

POWER BI

Power BI is nothing but Power Business Intelligence and it is one of the reporting tools designed and developed by Microsoft Corporation in 2013.

Before 2013, we have another reporting tool from Microsoft side is SSRS. There are some drawbacks in SSRS. With this we can connect with only 2 data sources. One is SQL Server and another one is Oracle.

It has another problem like having less no of visuals for building a report. To overcome these drawbacks Microsoft introduced new tool called “**POWER BI**”.

It is easiest and simplest tool to build reports by using visuals to show the data into a graphical representation and it is just like an Excel application.

The functions which are available in Excel, same kind of functions and options are present in Power BI as well.

The main advantage of this tool is that it has all the options and icons in a single page.

It has more than 200+ visuals to build a report and by using this tool we can easily connect with different data sources like SQL Server, Oracle, Azure, Sales force, IBM Netezza, Json, Text, PDF, Excel, Files etc.

Power BI has software in different ways like below

- ✓ Power BI Desktop
- ✓ Power BI Service

We need to install Power BI Desktop, because it is a development environment where we can build our reports.

That means we need to build reports in our system and publish into the service.

Installing Power BI Desktop Software:

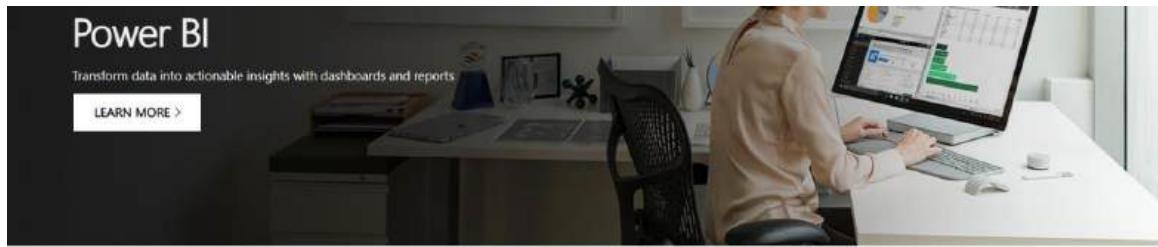
Go to Google and type “**Download Power BI Desktop**” and press the enter button.



Now click on “**Microsoft Power BI Desktop**” option.

Google search results for "download power bi desktop". The top result is "Downloads - Microsoft Power BI". A green box highlights the "Download" button on the Microsoft Power BI Desktop page.

Click on “Download” option.



Microsoft Power BI Desktop

Important! Selecting a language below will dynamically change the complete page content to that language.
Select Language: English Download

Select “**PBIDesktopSetup.x64.exe**” files and click on “**Next**”

Choose the download you want

File Name	Size
<input checked="" type="checkbox"/> PBIDesktopSetup_x64.exe	386.4 MB
<input type="checkbox"/> PBIDesktopSetup.exe	348.0 MB

Download Summary:
1. PBIDesktopSetup_x64.exe
Total Size: 386.4 MB

Next

Note: Here you have to select executable file according to your system type.

Right click on “**This PC**” or “**My computer**”. Then go to “**Properties**”. You will find system type.

LAPTOP-2BMPN47J
HP 250 G8 Notebook PC

(i) Device specifications

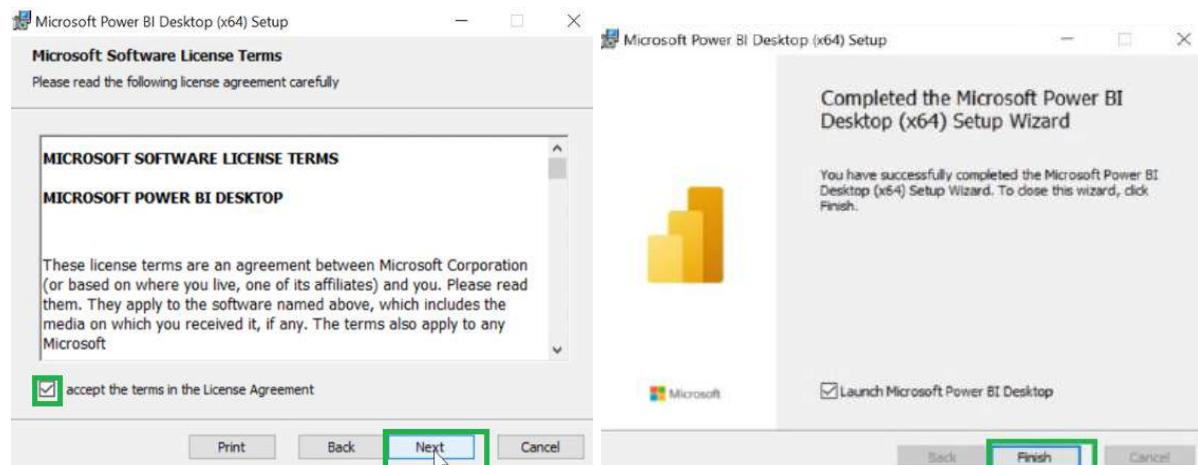
Device name	LAPTOP-2BMPN47J
Processor	11th Gen Intel(R) Core(TM) i3-1115G4 @ 3.00GHz 3.00 GHz
Installed RAM	8.00 GB (7.75 GB usable)
Device ID	D1B3B8A1-DD83-476A-980B-FB07FEC2E04B
Product ID	00356-24523-13333-AAOEM
System type	64-bit operating system, x64-based processor
Pen and touch	No pen or touch input is available for this display

My system Type is **64-bit operating system, x64-based processor**. So, I have selected **64 bit executable** file.

Go to downloads and click on “**PBIDesktopSetup.x64.exe**” file and click on “**Next**”

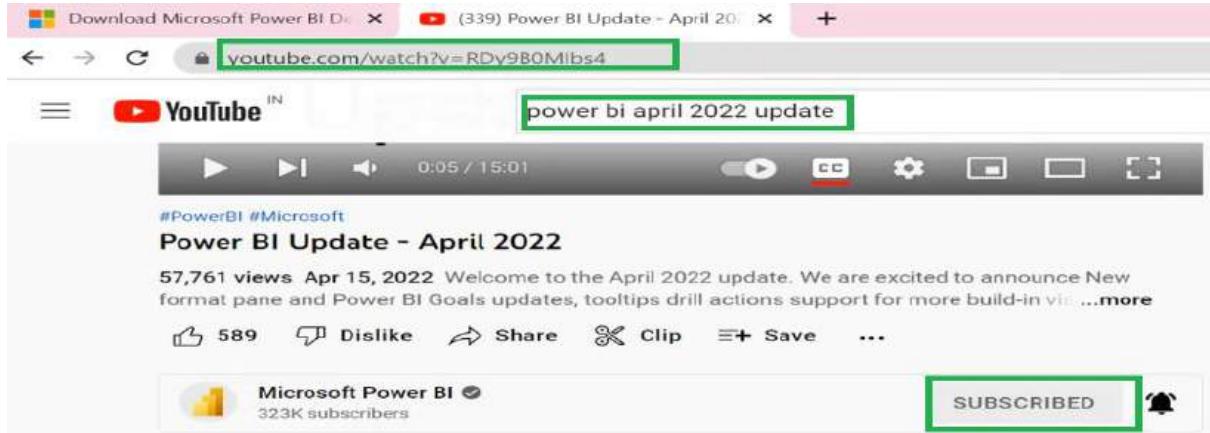


Next put tick mark in check box of license agreement and click on “**Next**”. Finally click on “**Finish**” button



Note: Every month try to download latest software and install it. How you know whether your software is latest one or not. Go to **YouTube** and type “**Power bi April**

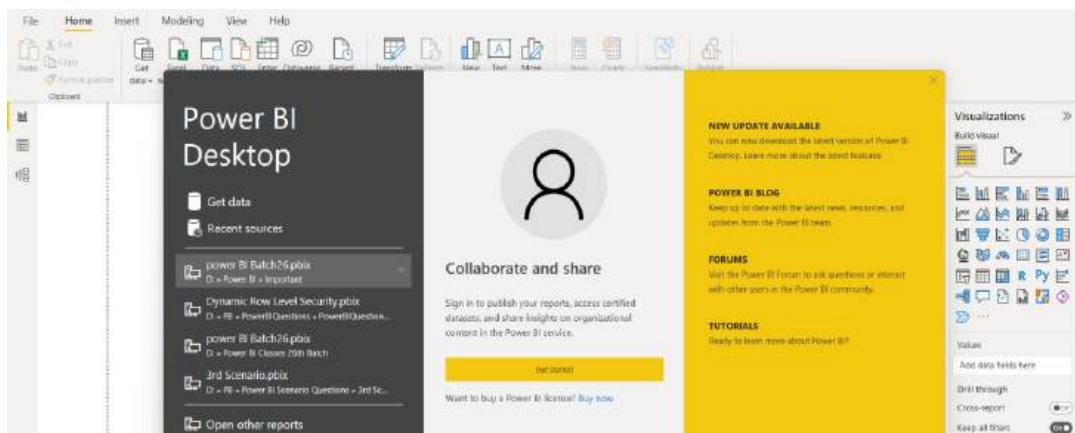
2022 update". Like this, then you can see the latest one. Click on "Subscribe" button.



Link for power bi desktop software:

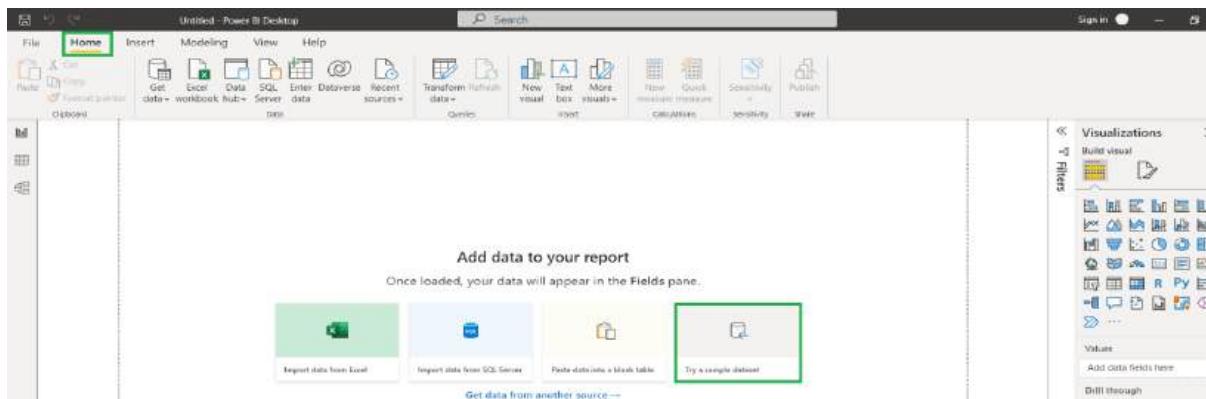
<https://www.microsoft.com/en-us/download/details.aspx?id=58494>

Once installation is done, you can now open and see the below screen. Close the "Get started" window.

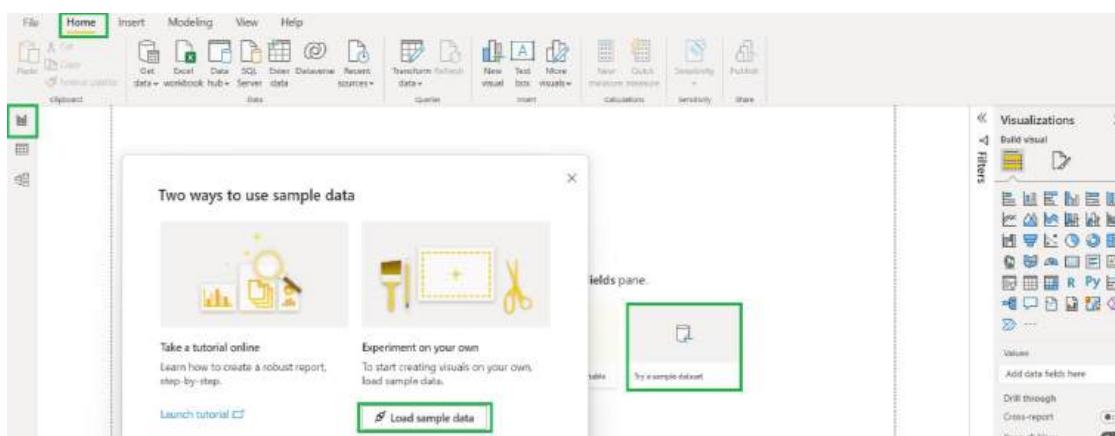


For this we will discuss in later modules. If you close that window, you will see the first page of the development environment. Here you can see bunch of Microsoft ribbons like "**File**", "**Home**", "**Insert**", "**Modeling**", "**View**", "**Help**".

If you want to work with Power BI, you will need the data right. So, for the first time you have to load the sample data set. By default Microsoft itself will provide the sample dataset.



We will get four options in the center part of the screen for the first time only. We need to click on the “**Try a sample dataset**” option to load the sample data.



When you click on “**Load sample data**” we will get below screen and you have to select particular data set and click on “**Load**” option. Automatically that data set will be loaded into power BI and visible under the “**fields**” option.

The screenshot shows the Power BI Desktop interface with the 'Home' tab selected. On the left, the 'Navigator' pane is open, showing a tree view of data sources. Under 'Financial Sample.xlsx [2]', the 'Financials' item is selected and highlighted with a green border. To the right, a preview of the 'financials' table is displayed, showing columns: Segment, Country, Product, Discount Band, and Unit. The data includes rows for various countries and segments like Canada, Germany, France, Mexico, etc.

If you expand the particular data source you will get column names.

The screenshot shows the Power BI Fields pane. At the top, there are tabs for Filters, Visualizations, and Fields. The Fields tab is selected. A search bar is at the top right. Below it, the dataset name 'financials' is shown along with its storage mode (Import) and last refresh time (9/27/2022, 5:38:55 AM). A 'Search' button is also present. The main area displays the fields: Sales (sum), COGS (sum), Country, and Date.

Once data is loaded we will go to the left side tabs to build a report. In the left pane we have 3 tabs.

