calculation for Due date and Ship date as relationship is InActive
 4. Create Following Measures:

```
SalesByDueDate = CALCULATE([Total Sales],
USERELATIONSHIP('Date'[FullDateKey],InternetSales[DueDate]))
SalesByShipDate = CALCULATE([Total Sales],
USERELATIONSHIP('Date'[FullDateKey], 'InternetSales'[ShipDate]))
```

5. Create table visual to show Total Sales, SalesByDueDate, SalesByShipDate,CalenderYear.You can Slice by City or some other field to observe difference

Working With Calculated Table

Create Date Table Using Dax:

1. CALENDARAUTO:CALENDARAUTO() function returns a contiguous, complete range of dates that are automatically determined from your dataset.

The starting date is earliest date in your dataset, and the ending date is the latest date in your dataset .

You can pass in a single optional parameter that's the end month of the year by default it is considered as 12

You can give other value representing **Fiscal month**.

Example:

Dates = CALENDARAUTO(6)

2. CALENDAR: The CALENDAR() function returns a contiguous range of dates based on a start and end date that are entered as arguments in the function.

Example:

Dates = CALENDAR (DATE (2005, 1, 1), DATE (2015, 12, 31))

Dates = CALENDAR ("01/01/2005", "12/31/2015")

Dates = CALENDAR (MIN('Internet Sales'[Order Date]),MAX('Internet Sales'[Order Date]))

Note: These functions Returns a single-column table consisting of date values.

You can create other columns in date table using DAX calculation.

Lab 5: Create Date Table and Other Columns in Date Table

Option 1:

1. **Create Date Table:** Table View → Home → Calculations → New Table

DimDate = `CALENDAR (MIN('InternetSales'[OrderDate]),MAX('InternetSales'[OrderDate]))`

2. **Create Other Columns Using Calculations:** Select Dates Table → Table

View → Home → Calculations → New Column

```
Year = YEAR(Dates[Date])
MonthNum = MONTH(Dates[Date])
WeekNum = WEEKNUM(Dates[Date])
DayoftheWeek = FORMAT(Dates[Date], "DDDD")
Month = FORMAT(Dates[Date], "MMM")
MonthYear = FORMAT(Dates[Date], "mmm yyyy")
Month Year = FORMAT('Dates'[Date], "mm-yyyy")
```

Option 2:

3. Directly Create All Columns Together

Table view → Table tools → New Table →

```
DimCalender = ADDCOLUMNS(
    CALENDAR(MIN('InternetSales'[OrderDate]),MAX('InternetSales'[OrderDate])),
    "Month", month([Date]),
    "Day", day([Date]),
    "Year", year([Date]),
    "Quarter", year([Date]) & "-Q" & QUARTER([Date]),
    "MonthLongName", FORMAT([Date], "MMMM")
)
```

4. Remove DimCalender and DimDate Table.

Refer: <https://docs.microsoft.com/en-us/dax/format-function-dax>

Time Intelligence Functions

- Time-intelligence functions that enable you to manipulate data using time periods, including days, months, quarters, and years.
- You can compare and aggregate data over those periods.

TOTALYTD:

- Evaluates the specified expression over the interval which begins on the first day of the year and ends with the last date in the specified date column after applying specified filters.
- Works at qtr, month, day level.

TOTALQTD, TOTALMTD:

- Evaluates expression starting on first day of qtr and first day of month respectively

Lab6: Create Quick measure for Quarter to date Sale and Year to date Sale

1. Mark your Date Table as Date.
2. Select Date Table → Right Click → Mark as Date Table → Mark as Date Table

3. In Mark as date table window select

Date Column :FullDateKey

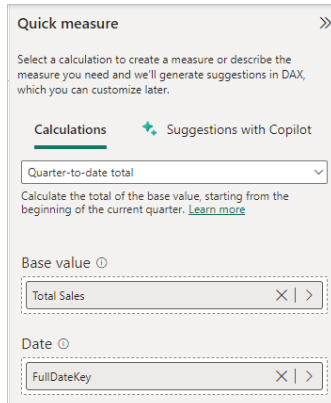
Click OK

4. Report View → Modeling Tab → Calculations → Quick measure →

Calculations:Quarter-to-date total


Base value:Total Sales

Date:FullDateKey



Quick measure

Select a calculation to create a measure or describe the measure you need and we'll generate suggestions in DAX, which you can customize later.

