# Calculated columns, Measures and Calculated Tables:

1. Calculated Columns:
   - These are like adding new columns to your existing table in Power BI.
   - You can create them using formulas that work on individual rows of data.
   - They're useful for making new data based on existing data in the same row.

2. Measures:
   - Measures are calculations that work across many rows of data, like finding totals, averages, or counts.
   - They're dynamic and change based on what you're looking at in your report.
   - You use measures to create summaries or key numbers for your analysis.

3. Calculated Tables:
   - These are like new tables you create based on calculations or filters from existing tables.
   - You use formulas to define what data goes into these tables.
   - They're handy for creating subsets of data or organizing data in a different way based on certain conditions.

# What is Date Table?

In Power BI, a Date table is a table that contains a column of dates, along with other columns that provide additional information about those dates, such as year, quarter, month, day of the week, and so on. A date table is an essential element of any Power BI data model that involves time-based analysis or reporting.

You can create powerful time-based calculations and analyses, such as year-to-date (YTD) totals, rolling averages, and comparisons between different periods.

# Different Ways to Create Date Table

- Auto Date/Time Table
- Create a Calendar table using DAX
- Create a Calendar table using Power Query
- Import Calendar Table

## CALENDAR() vs CALENDARAUTO() | Calendar Function in Power BI

In DAX, both the CALENDAR() and CALENDARAUTO() functions are used to create date tables that can be used for time intelligence calculations. However, they differ in the way they generate the dates in the calendar table.

# Calendar() Function in DAX

The CALENDAR() function is used to create a custom calendar table that includes a range of dates that are specified by the user. The syntax for the CALENDAR() function is as follows:

**CALENDAR( start_date, end_date )**

where start_date and end_date are date values that define the start and end of the date range for the calendar table.

## CalendarAuto() DAX Function

CALENDARAUTO() function is used to automatically generate a calendar table based on the dates present in a date column in the data model. The CALENDARAUTO() function determines the start and end dates of the date column and generates a calendar table that includes all dates within that range. The syntax for the CALENDARAUTO() function is as follows:

CALENDARAUTO()

## How to Create a Simple Date Table in Power BI using DAX

Here are the steps to create a date table in Power BI using DAX:

- Open Power BI Desktop and click on the “Modeling” tab.
- Click on “New Table” to create a new table.

In the formula bar, enter the following DAX formula to create a date table:

```
DAX DateTable =
ADDCOLUMNS (
    //CALENDAR(DATE(2020,1,1), DATE(2024,12,31)),
    CALENDARAUTO(),
    "Year", YEAR([Date]),
    "Quarter", "Q" & FORMAT(CEILING(MONTH([Date])/3, 1), "#"),
    "Quarter No", CEILING(MONTH([Date])/3, 1),
    "Month No", MONTH([Date]),
    "Month Name", FORMAT([Date], "MMMM"),
    "Month Short Name", FORMAT([Date], "MMM"),
    "Month Short Name Plus Year", FORMAT([Date], "MMM,yy"),
    "DateSort", FORMAT([Date], "yyyyMMdd"),
    "Day Name", FORMAT([Date], "dddd"),
    "Details", FORMAT([Date], "dd-MMM-yyyy"),
    "Day Number", DAY ( [Date] )
)
```

- You can adjust the start and end dates to fit your specific needs.
- Name the table “Date” or any other appropriate name.
- Click on the “New Column” button in the “Modeling” tab to add a new column to the “Date” table.
- Select the Date Column and change the data type to Date.
- In the “Table Tools” tab, click on the “Mark as Date Table” button

## How to Create Fiscal Date Table in Power BI using DAX

Here are the steps to create a Fiscal Date Table in Power BI using DAX:

- Open Power BI Desktop and click on the “Modeling” tab.
- Click on “New Table” to create a new table.

In the formula bar, enter the following DAX formula to create a Fiscal date table:

```
Fiscal DAX DateTable =
VAR FISCALMONTHSTART = 4
RETURN
ADDCOLUMNS (
    CALENDARAUTO (FISCALMONTHSTART - 1 ),
    "Year", YEAR ( [Date] ),
    "Month", MONTH ( [Date] ),
    "Month Name", FORMAT ( [Date], "MMMM" ),
    "Month Short Name", FORMAT ( [Date], "MMM" ),
    "Month + Year Short", FORMAT ( [Date], "M/yy" ),
    "Year No + Month Number", FORMAT ( [Date], "yyyyMM" ),
    "Quarter", CEILING ( MONTH ( [Date] ), 3 ),
    "Quarter No", CEILING ( MONTH ( [Date] ), 3 ) / 3,
    "Day Name", FORMAT ( [Date], "dddd" ),
    "Day Number", DAY ( [Date] ),
    "Fiscal Year", IF ( MONTH ( [Date] ) >= FISCALMONTHSTART, YEAR ( [Date] ), YEAR (
[Date] ) - 1 ),
    "Fiscal Quarter", CEILING ( MONTH ( EDATE ( [Date], - FISCALMONTHSTART + 1 ) ), 3 ) /
3,
    "Fiscal Month Number", MONTH ( EDATE ( [Date], - FISCALMONTHSTART + 1 ) )
)
```

- Select the Date Column and change the data type to Date.
- In the “Table Tools” tab, click on the “Mark as Date Table” button
- Once you have completed these steps, you will have a fully functional date table in Power BI

## How to Create Date Table in Power BI using Power Query

To create a date table in Power Query, follow these steps:

- Open Power BI Desktop and click on the “Transform data” button to open Power Query Editor.
- In Power Query Editor, click on the “Home” tab, and then click on “New Source” and select “Blank Query.”
- Rename the query by double-clicking on “Query1” in the “Queries” pane, and typing in a new name, such as “DateTable.”