- **Report**
- **Data**
- **Model**

Report tab: Whatever the data we have, with that data will build a report by using different visuals. (Take any visual and design the report)

Data tab: Whatever the data set under field's option that corresponding data will be visible in the table format. At a time we can see only single data set data.

Model tab: To provide a relationship between data sets we will use model tab and we can hide or unhide some columns whenever required.

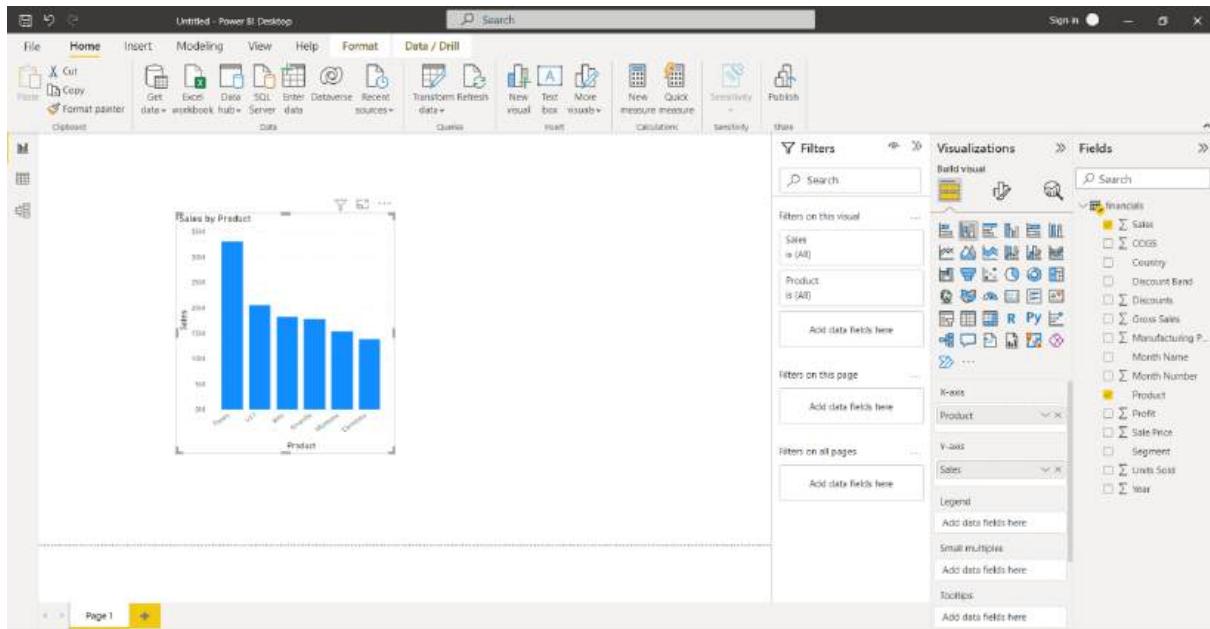
Note: We can hide or unhide the columns either in “**Model Tab**” or “**Data Tab**”. To minimize the memory usage, we will hide some columns. We can't do in report tab.

The screenshot shows the Power BI Model tab. It displays the same 'financials' dataset with the same four columns: Sales, COGS, Country, and Date. The 'COGS' column is highlighted with a gray background, indicating it is selected or being edited.

We can't hide or unhide the multiple columns in “**Data**” tab. Only we can do in Model tab itself. But you can see the hidden columns in “**Data**” tab.

Note: For selecting multiple columns we have to use [ctrl+ select no of columns]

Sample Report



Take any one of the visual from visualizations section, select a dataset from right side, and put some columns in **x-axis** and **y-axis** (Drag columns from the data set from the fields section). Then you can see the sample visual.

Filters Pane:

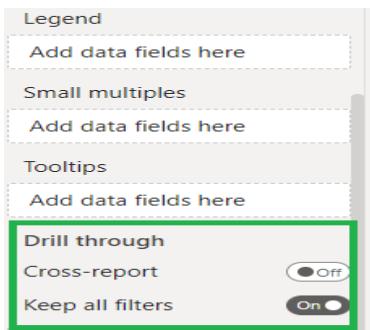
In filters pane, we can filter the data based on the requirement. There are 5 filters in this section. By default we have only 3 filters, and by selecting any visual we can see rest of the 2 filters.

- **Filters on this page**-----→Default
- **Filters on all pages**-----→Default
- **Filters on this visual**
- **Drill through**-----→Default
- **Drill-Down and Drill-Up**

If we want to see the Drill Down filter we have to put multiple columns in x-axis.



If we want to see the drill through filter you can scroll down near the x-axis and y-axis section.



Visualizations:

For representing the business data in the graphical or image format we will use visualizations. It is difficult to understand the data by seeing that in a table format or text format. But we can easily understand by preparing the reports in image (Visual) format.

In Visualizations pane, we have 3 options

- Build visual
- Format Visual
- Analytics

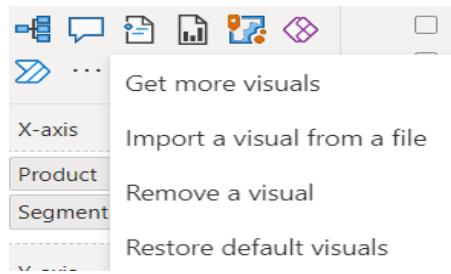
In “**Build visual**” option we can select particular visual to create report where as if we want to change the design part for the particular visual we can use “**Format visual**” option, there you can add colors to the columns and background. Change header and size of the text.

If you want to add further analysis to your visual, you have to use “**Analytics**” option.

The image contains three side-by-side screenshots of the 'Visualizations' pane in Power BI.
 - The left screenshot shows the 'Format visual' tab selected, with icons for grid, chart, and search. Below is a search bar and tabs for 'Visual' (highlighted with a yellow box) and 'General'. Under 'Visual', there are sections for 'X-axis' (on), 'Y-axis' (on), and 'Legend' (off).
 - The middle screenshot shows the 'General' tab selected. It includes a search bar, tabs for 'Visual' (highlighted with a yellow box) and 'General' (highlighted with a yellow box), and sections for 'Properties', 'Title' (on), 'Effects', 'Header icons' (on), 'Tooltips' (on), and 'Alt text'.
 - The right screenshot shows the 'Analytics' tab selected. It includes a search bar and sections for 'Constant line' and 'Error bars'.

Whatever the visuals we can see in the power bi window, those are called as default visuals. If we want to get more visuals we have to click on the dots mentioned under the default visuals. Then you will get 4 options. Those are

- **Get more visuals**
- **Import a visual from a file**
- **Remove a visual**
- **Restore default visuals**



Other than default visuals all the visuals are placed in the “**Market Store**”. Those visuals are called custom visuals. If we want, we have to connect with the market store and add that visual.

Get more visuals: It will ask the credentials. Once credentials supplied and click on **OK** button, it will show you lot of custom visuals. From there itself you can select required visual and add into the power BI.

Note: If you want to get more visuals, you have to connect with market store, for that we require organization credentials. That means you have to supply organization mail id and password. Personal mail id and password will not work here.

For practice purpose, I am creating a temporary mail id to work as an organization mail.

Creating temporary mail id and password:

Go to “**Google**” and type “<https://powerbi.microsoft.com/en-us/>” and press “**Enter**” button in your keyboard. Then click on “**Try free**” option on the top right corner of the window.

A new window will be opened. There you have to provide your mail id. Right now, you don't have any mail id.

Power BI

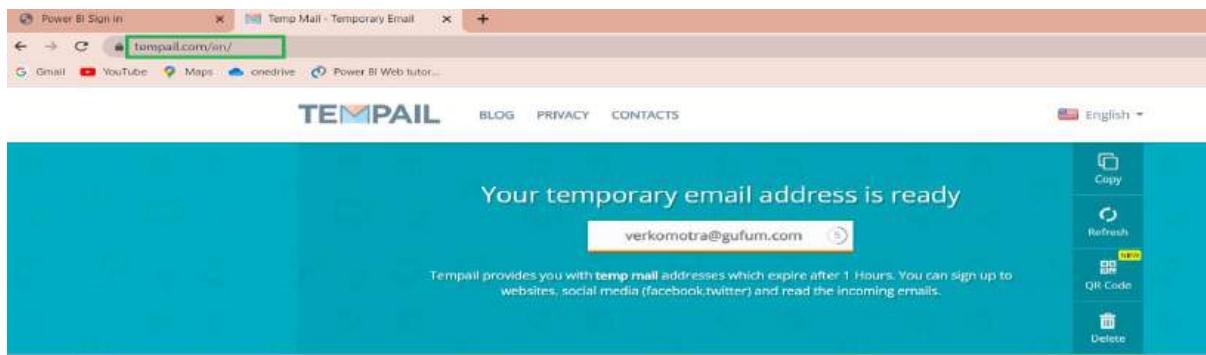
Enter your email, we'll check if you need to create a new account.

Email

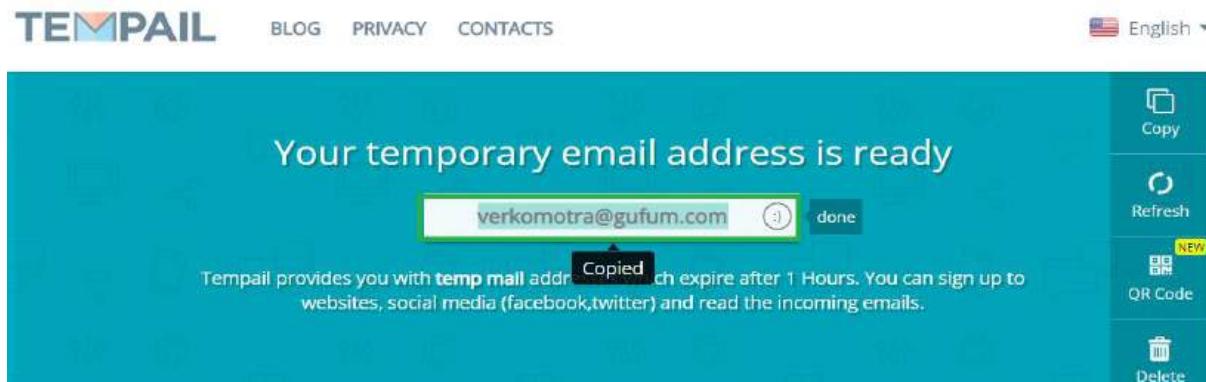
By proceeding you acknowledge that if you use your organization's email, your organization may have rights to access and manage your data and account. [Learn more about using your organization's email](#).

By clicking submit, you agree to these [terms and conditions](#) and allow Power BI to get your user and tenant details. [Microsoft Privacy Statement](#).

That's why you have to create that mail id. For that again click on “**New tab**” in “**Google Chrome**” and type “**Tempail.com**” and then press on “**Enter**” button in your keyboard.



Click on the random mail id displayed in the center part of the screen, automatically that mail id will be copied and then paste that it in the “**Power bi**” sign up window. Click on “**Submit**” button.



Power BI

Enter your email, we'll check if you need to create a new account.

Email

By proceeding you acknowledge that if you use your organization's email, your organization may have rights to access and manage your data and account. [Learn more about using your organization's email](#).

By clicking Submit, you agree to these [terms and conditions](#) and allow Power BI to get your user and tenant details. [Microsoft Privacy Statement](#).

Another window will be opened. In that window, select “I got it from organization” option and click on “Next”.

You've selected Microsoft Power BI

1 Let's get you started

Looks like you need to create a new account.

Microsoft Power BI is designed to be used by people collaborating within an organization, so your email will be visible to others who also use @gufum.com email addresses to signup

For that reason, emails from shared email services like outlook.com should not be used.

What kind of email is **verkomotra@gufum.com?**

I got it from my organization

It's my personal email

Next

Enter the “First Name” and Last Name” and select the “Country/Region” from the drop down menu. Select “India”.

2 Create your account

First name	Last name	Guinea
Vijay	Ramaraju	Guinea-Bissau
Country or Region	Guyana	Haiti
India	Heard Island and McDonald Islands	Honduras
Business phone number	Hong Kong SAR	Hong Kong SAR
	Hungary	Hungary
Email	Iceland	Iceland
verkomotra@gufum.com	India	India
Enter a password to sign in to your account.	Indonesia	Indonesia

Enter the phone number in “Business phone number” text box and enter the password in “Create password” tab as desired. And again enter the password in “Confirm password” option. Automatically, verification code will be sent to the “Tempail.com” inbox.

Business phone number	9999999999
Email	verkomotra@gufum.com
Enter a password to sign in to your account.	
Create Password	*****
Confirm password	*****
We've sent a verification code to verkomotra@gufum.com . Enter the code to complete sign up.	
Verification code	
Resend code	

Go to “Tempail.com” and copy the power bi verification code and paste it in the “Verification” code text box in “Power Bi” sign up window.

The screenshot shows a temporary email address generated by Tempail: Verkomotra@gufum.com. The subject of the email is "B61084 is your Microsoft Power BI verification code". The verification code is B61084, which is highlighted with a green checkmark. There is also a "Resend code" button. Below the email preview, there is a section for agreeing to terms and conditions and a privacy statement. At the bottom, there are "Next" and "Back" buttons.

Once verification code given, then click on “**Next**” button. And then click on “**Sign in**” to confirm the details.



Then new window will be opened. Click on “**Ask Later**” option in that window. Again new window will be opened. Click on “**Yes**” button.

Action Required

Security defaults are turned on to keep your organization secure. Set up the Microsoft Authenticator app to use two-step verification.

[Use a different account](#)

[Learn more about the Microsoft Authenticator app](#)

You have 14 days until this is required.

Ask later **Next**

Stay signed in?

Do this to reduce the number of times you are asked to sign in.

Don't show this again

No **Yes**

Finally click on “Get started” option in the window. Automatically it will login in to the “Power BI Service”.