Calculations  Suggestions with Copilot

Quarter-to-date total

Calculate the total of the base value, starting from the beginning of the current quarter. [Learn more](#)

Base value ⓘ

Total Sales

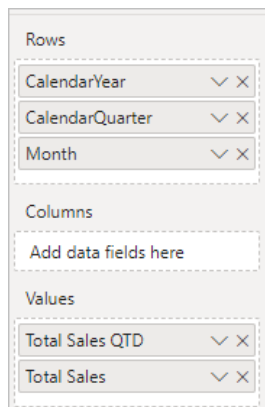
Date ⓘ

FullDateKey

→Add

5. The measure will be created as following:

6. Total Sales QTD =
`TOTALQTD([Total Sales], 'Date'[FullDateKey])`
7. Create Metrix Visual with following settings to verify.



Rows

CalendarYear

CalendarQuarter

Month

Columns

Add data fields here

Values

Total Sales QTD

Total Sales

8. Similarly Create Measure for Year to date sale by selecting

Calculation : Year-to-date-total

Base value:Total Sales

Date:FullDateKey

9. The measure will be created as following:

Total Sales YTD =
`TOTALYTD([Total Sales], 'Date'[FullDateKey])`

Lab 7: Using Calculate with Time Intelligence:

Calculate previous year Sales

Previous Year Sales = `CALCULATE([Total Sales],PREVIOUSYEAR('Date'[FullDateKey]))`

Calculate Previous respective time period(year/qtr/month) sales.

Prior Year Sales1 =
`CALCULATE([Total Sales],
SAMEPERIODLASTYEAR(
 'Date'[FullDateKey]))`

Calculate 2 month back sales to compare with current month context , using custom date shift with DateAdd Function .

Prev2monthSales = `CALCULATE([Total Sales],DATEADD('Date'[FullDateKey],-2,MONTH))`

Other Functions:

//Calculate Previous Year Sale using PARALLELPERIOD

PARALLELPERIOD: Returns a table that contains a column of dates that represents a period parallel to the dates in the specified **dates** column, in the current context, with the dates shifted a number of intervals either forward in time or back in time.

Last year sale = `CALCULATE([Total Sales], PARALLELPERIOD('Date'[FullDateKey],-1,YEAR))`

Calculate Previous Year Sale using custom date shift with DateAdd Function

Last year sale = `CALCULATE([Total Sales], DATEADD('Date'[FullDateKey],-1,YEAR))`

Semi-Additive Measure

Semi-Additive measures can be summarized across some dimension but not all. (Typically Time)

Lab8: Work with Semi-Additive Measures

Product Inventory = `SUM('ProductInventory'[UnitsBalance])`

//Incorrect result as it should not be summed up over time

Closing Balance = `CALCULATE([Product Inventory],LASTDATE('Date'[FullDateKey]))`

// LastDate: Returns the last date in the current context for the specified column of dates.

More Functions:

//LASTNONBLANK: Return for last date which has nonblank value

Closing Balance (Non Blank) =
`CALCULATE(
 [Product Inventory],
 LASTNONBLANK(`

'Date'[FullDateKey], [Product Inventory]))
Opening balance Month = OPENINGBALANCEMONTH([Product Inventory], 'Date'[FullDateKey])
Opening(NON BLANK) = CALCULATE([Product Inventory], //FILTER PORTION LASTNONBLANK(PARALLELPERIOD('Date'[FullDateKey], -1, MONTH),[Product Inventory])) //Opening balance for this month is closing balance for last month