Open advance editor and paste below Code:

```
let
    // Set the start and end date for the date table
    StartDate = #date(2020, 1, 1),
    EndDate = #date(2021, 12, 31),

    // Create a list of dates from the start to end date
    DateList = List.Dates(StartDate, Duration.Days(EndDate - StartDate) + 1, #duration(1,0,0,0)),

    // Convert the list to a table and add columns for year, month, day, etc.
    DateTable = Table.FromList(DateList, Splitter.SplitByNothing(), {"Date"}),
```

```
    #"Changed Type" = Table.TransformColumnTypes(DateTable,{{"Date", type date}}),
    #"Inserted Year" = Table.AddColumn(#"Changed Type", "Year", each Date.Year([Date]),
Int32.Type),
    #"Inserted Month" = Table.AddColumn(#"Inserted Year", "Month", each Date.Month([Date]),
Int32.Type),
    #"Inserted Day" = Table.AddColumn(#"Inserted Month", "Day", each Date.Day([Date]),
Int32.Type),
    #"Inserted Weekday" = Table.AddColumn(#"Inserted Day", "Weekday", each
Date.DayOfWeek([Date]), Int32.Type),
    #"Inserted MonthName" = Table.AddColumn(#"Inserted Weekday", "MonthName", each
Date.ToText([Date], "MMMM"), type text),
    #"Inserted MonthNameShort" = Table.AddColumn(#"Inserted MonthName",
"MonthNameShort", each Date.ToText([Date], "MMM"), type text),
    #"Inserted QuarterNo" = Table.AddColumn(#"Inserted MonthNameShort", "QuarterNo", each
Date.QuarterOfYear([Date]), Int32.Type),
    #"Inserted YearQuarter" = Table.AddColumn(#"Inserted QuarterNo", "YearQuarter", each
Text.Combine({Text.From([Year]), " Q", Text.From([QuarterNo])}), type text),
    #"Inserted YearMonth" = Table.AddColumn(#"Inserted YearQuarter", "YearMonth", each
Text.Combine({Text.From([Year]), "-", Text.PadStart(Text.From([Month]), 2, "0")}), type text),
    #"Inserted YearMonthDay" = Table.AddColumn(#"Inserted YearMonth", "YearMonthDay",
each Text.Combine({Text.From([Year]), Text.PadStart(Text.From([Month]), 2, "0"),
Text.PadStart(Text.From([Day]), 2, "0")}), type text),
    #"Added Custom Column" = Table.AddColumn(#"Inserted YearMonthDay", "Month,Year",
each Text.Combine({[MonthNameShort], ",", Text.Middle(Text.From([Year], "en-IN"), 2)}), type
text),
    #"Inserted Merged Column" = Table.AddColumn(#"Added Custom Column", "Quarter", each
Text.Combine({"Q", Text.From([QuarterNo], "en-IN")}), type text)
in
    #"Inserted Merged Column"
```

- Close and apply the changes to the query by clicking on the “Close & Apply” button.
- Once you have completed these steps, you will have a fully functional date table in Power BI using Power Query.

## Implementing DAX Calculations

We will use Data Analysis Expressions (DAX) to create calculated columns, measures, and calculated tables to perform complex calculations and aggregations. DAX is a powerful formula language that allows you to manipulate data within Power BI.

```
//Measures Total Sales
Sales = SUM(Sales_Data[Sales])
//Measures Previous Year Toal Sales
Sales PY = CALCULATE([Sales], SAMEPERIODLASTYEAR(DateTable[Date]))
//Diffrence Between Current Year Sales & Previous Year Sales
Sales vs PY = [Sales] - [Sales PY]
//Percentage Increase or Decrease in sales year on year (YOY%)
Sales vs py % = DIVIDE([Sales vs PY],[Sales],0)
```

```
>> Products Sold = SUM(Sales_Data[Order Quantity])
>> Profit = SUM(Sales_Data[Profit])
>> Profit LY = CALCULATE([Profit], SAMEPERIODLASTYEAR(DateTable[Date]))
>> Profit Vs LY = [Profit]- [Profit LY]
>> Profit vs LY % = [Profit Vs LY]/[Profit]
>> Profit Margin = DIVIDE([Profit],[Sales],0)
>> Total Cost = SUM(Sales_Data[Total Cost])
```