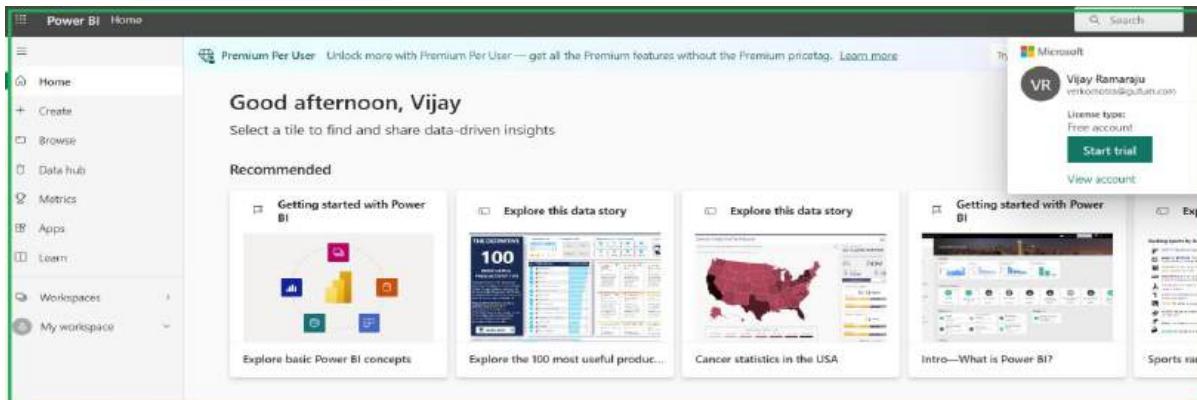
You've selected Microsoft Power BI

- 1 Let's get you started
- 2 Create your account
- 3 Confirmation details

Thanks for signing up for Microsoft Power BI

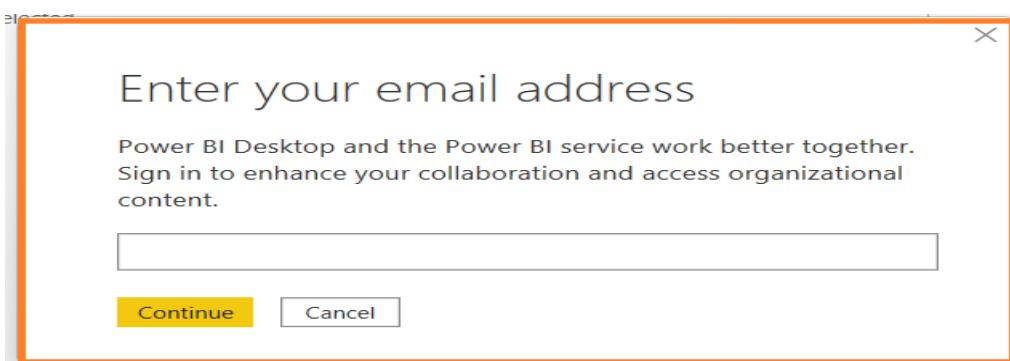
Your username is **verkomotra@gufum.com**

Get Started

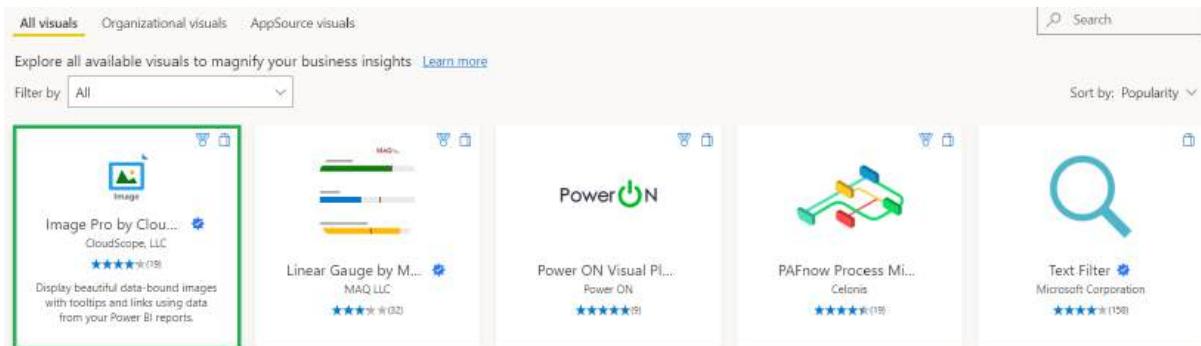


Note: If you want to work with “Power BI Service” then you can use the credentials we have created just now. If you want to get more visuals from market store also, you can use these credentials.

Navigation to get more visuals: Go to “Get more visuals” -> “Enter your Email Address” that you have created earlier -> Click on “Continue” -> “Enter Password” that you have generated earlier -> Click on “Sign in” -> Select required “Visual” -> Click on “Add” option. You can see the custom visual in the “Power BI” window.

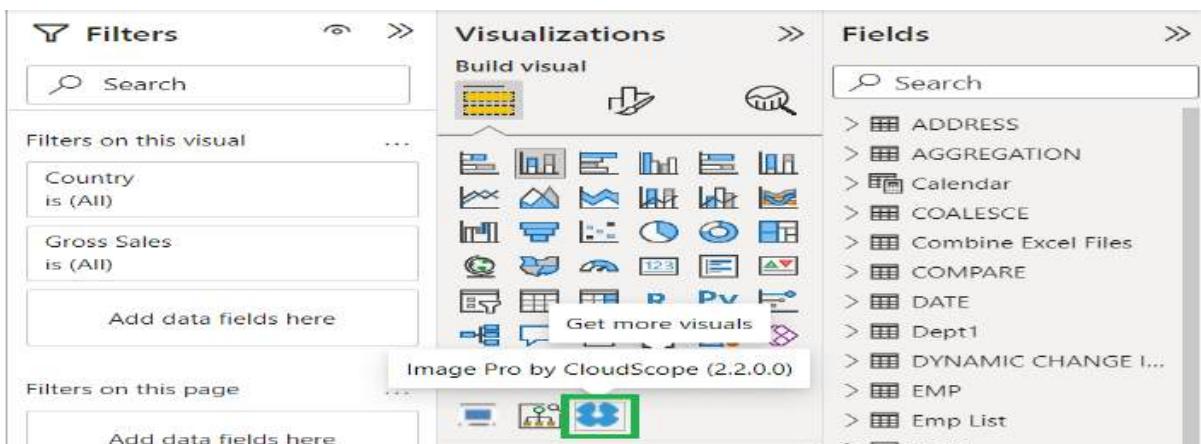
A screenshot of a dialog box with a green border. It has the title "Enter password" and a text input field labeled "Password". Below it is a link "Forgot my password" and a blue "Sign in" button at the bottom.

If you click on “**Add**” symbol, then that visual will be automatically present in the power BI desktop, or else if you click on “**download Sample**” option. That visual will be downloaded in Power BI file format and it will show you how to use that visual.



This is a detailed view of the 'Image Pro by CloudScope' visual card. It includes the visual icon, name, developer, rating, PBI Certified badge, overview, and ratings/reviews section. The overview describes the feature of displaying data-bound images with tooltips and links.

Custom Visual: Other than default visuals are called custom visuals.



Most of the times organization will not give permission to connect to the market store. In that case developers will download all the visuals from the market store once at a time and save it into the folder. From there itself we have to use those visuals. In that case we need to go for import a visual from a file option.

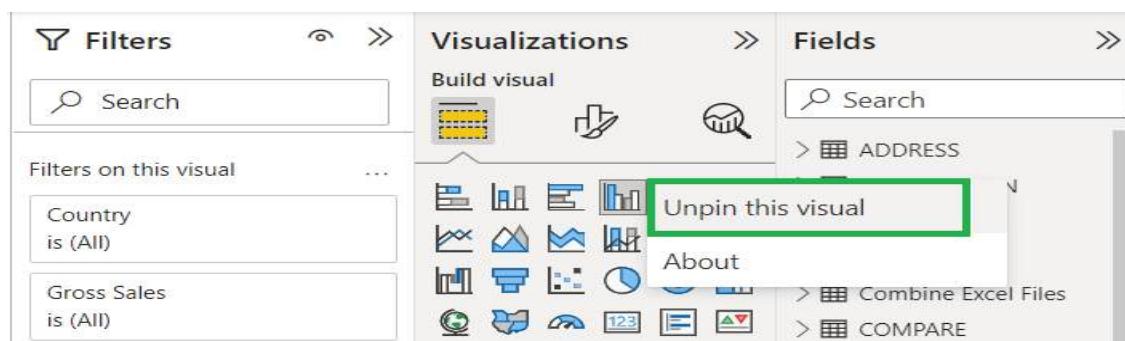
Import a visual from a file: It is also one type of getting custom visual but first we need to download a visual from a market store to the local system and there itself we need to import to get that visual to the power BI.

Navigation: Go to →Google-->Type “**Power BI Visuals download**” --> Click on “**Get it now**” option on the custom visual.

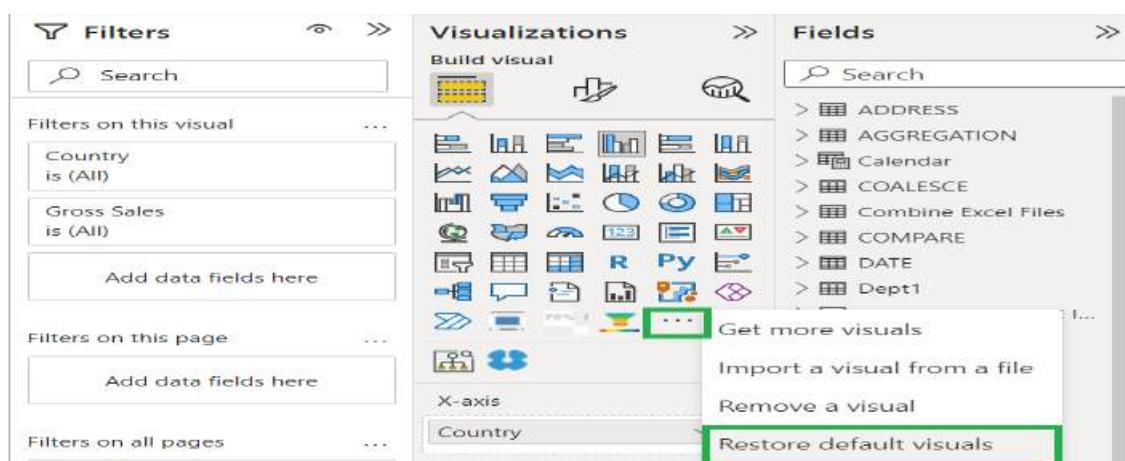
[Link](https://appsource.microsoft.com/en-us/marketplace/apps?product=power-bi-visuals):<https://appsource.microsoft.com/en-us/marketplace/apps?product=power-bi-visuals>

Remove a Visual: We can't remove the default visuals. We can only remove the custom visuals. Why because the default visuals are stored in power BI itself. Those are not stored in any other location. If we delete the default visual, we can't get back it again. That's why default visuals will not be removed. We can remove multiple custom visuals at a time by pressing on **CTRL** button.

Restore default visuals: Sometimes we don't want to show default visuals to the client to hide the process of building a report (That means what type of visuals we have used to create a report). In that case we have to use “**unpin this visual**” option by right clicking on the particular default visual. Then it will comes under custom visual place.



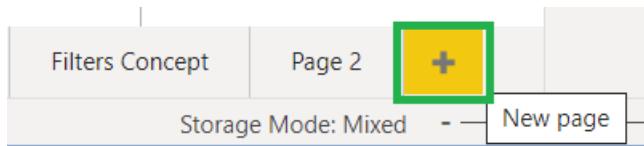
If we want to restore it back to the original place we need to click on the “**Restore default visuals**” option. Then automatically these visuals will be placed in their respective original places.



- If we want to take any visual to the development window, first double click on any visual from the default visuals or custom visuals. Then drag the columns from the selected dataset to the “**columns**” option under the visualization pane.
- Then you can see x-axis (Horizontal line), y-axis (Vertical Line), and legend options.

Pages:

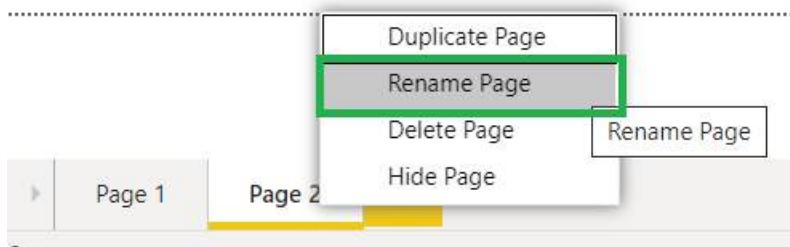
- **Creating new page:** Click on the ‘+’ button on the right side of the first page



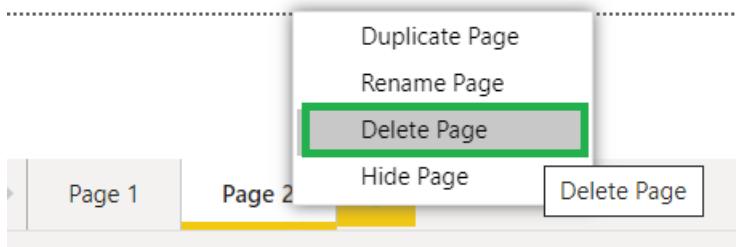
- **Creating duplicate page:** Right click on the particular page select “**duplicate page**” option



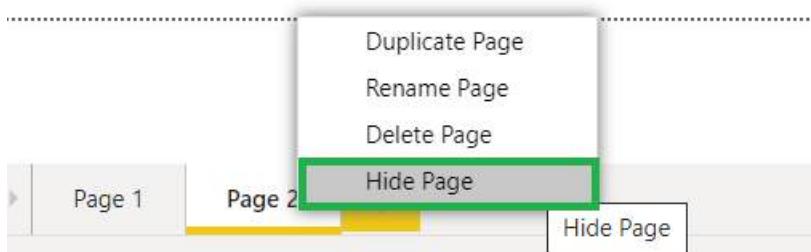
- **Renaming the page:** Double click on the page or Right click on the page and select “**Rename page**” option



- **Delete a page:** Click on the ‘X’ button available on the top right corner of the page or Right click on the page and select “**Delete page**” option



- **Hide a page:** Right click on the page and select “**Hide page**” option



- ✓ Whenever modifications required, at that time take duplicate page and do the modifications on that page.
- ✓ If you don't want to show a particular page to the user after publishing the report to the service then you can use "**Hide page**" option.
- ✓ If you want to publish a report click on "**publish**" option. Symbol shown like below



- ✓ In Power BI we have some options on the top of the file called "**Ribbons**" as shown below



❖ **File Ribbon:**

New: Here new power bi file will be opened.

Open Report: Recently opened files will be visible here and file location along with extension (.pbix)

Save: It will save the file after completion of the operation

Save as: It will ask the location where to save a file

Get data: It is useful to fetch data set from different data sources like SQL Server, Oracle, Azure, IBM Netezza etc.

Import: It is useful to import any file from a particular location that means it is useful for reference. It has below options

- ✓ Power BI template
- ✓ Power BI visual from file
- ✓ Power BI visual from AppSource
Power Query, Power Pivot, Power View (Components)

Power BI template: It is used for reference for the future requirement (Reusability) For example we have created a report with 2021 year and exported as a template for future use. That file may be imported in this option and will use this report for 2022 data. Automatically 2022 data will be pointed by the same visual.

Power BI visual from file: It is just like an "**import a visual from a file**" option (getting custom visual from a file).

Power BI visual from AppSource: It is just like a "**Get more visuals**" option (getting custom visual from market store)

Components of the Power BI:

Power Query: It is nothing but transforming data. i.e cleansing data, removing blank spaces. Etc.

Power Pivot: The process of getting data source from different place to the power BI model.

Power View: After completion of preparing report and viewing that report is called power view.

Export: It is nothing but exporting a power bi file into template form or in **PDF** form.

Power BI template-> creating a template for future use

Export to PDF-> We can share power bi file as **PDF** for the users

Publish: By using this option we can publish the generated report to the service for sharing different users.

Options and Settings: Just like settings for the application. There are different options to enable whenever required.

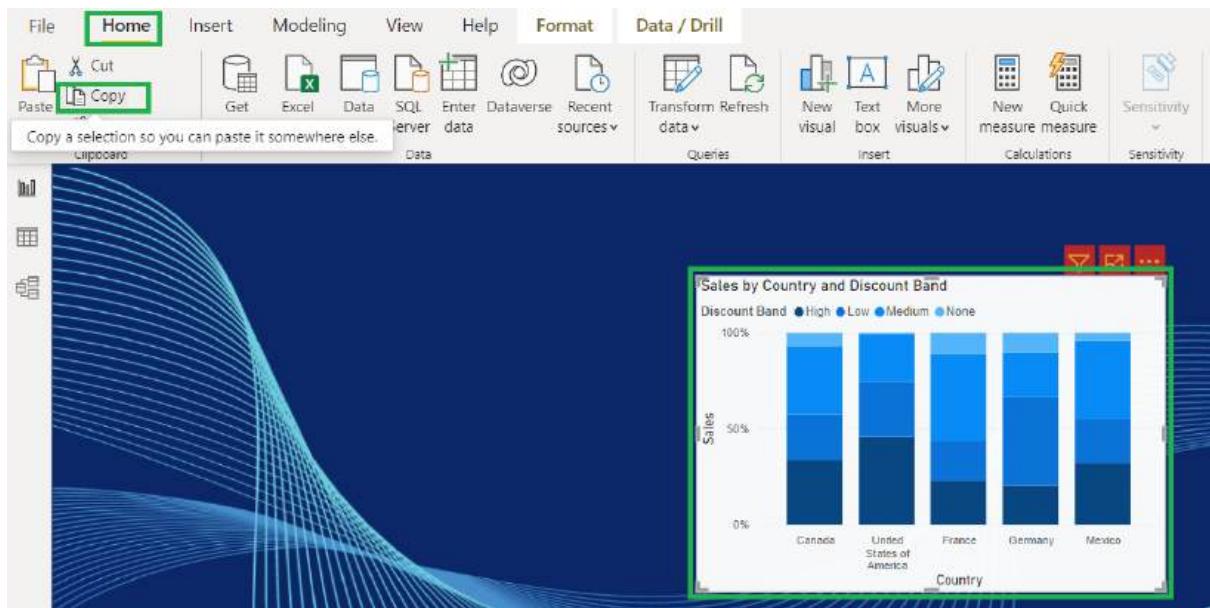
Get Started: First time power BI file will be opened and the recently used files visible there and the default videos for the reports are available in this option.

❖ Home Ribbon:

We have options available like **Cut, Copy, Paste and Format Painter** in visual level instead of data. By using these options we can cut or copy the visual and paste into different pages.

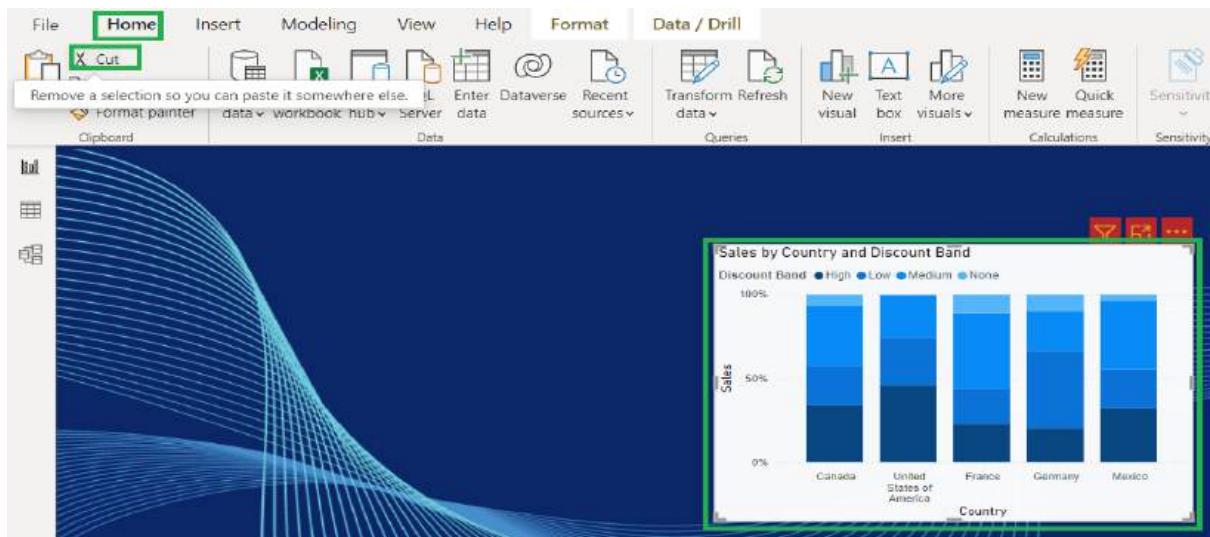
Copy: Select the visual and click on “**Copy**” option in the “**Home**” ribbon. So that selection will be copied to the internal memory to paste somewhere else.

Or simply you can use “**CTRL+C**” option to copy the selected object.

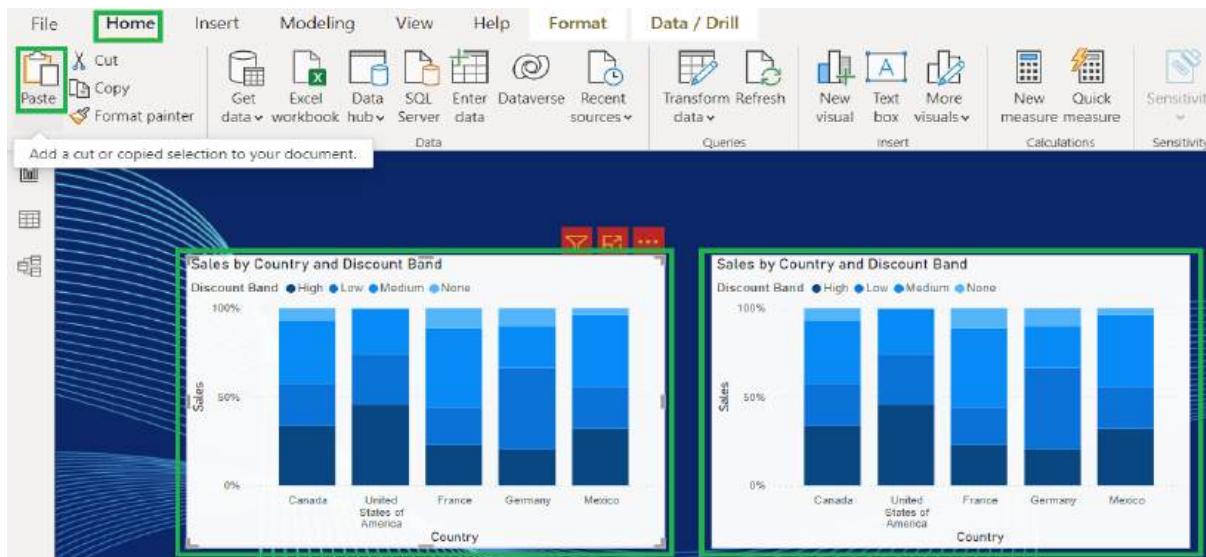


Cut: Select the visual and click on “**Cut**” option in the “**Home**” ribbon. So that selection will be moved or cut from the place where we have selected and saved to the internal memory to paste somewhere else.

Or simply you can use “**CTRL+X**” option to copy the selected object.

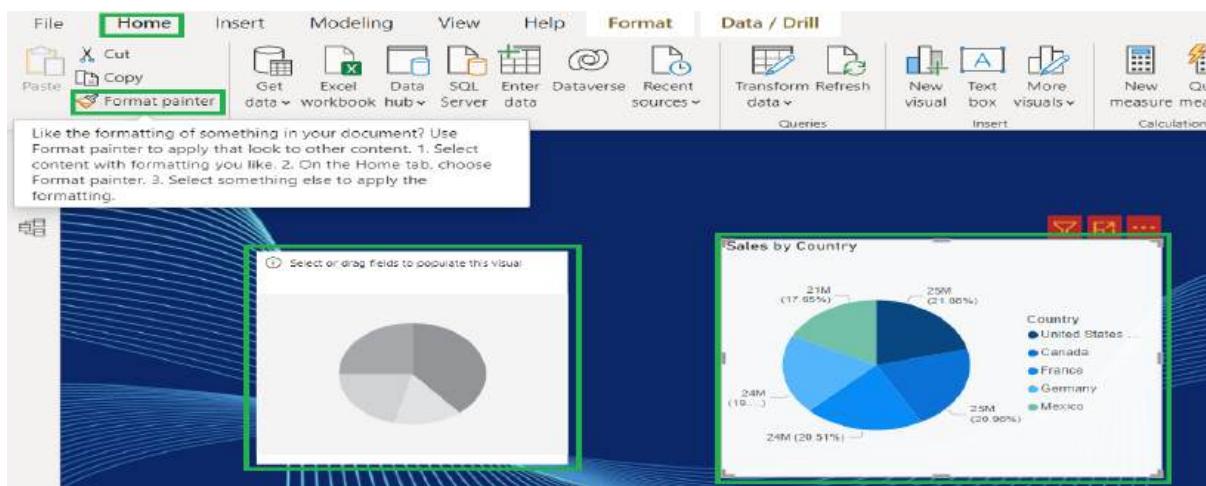


Paste: Already copied or cut object can be pasted anywhere. That means you can paste the selected visual that was already cut or copied.

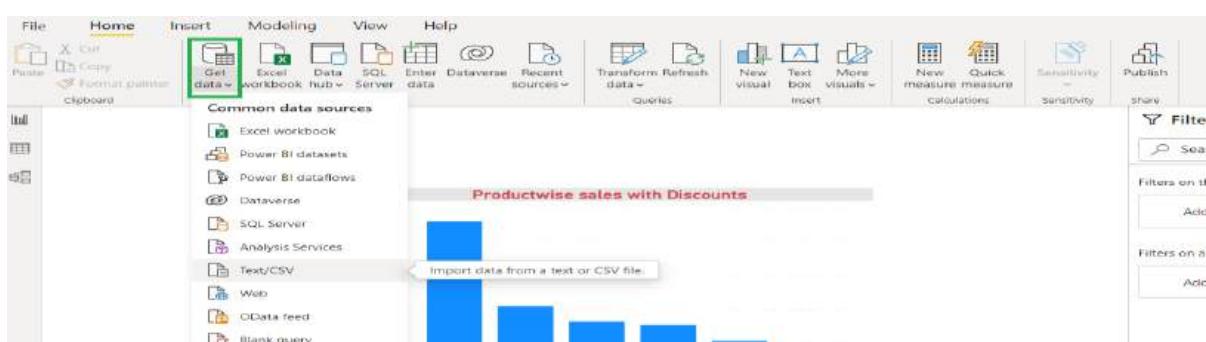


Format Painter: By using format painter we can change the design part of the visual. We can take design from the visual and apply design of it to the same type of another visual. But we can't apply this rule for the other type of visual.

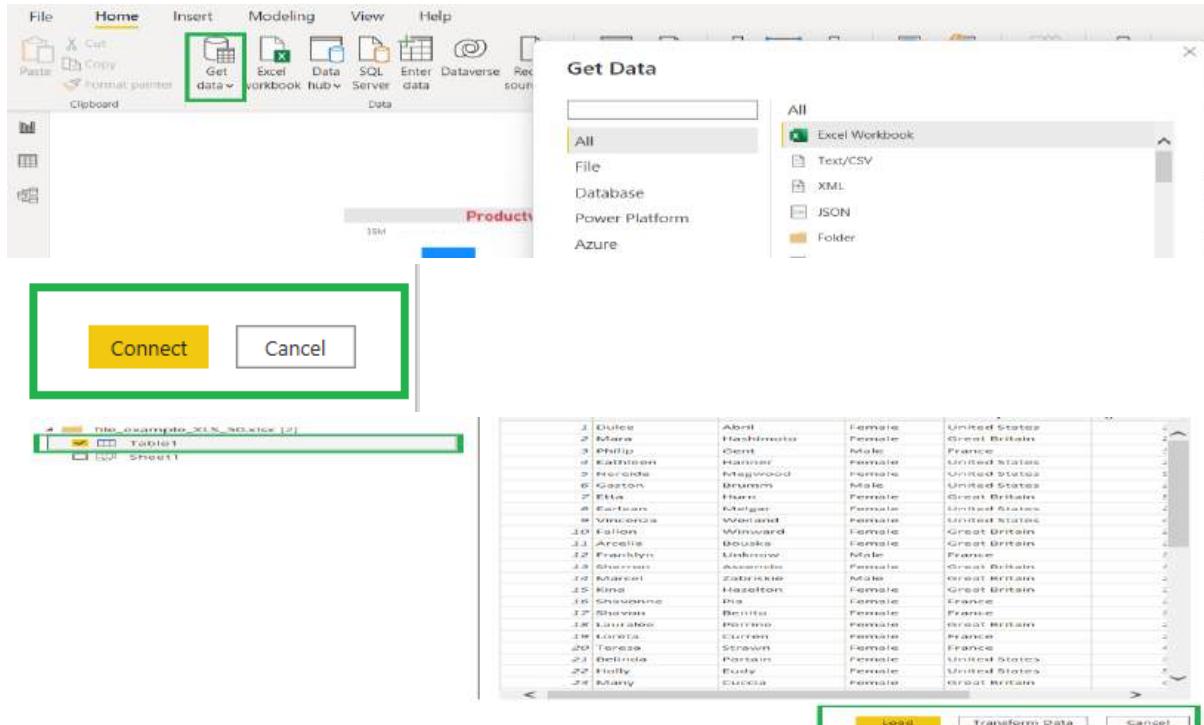
Ex)Table to Table or Pie chart to Pie chart etc.



- **Get data:** In this option, we can see all the available data sources. By selecting any one the data source and get the required data set.



If you select one of the data source again it will pop a window. Then select one data source and click on “**Connect**” button. Then you can see location of the file in the data source. Select the “**Table**” file and click on “**Load**” option. Automatically data will be loaded in the Power BI model.



- **Excel Workbook:** By using this option we can bring the Excel data set to the power BI model.
- **SQL Server:** By using this option we can connect to the SQL Server and fetch the required data set into the power BI model.

When you want to connect with SQL Server, it will ask the credentials

SQL Server database



Here you have to give server name and data base name to connect with server.

Data connectivity mode:

- **Import(Default):** By using this option we can load the entire data set into the power BI memory. From that memory itself, we can fetch the data and utilize for generation of the report. Performance wise it is the best option. By this option we can load “N” number of data sets at a time into power BI memory.

- **Direct Query:** If we don't want the entire dataset into power BI then we can use this option by writing queries. But in this option, for each and every query or operation it will hit the database and fetch the data into power BI memory. It is time consuming process and performance also degrades. It will impact the SQL server.

Once connected, another window will open and there you will see list of objects in the particular database and options like load, transform data and Cancel.

If you click on Load after selecting a dataset, then automatically that data set will be saved into the power BI model. You can change any column data type, size of the column or row lever operations by directly clicking on transform option. Finally you can cancel that window if operation over.

Enter Data:

- ✓ By using this option we can create a data set. We can create columns and rows.
- ✓ If we want to delete or insert columns or rows that we can do here by clicking on the right click option.

Create Table

A	1
2	2
3	3
4	4
5	5

- ✓ If we want to copy data from any table to this table that we can do here from transform data option. Then you have to give name of the table also. Once data copied you then click on either “Load” or “Edit” option.



- ✓ Here we have to double click on the source option under Applied steps tab. Then paste the data by placing the cursor point on the first column.

- ✓ Whatever the data set created by the “Enter Data” option that data set can be modified. But we can’t modify the data set that is getting from different data sources.
- ✓ If you want to add column or row you can click on “+” button or right click on any column. If you want to rename the column you can double click on the column and change the name.

Create Table

	A	X	+
1	1	A	
2	2	B	
3	3	C	
4	4	D	
5	5	E	
+			

Name:

Load **Edit** **Cancel**

- **Recent sources:** Whatever the data sources we have recently used those can be available here. From there itself we can directly go to the particular data source without giving credentials again.
- **Transform Data:** Data level modifications we can do in this option. When you click on this option new window will be opened. That window is called transform data window or Power Query Editor. After getting data into power BI, if we want to do any modifications on that data in that case we can click on this option.

Power Query Editor: After clicking on the “Transform data” option in “Home” ribbon, then you can go to the “Transform Data” window or “Power Query Editor”. In this section we have lot of options again. See below screenshot.

Power Query Editor Options

❖ Home Tab:

- **Close & Apply:** Close the power query editor after applying pending changes. That means if you take any option and do modifications on any data source that will not be saved directly into the power memory. So we need to click on “**Close & Apply**” option. It will save the changes and close the power query editor.
- **Apply:** Apply any pending changes. If you want to save the changes and don’t want to close the power query editor, then use this “**Apply**” option.

- **Close:** Close the power query editor window without applying pending changes. If you want to close without saving the changes then we have to click on this “**Close**” option. Once choose this option if you go to the home page, it will raise a popup denotes that weather we need to do “**Apply Changes**” or “**Discard Changes**”

You have to click on “**Apply Changes**” if you want to save pending changes in the memory.

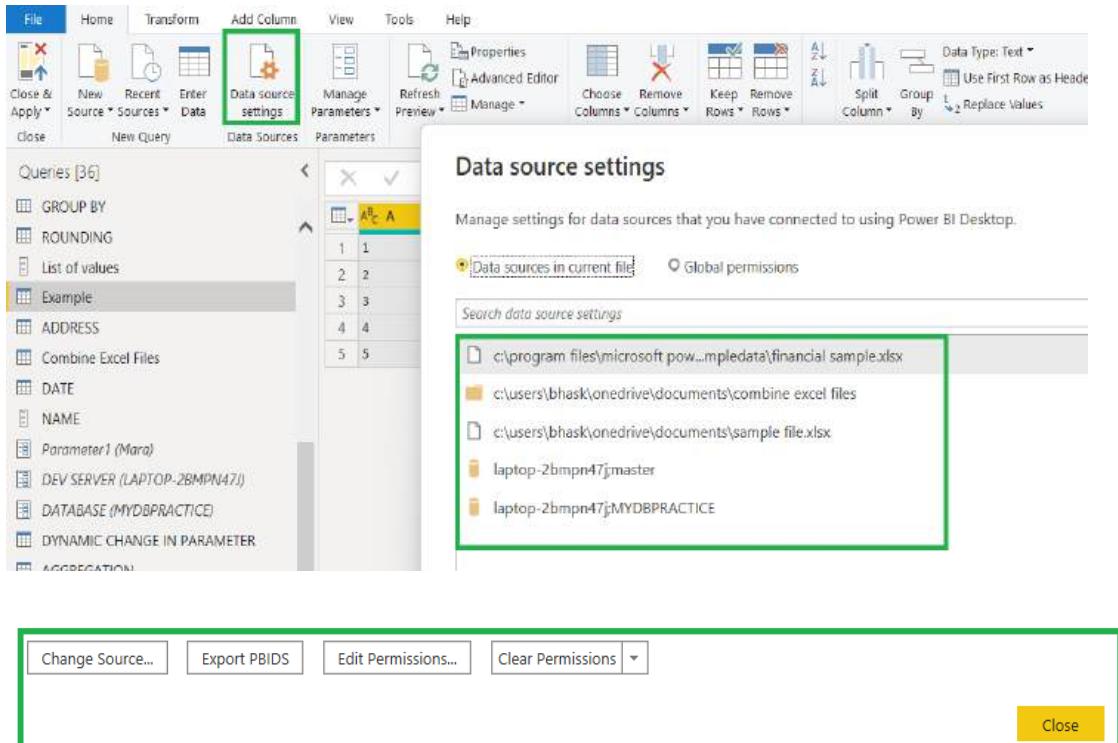
- **New Source:** Whatever the option we have in “**Model**” tab. i.e “**Get Data**”. Same kind of operation will be done in “**New Source**” as well. Only difference is when you connect with “**Get data**” we have three options. **Load, Transform Data and Cancel**. But in “**New Source**” you will see only two options “**OK**” and “**Cancel**”

ID	First Name	Last Name	Gender	Nationality
1	DUCIE	April	Female	United States
2	Maria	Hashimoto	Female	Great Britain
3	Philip	Izant	Male	France
4	Kathleen	Hanner	Female	United States
5	Nereida	Magwood	Female	United States
6	Gaston	Brumm	Male	United States
7	Ella	Hurn	Female	Great Britain
8	Earleen	Melgar	Female	United States
9	Vincenza	Welland	Female	United States
10	Fallon	Wimeward	Female	Great Britain
11	Arcelia	Bouska	Female	Great Britain
12	Franklyn	Unknow	Male	France
13	Sherron	Ascencio	Female	Great Britain
14	Marcel	Zabriskie	Male	Great Britain
15	Kina	Hazelton	Female	Great Britain
16	Shavonne	Pia	Female	France
17	Shavon	Bonito	Female	France
18	Lauralee	Perrine	Female	Great Britain
19	Loretta	Curren	Female	France
20	Teresa	Strawn	Female	France
21	Belinda	Parkan	Female	United States
22	Holly	Eudy	Female	United States
23	Many	Cuccia	Female	Great Britain

- **Recent Sources:** Whatever data sources you have recently connected those data sources will be shown here. Whatever the option “**Recent Sources**” we have in the model, same kind of thing will be done here in the transform as well. Credentials for the data base related things are already set. No need to connect again from this option. You can directly connect by just connecting.

Country	Product
Canada	Carretera
Germany	Carretera
France	Carretera
Germany	Carretera
Mexico	Carretera
Germany	Carretera

- **Enter Data:** If you want to create a table or Data set you can use an option called “**Enter Data**”. Whatever the table we have created by using this option can be modified. You can create a table directly from model tab or transform data option. Both are same.
- **Data source Settings:** By using this option, we can manage the settings of the data sources we have connected to use power bi desktop. You will see all the paths of the all data sources here.



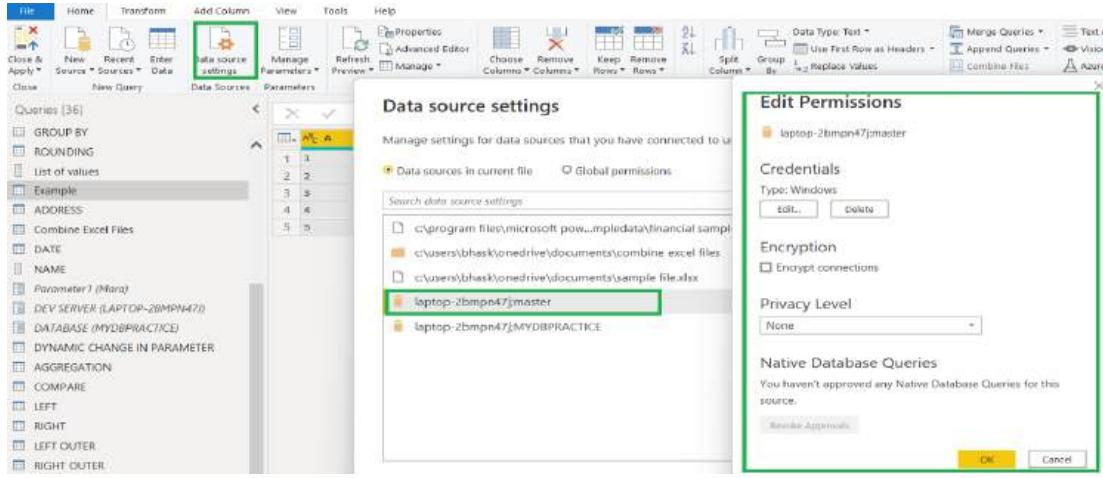
- Suppose if we have 4 files from the one location. Then only one data set path will be created for the all 4 files.
- Here we have two options
 - Data sources in current file
 - Global permissions
- Whatever the current files are there in the power bi desktop, you will see their corresponding paths.
- In Global permissions, whatever the paths you have by using credentials those related file paths will be visible here. Ex) Sql paths, onedrive paths etc.
- Bottom of this page you can see 4 options.
 - Change Source
 - Export PBIDS
 - Edit Permissions
 - Clear Permissions

Change Source: If you want to change the path of the file you can choose the option called “**Change Source**”. If you want to connect with different data base server, you can use this option.

Export PBIDS: Whatever the data set we have selected, those related data set will be downloaded as a Power BI file to the local system. If you open the downloaded file, automatically it connects to that data set. Here **PBIDS** means “**Power BI Data Set**”

Edit Permissions: If you want to edit permission of the any data source, you can use this option. Ex) Select SQL Server path and click on “**Edit**

Permissions" option. It will show you edit permission window. There you can edit the credentials.



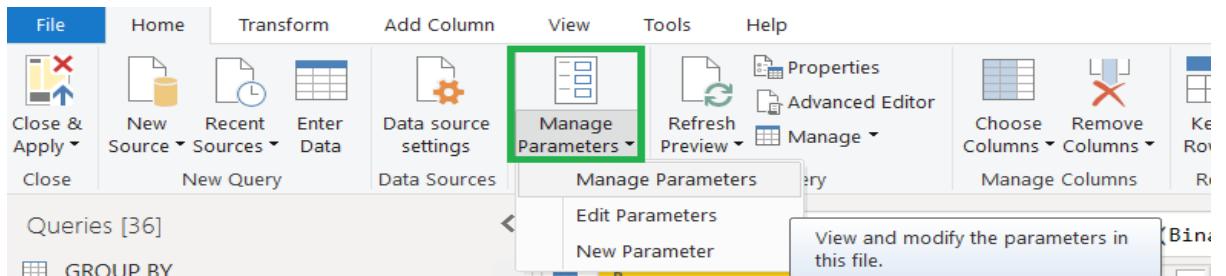
Clear Permissions: In this, we have two options i) **Clear permission** ii) **Clear all permissions**

Clear permission: Whatever data set you have selected, those data set related path permissions will be cleared.

Clear all permissions: Whatever the data sets available in the data source settings window, those all related path permissions will be cleared.

- **Manage parameters:** Parameter is nothing but a piece of information that you can supply dynamically on run time. So it is useful for run time execution. By using manage parameters we can create parameter, Edit parameter and we can manage them. In this, we have 3 options.

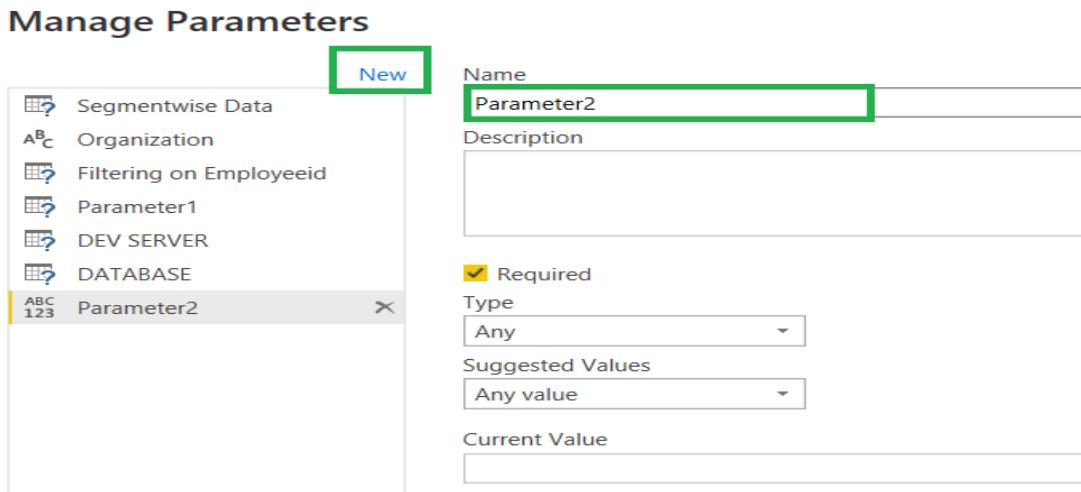
- Manage Parameter
- Edit Parameter
- New Parameter



Manage Parameter: Whatever the parameters we have already, those can be managed here.

Edit Parameter: If you want to edit a parameter, you have to use this option.

New Parameter: If you want to create a new parameter, you can use this option. Here you can pass the parameter Name, Required values like, Type of parameter, Suggested values and a current value.



Note: Parameter will be used in two ways.

- Filtering the column values
- Changing the Environments.

Filtering the column values:

In this section, we have to discuss about how to filter the column values by using parameter.

For example take “**Financials**” data set and create a parameter.

Go to “**Manage Parameters**”->Click on “**New Parameter**”

Name: Name of the parameter you can give. Ex) **Filtering Column Values**

Description: You can give the detailing of the parameter to understand better

Type: You can give the type of the value that parameter relates to. Ex) **Any**

Suggested Values: Any Value

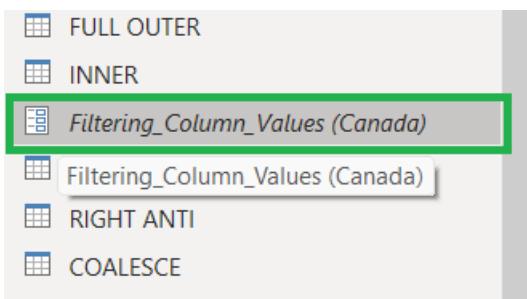
Current Value: By which value we need to filter that value you have to give here. Ex) “**Canada**”

After supply all the above details click on “**OK**”. Then parameter will be created.

Manage Parameters

The screenshot shows the 'Manage Parameters' dialog box. A parameter named 'Filtering_Column_Values' is selected. The dialog includes fields for Name, Description, Required (checked), Type (Any), Suggested Values (Any value), and Current Value (Canada). A preview pane on the left shows the parameter icon next to its name.

Once parameter created you can see the icon of that parameter as shown in below screen.



That means we have to filter the “**Financials**” data set by using this parameter based on the current value “**Canada**”.

Navigation: Go to “**Financials**” data set-> Select the “**Country**” column ->Click on “**drop down**”->**Text filters**-> You can select “**Operator**”->**Equals**->“**Canada**” Click on “**Ok**”. You can see the “**Canada**” related values in the “**Financials**” data set.

Note: You can give the value as “**Canada**” directly or you can pass the parameter here to filter “**Canada**” values.

The screenshot shows the Power BI interface with the 'financials' data set selected. The 'Country' column is currently being edited. A 'Text Filters' dialog box is open, showing the 'Equals...' operator selected and 'Canada' chosen from the list of values. Other operators like 'Does Not Equal...', 'Begins With...', and 'Contains...' are also visible.

Filter Rows

Apply one or more filter conditions to the rows in this table.

Basic Advanced

Keep rows where 'Country'

The screenshot shows the 'Filter Rows' dialog at the top and the 'Queries [37]' pane and 'Table.SelectRows' query editor at the bottom. In the dialog, the 'equals' dropdown is selected, and the 'Filtering_Column_Values' dropdown is open, showing 'Parameter' highlighted. Below it are 'And' and 'Or' radio buttons, and a dropdown menu. At the bottom right are 'OK' and 'Cancel' buttons. The 'Queries [37]' pane lists various queries like 'financials', 'EMP', etc. The 'Table.SelectRows' query editor displays a table with four columns: 'segment' (values: Government, Channel Partners, Enterprise, Enterprise, Government, Channel Partners, Government, Midmarket, Government, Enterprise, small business, Government, Government, Channel Partners, Government, Enterprise), 'Country' (values: Canada, Canada), 'Product' (values: Carretera, Montana, Montana, Montana, Paseo, Paseo, Paseo, Paseo, Paseo, VTT, VTT, VTT, Carretera, Montana, Velo), and 'Discount Band' (values: None, Low, Low, Low). The 'Country' column is highlighted with a green border.

Note: If you change parameter value in the manage parameters option, then data will be filtered automatically based on that value.

Note: If anything you want to filter that should be done in the “**Transform Data**” itself. But don’t apply in “**Model**”. If you apply it on “**Model**” it won’t be affected in the Visualization.

Environment Changing: If you want to change the source of the data set. You can do it in the parameter itself.

Suppose for example, we have fetched some tables from “**SQL Server**”. All these tables related to “**Dev Server**”. In real time, once data is ok in “**Dev Server**”, then we will move that tables to “**Prod Server**”.

If you want to change the path for the table from “**Dev Server**” to “**Prod Server**” each and every table you need to go to “**Advance Editor**” and change path. It is difficult to change for each and every table. So to overcome this problem we can use parameter to change the path dynamically.

The screenshot shows the Power Query Advanced Editor interface. On the left, there's a navigation pane with various query items like 'financials', 'EMP', 'EMPINFO', etc. The 'EMP' item is selected and highlighted with a green border. The main area displays the M code:

```

let
    Source = Sql.Database("LAPTOP-2BMPN47J", "master"),
    dbo_emp = Source{[Schema="dbo",Item="emp"]}[Data],
    #"Changed Type" = Table.TransformColumnTypes(dbo_emp,{{"sal", Int64.Type}}),
    #"Added Custom" = Table.AddColumn(#"Changed Type", "Custom", each Excel.Workbook[sal]),
    #"Added Custom1" = Table.AddColumn(#"Added Custom", "Custom.1", each Date.MonthName[hiredate]),
    #"Removed Columns" = Table.RemoveColumns(#"Added Custom1",{"Custom", "Custom.1"})
in
    #"Removed Columns"

```

If you want to create parameter for the data base table, you can give the server name and table details in the parameter creation itself. Once parameter is created for that table, whatever the tables present in that server, all the tables will automatically point to the same server.

Now we will create parameter change the source of the database tables.

Go to “Manage Parameters”->Click on “New Parameter”

Name: Name of the parameter like “**Dev_Server**”

Description: You can give the detailing of the parameter to understand better

Type: You can give the type of the value that parameter relates to. Ex) **Any**

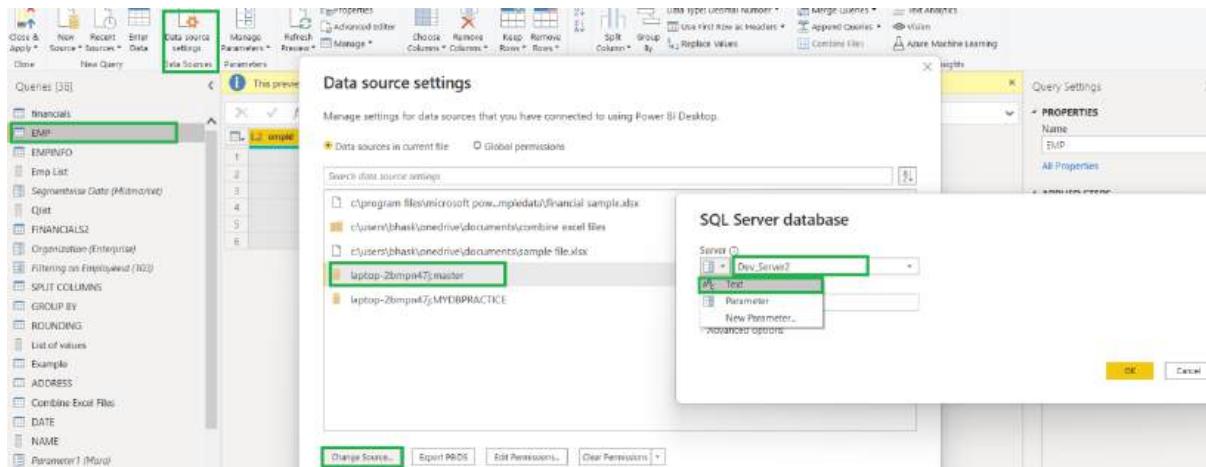
Suggested Values: Any Value

Current Value: You can give the server name here. Like “**LAPTOP-2BMPN47J**”

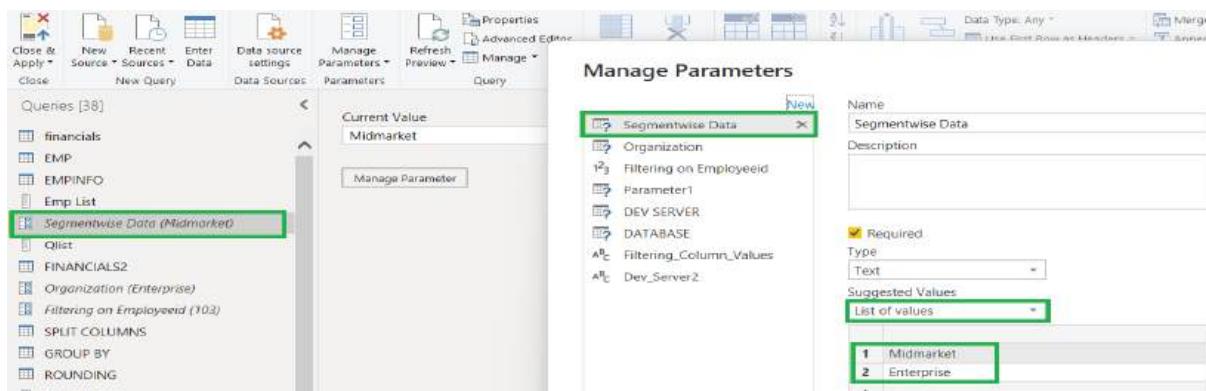
The screenshot shows the ‘Manage Parameters’ dialog. On the left, there's a tree view with items like ‘Segmentwise Data’, ‘Organization’, ‘Filtering on Employeeid’, ‘Parameter1’, ‘DEV SERVER’, ‘DATABASE’, and ‘Filtering_Column_Values’. The ‘Filtering_Column_Values’ item is selected and highlighted with a green border. On the right, there's a form to create a new parameter:

- Name:** Dev_Server2
- Description:** (empty)
- Required:**
- Type:** Any
- Suggested Values:** Any value
- Current Value:** LAPTOP-2BMPN47J

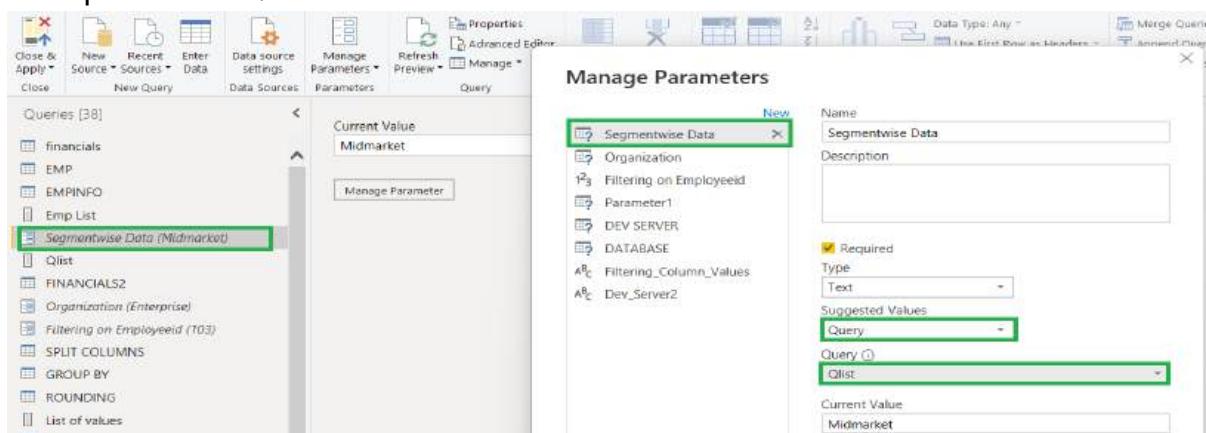
Navigation: Select “**Emp**” table -> “Select “**Data Source Settings**” -> Select “**Database related path**” and click on “**Change Source**”-> Click on “**Parameter**” option -> Click on “**Ok**” -> Click on “**Cancel**”.



Note: If you want to filter a data set based on selected values, you can use the “List of values” option in “Suggested Values” of parameter.



Note: If you want to filter a data set based on a query result, you can use the “Query” option in “Suggested Values” of parameter. Most of the times we can use this option for “SQL Server”



Now select “Emp” table and click on “Advance Editor”. Now you can see the “Parameter”

The screenshot shows the Power BI Advanced Editor interface. On the left, there's a navigation pane with 'Queries [38]' and a tree view of tables like 'financials', 'EMP', 'EMPINFO', etc. The 'EMP' table is selected. The main area displays the M code for the 'EMP' table:

```

let
    Source = Sql.Database("Dev Server2", "master"),
    dbo_emp = Source([Schema="dbo",Item="emp"])[Data],
    #"Changed Type" = Table.TransformColumnTypes(dbo_emp,{{"sal", Int64.Type}}),
    #"Added Custom" = Table.AddColumn(#"Changed Type", "Custom", each Excel.Workbook[sal]),
    #"Added Custom1" = Table.AddColumn(#"Added Custom", "Custom.1", each Date.MonthName[hiredate]),
    #"Removed Columns" = Table.RemoveColumns(#"Added Custom1",{"Custom", "Custom.1"})
in
    #"Removed Columns"

```

Whatever the tables are there, those all the table paths can be automatically pointed to the same server.

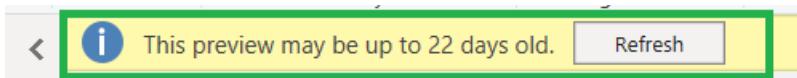
- **Refresh Preview**: Refresh is nothing but reloading the data again from the source. Here we can have 2 options
 - Refresh Preview
 - Refresh all

Refresh Preview: Whatever the data set you have selected, that data set getting refreshed. Latest data will be present once click on “**Refresh Preview**”

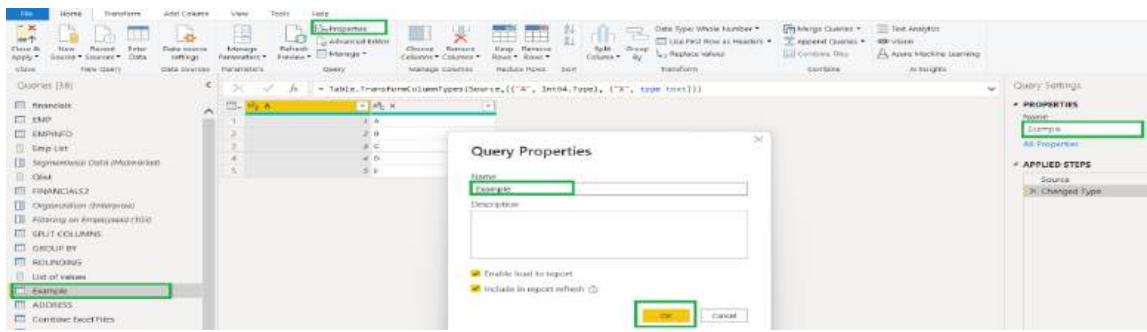
Refresh all: All the data sets will be refreshed.

The screenshot shows the Power BI ribbon. The 'Tools' tab is selected. In the 'Manage' section, the 'Refresh Preview' button is highlighted with a green box. A tooltip for 'Refresh Preview' is visible, stating: 'Refresh the preview results of all queries.'

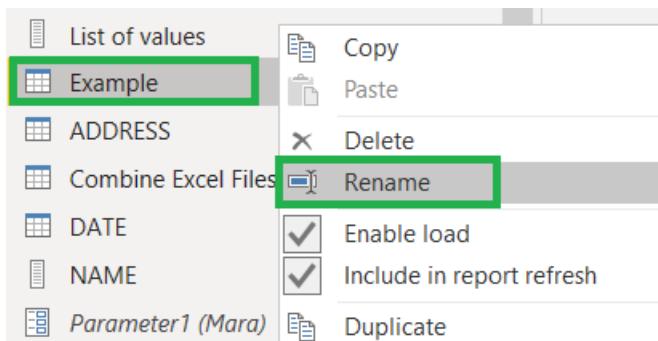
If you don't refresh the data for a long time you can see below notification.



- **Properties**: If you want to change the name of the data set you can use this option. We have different methods to rename the data set.
 - Select “**Data set**” -> Go to “**Properties**” tab -> Write the desired “**Name**” -> Click on “**OK**”



- Select “Data set” -> Go to “Properties” tab on the right side -> Write the desired “Name”
- Select “Data set” -> Right click on that -> Choose the “Rename” option.
- Select “Data set” and double click on that. It will convert into editable mode. Automatically you can change the name there itself.



Applied Steps: Whichever options you have applied in the transform data, those related steps will be added here

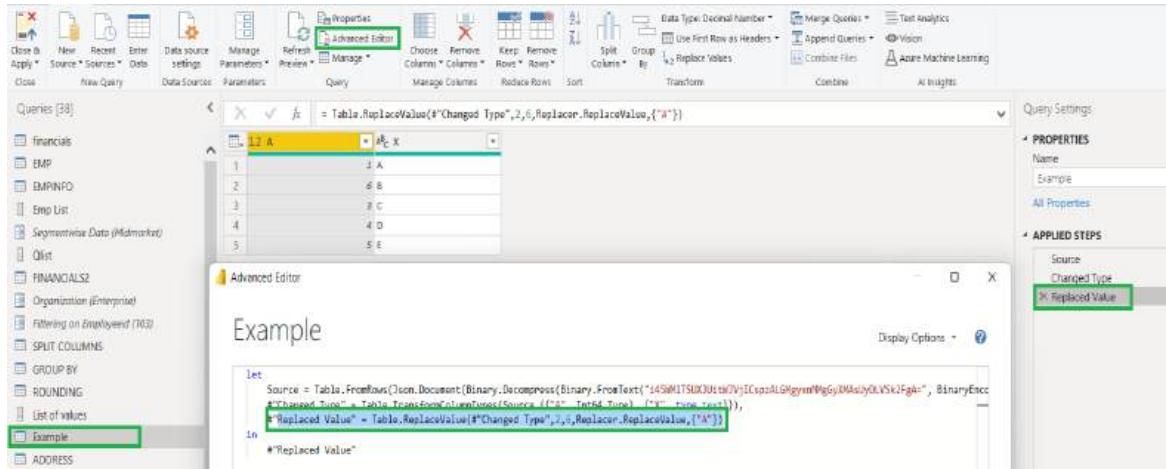
L1	L2	ename	job	sal	hiredate
1	100	SACHIN	CLERK	4000	03-11-2021
2	100	KUMAR	ANALYST	9000	06-08-2022
3	102	RAMIL	MANAGER	6000	10-05-2020
4	103	DAVID	CLERK	5000	05-10-2020
5	104	PHANI		null	06-08-2022
6	105	VENKAT		null	15-06-2023

Suppose select the “Data set” and select “value” click on “Replace Values”. Then pass the “new value”. You can see the step in “Applied Step” options.

Notes:

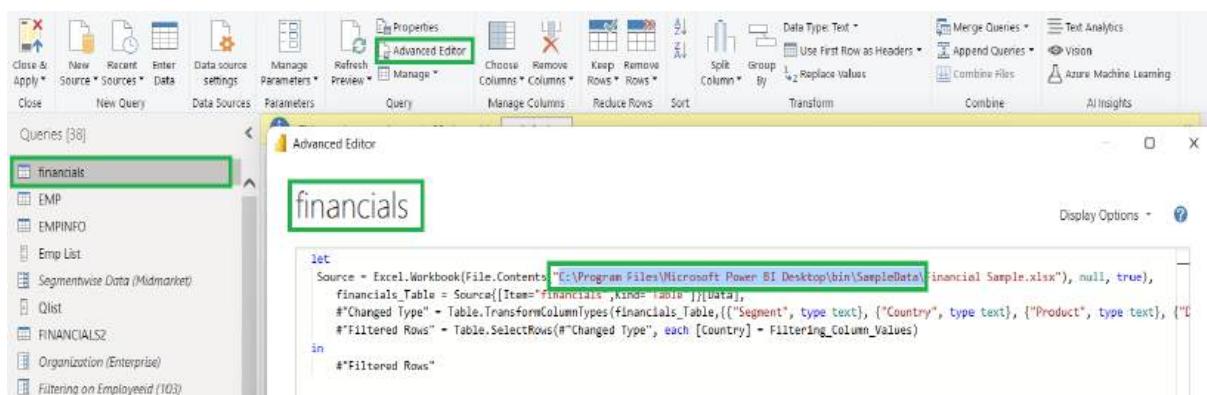
- ✓ If you want to delete the particular step, you can click on the cross button of particular applied step.
- ✓ If you want to go back to the previous step, just select the previous step. Now you can see the formula of that particular step.
- ✓ If you want to edit the step, you can edit by double click on that step. Whatever the steps having settings icon, those steps only will be edited.

- ✓ Whatever the steps we have in applied steps option, same steps are visible in “Advanced Editor” also.
- ✓ Advanced Editor Formulas are case sensitive and these are the coding paths of the particular step. We can call it as “M-Language” coding.

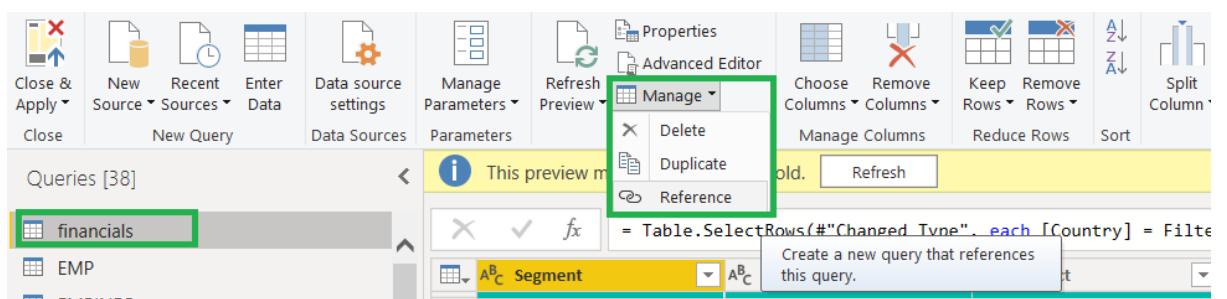


- **Advanced Editor:** If you want to modify any path or any value in the applied step, you have to use “Advanced Editor” option.

If you select the Excel related file and open “Advanced Editor”, you can see the path of that file. In future, if the file is moved to another location, then that path should be mentioned in this “Advanced Editor” to point that file.

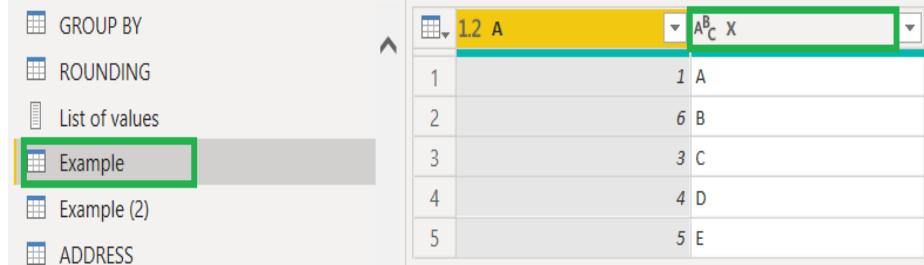


- **Manage:** If you want to manage data sets you can use this option. Here we have three options.



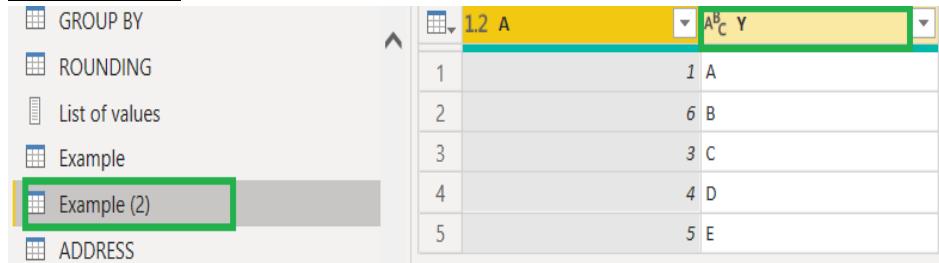
- **Delete**: If you want to delete any particular data set, you can click on this. Or simply right click on the data set and click on “Delete” option.
- **Duplicate**: If you want to create a copy of the data set you can click on this. Or simply right click on the data set and click on “Duplicate” option.

Parent Table:



The screenshot shows the Power Query Editor interface. On the left, there's a list of queries: GROUP BY, ROUNDING, List of values, Example (highlighted with a green border), Example (2), and ADDRESS. The main area displays a table with three columns: A, B, and C. The first row is highlighted in yellow and contains the header "1.2 A". The second row contains "1 A". The third row contains "2 B". The fourth row contains "3 C". The fifth row contains "4 D". The sixth row contains "5 E".

Child Table:

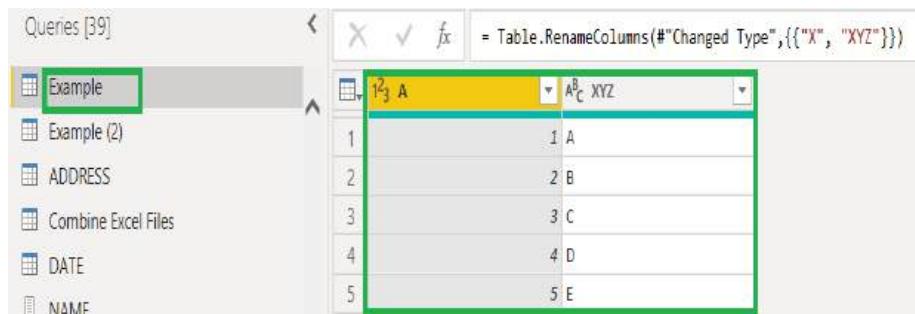


The screenshot shows the Power Query Editor interface. On the left, there's a list of queries: GROUP BY, ROUNDING, List of values, Example, Example (2) (highlighted with a green border), and ADDRESS. The main area displays a table with three columns: A, B, and C. The first row is highlighted in yellow and contains the header "1.2 A". The second row contains "1 A". The third row contains "2 B". The fourth row contains "3 C". The fifth row contains "4 D". The sixth row contains "5 E".

If you change column name in child table also it's not reflected in parent table.

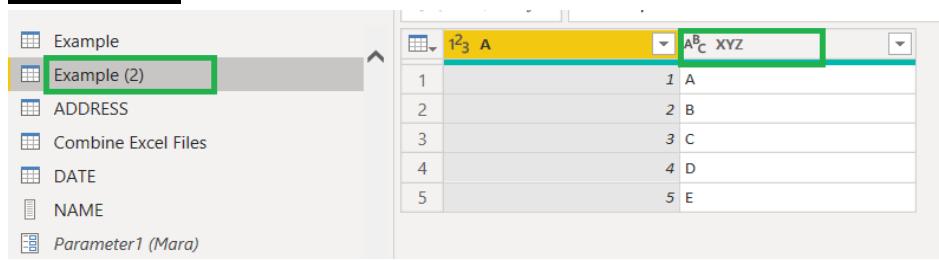
- **Reference**: If you want to create a copy of the data set you can click on this option.

Parent Table:



The screenshot shows the Power Query Editor interface. On the left, there's a list of queries: Queries [39], Example (highlighted with a green border), Example (2), ADDRESS, Combine Excel Files, DATE, and NAME. The main area displays a table with three columns: A, B, and C. The first row is highlighted in yellow and contains the header "1.2 A". The second row contains "1 A". The third row contains "2 B". The fourth row contains "3 C". The fifth row contains "4 D". The sixth row contains "5 E".

Child Table:



The screenshot shows the Power Query Editor interface. On the left, there's a list of queries: Example, Example (2) (highlighted with a green border), ADDRESS, Combine Excel Files, DATE, NAME, and Parameter1 (Mara). The main area displays a table with three columns: A, B, and C. The first row is highlighted in yellow and contains the header "1.2 A". The second row contains "1 A". The third row contains "2 B". The fourth row contains "3 C". The fifth row contains "4 D". The sixth row contains "5 E".

You can see the above three options by right clicking on any data set.

Country	Product	Discount Band
Canada	Carretera	None
Canada	Montana	None
Canada	Montana	None
Canada	Montana	None
Canada	Paseo	None
Canada	Velo	None
Canada	VTT	None
Canada	VTT	Low
Canada	Carretera	Low

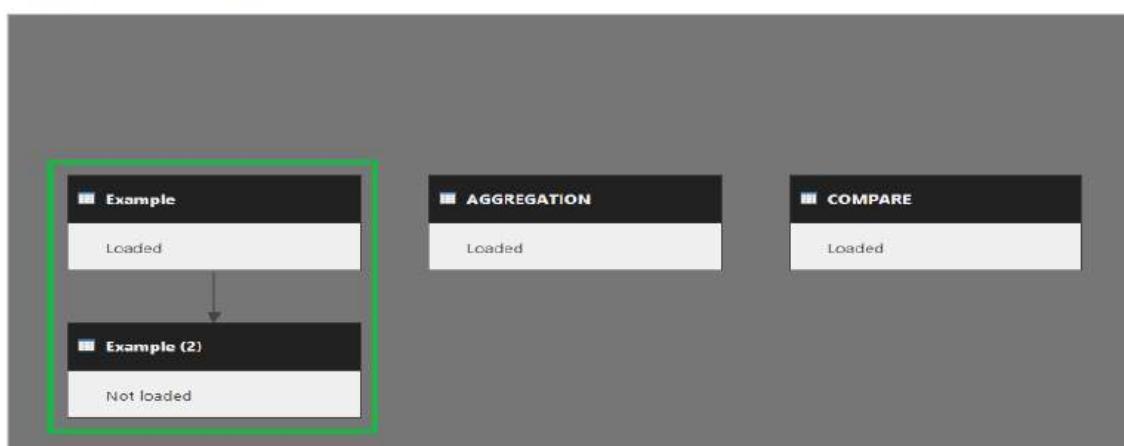
❖ **How you can identify whether the data set created by duplicate or reference option?**

Go to “View” tab -> Click on “Query Dependencies”. Then you can see the relationship between parent table and child table.

A	XYZ
1	A
2	B
3	C
4	D
5	E

See the below screen shot to identify the reference data set.

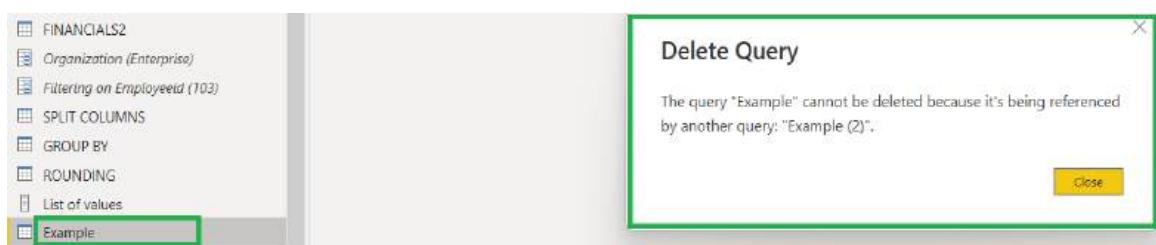
Query Dependencies



❖ **Difference between Duplicate and Reference:**

- ✓ **Duplicate:** Copy of the data set will be created
 - i) If you change anything in parent table it won't reflect in the child table
 - ii) If you change anything in child table it won't reflect in the parent table
 - iii) If you try to delete parent table, it won't throw an error message

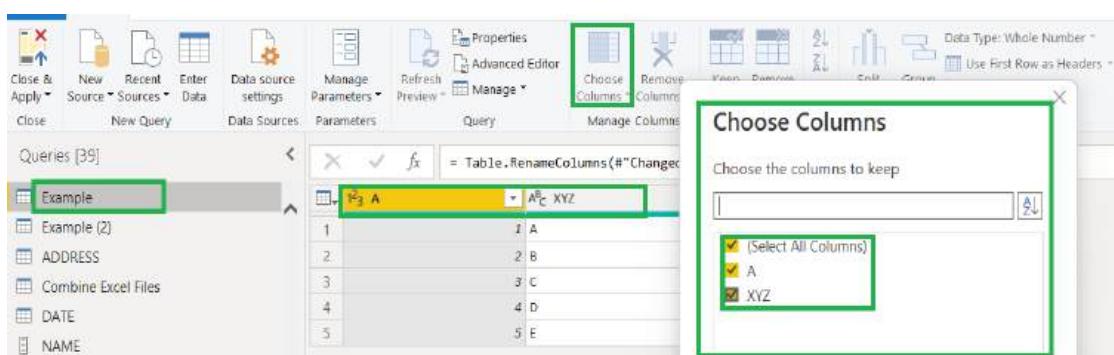
- ✓ **Reference:** Copy of the data set will be created
 - i) If you change anything in parent table it will reflect in the child table as well
 - ii) If you change anything in child table it won't reflect in the parent table
 - iii) If you try to delete parent table, it will throw an error message.



Note: If you want to delete parent table, first you need to delete child table, then delete parent table. It will not through an error in this case.

➤ **Choose Columns:** You can select the required columns. In this, we have two options.

- **Choose columns:** If you want to select required columns then you can do here. Select data set click on “choose columns” option; you will see the list of columns. There you can select and un-select the columns.



- **Go to Columns:** If you want to go to the column directly for selecting, then you can use this option. Or else you can directly select the column by manually.

The screenshot shows the Power Query Editor interface. The 'Choose Columns' button in the ribbon is highlighted with a green box. A tooltip window appears, stating 'Select the column that you would like to go to in the Query Editor preview.' The preview pane shows a table with columns A, B, and C, and rows 1 through 5. The first row is selected.

The screenshot shows the Power Query Editor interface with the 'Go to Column' dialog box open. The dialog box lists the columns 'A', 'XYZ', and 'XYZ'. The column 'A' is highlighted with a green box. The preview pane shows the same table structure as the previous screenshot.

- **Remove Columns:** By using this option, you can remove the columns or the other columns. In this section, we have two options.

- **Remove columns:** selected columns will be removed except unselected columns.
- **Remove Other Columns:** Unselected columns will be removed except the currently selected columns.

Note: If you want to select multiple columns to remove, just press “**CTRL+Click**”

- **Keep rows:** By using this option, we can choose the rows that we want to keep. In this section, we have below options.

The screenshot shows the Power Query Editor interface with the 'Manage Columns' ribbon tab selected. A green box highlights the 'Keep Rows' dropdown menu. A tooltip window states 'Keep only rows containing errors in the currently selected columns.' The preview pane shows the same table structure.

- **Keep Top Rows**

- Keep Bottom Rows
- Keep Range of Rows
- Keep Duplicates
- Keep Errors

Keep Top Rows: We can select the top N rows to keep here. If you click on this option it will ask you “**No of rows to keep**”. Ex) 3. Then it will keep top 3 rows.

The screenshot shows the Power BI desktop interface with a query editor window open. The table in the editor has columns labeled 'A' and 'XYZ', with data rows 1, 2, and 3. A context menu is open over the table, and the 'Keep Top Rows' option is selected, opening a dialog box. The dialog box has a title 'Keep Top Rows' and a sub-instruction 'Specify how many rows to keep.' Below this is a dropdown menu with the value '1,2 - 3' highlighted. At the bottom right of the dialog are 'OK' and 'Cancel' buttons.

Keep Bottom Rows: We can select the bottom N rows to keep here. If you click on this option it will ask you “**No of rows to keep**”. Ex) 3. Then it will keep top 3 rows.

The screenshot shows the Power BI desktop interface with a query editor window open. The table in the editor has columns labeled 'A' and 'XYZ', with data rows 3, 4, and 5. A context menu is open over the table, and the 'Keep Bottom Rows' option is selected, opening a dialog box. The dialog box has a title 'Keep Bottom Rows' and a sub-instruction 'Specify how many rows to keep.' Below this is a dropdown menu with the value '1,2 - 3' highlighted. At the bottom right of the dialog are 'OK' and 'Cancel' buttons.

Keep Range of Rows: Specify the number of rows to keep starting at a specific row. In this, you have to specify first row and number of rows to keep.

The screenshot shows the Power BI desktop interface with a query editor window open. The table in the editor has columns labeled 'A' and 'XYZ', with data rows 2, 3, and 4. A context menu is open over the table, and the 'Keep Range of Rows' option is selected, opening a dialog box. The dialog box has a title 'Keep Range of Rows' and a sub-instruction 'Specify the range of rows to keep.' It contains two input fields: 'First row:' with the value '1,2' and 'Number of rows:' with the value '1,2 - 4'. At the bottom right of the dialog are 'OK' and 'Cancel' buttons.

Keep Duplicates: If you want to keep duplicate rows based on the currently selected column, you can use this option.

The screenshot shows the Power Query Editor interface. A green box highlights the 'Keep Errors' step in the 'Applied Steps' pane. The main area displays a table with columns 'A' and 'X'. Row 5 contains an error ('#DIV/0!'), which is highlighted with a red box. The status bar at the bottom right indicates 'Keep Errors'.

Keep Errors: Keep rows containing only errors in the currently selected columns.

This screenshot is similar to the previous one, showing the 'Keep Errors' step applied. The table now includes row 5 with the error '#DIV/0!' in column A. A tooltip explains: 'Keep only rows containing errors in the currently selected columns.'

➤ **Remove rows:** You can choose the rows that you want to remove. In this section, we have below options.

The screenshot shows the 'Remove Rows' step applied. The table has rows 1 through 5 removed, leaving only row 6. A tooltip says: 'Remove rows containing errors in the currently selected columns.'

- Remove Top Rows
- Remove Bottom Rows
- Remove alternate Rows
- Remove Duplicates
- Remove Blank Rows
- Remove Errors

Remove Top Rows: You can remove top N rows here. You need to specify how many rows that you want to remove.

This screenshot shows the 'Remove Top Rows' step applied. The table has the first two rows removed, leaving rows 3 through 4. A tooltip says: 'Remove the top N rows from this table.'

Remove Bottom Rows: You can remove bottom N rows here. You need to specify how many rows that you want to remove.

The screenshot shows the Power BI Data Editor interface. A dialog box titled "Remove Bottom Rows" is open, prompting the user to "Specify how many rows to remove from the bottom." An input field contains the value "3". The "OK" button is highlighted with a green border.

Remove alternate Rows: You can remove alternative rows by passing below three arguments.

- First row to remove
- No of rows to remove
- No of rows to keep

The screenshot shows the Power BI Data Editor interface. A dialog box titled "Remove Alternate Rows" is open, prompting the user to "Specify the pattern of rows to remove and keep." It has three input fields: "First row to remove" (set to "1"), "Number of rows to remove" (set to "2"), and "Number of rows to keep" (set to "1"). The "OK" button is highlighted with a green border.

Remove Duplicates: Whatever the duplicate rows we have in the data set, those duplicate rows will be deleted by using this option.

The screenshot shows the Power BI Data Editor interface. The ribbon has the "Transform" tab selected. A tooltip for the "Remove Duplicates" button in the ribbon area states: "Remove rows containing duplicated values in the currently selected columns."

Remove Blank Rows: Whatever the blank rows we have in the table, those Blank rows will be deleted by clicking on this option. Here "Null" is nothing but blank values (system generated nulls only).

The screenshot shows the Power BI Data Editor interface. A context menu is open over a column named 'Changed Type'. The 'Remove Errors' option is highlighted with a green box. A tooltip at the bottom right of the menu says: 'Removes all blank rows from this table.' The ribbon above shows various transform options like 'Close & Apply', 'Home', 'Transform', etc. The 'APPLIED STEPS' pane on the right shows the step 'Removed Columns'.

Remove Errors: Whatever the errors we have in the selected column, their corresponding rows will be removed.

This screenshot is similar to the previous one but shows a different column named 'Changed Type' with character values. Row 9 is empty. The 'Remove Errors' option is highlighted with a green box, and a tooltip says: 'Remove rows containing errors in the currently selected columns.' The 'APPLIED STEPS' pane shows the step 'Removed Columns'.

- **Sort:** Arranging the data in proper order either ascending order (lowest to highest) or descending order (highest to lowest) for numeric data and for character data sort will be applied on alphabetical order.

This screenshot shows the 'Sort Ascending' step applied to a column named 'Changed Type'. The column contains numeric values from 1 to 9. The 'Sort Ascending' option is highlighted with a green box, and a tooltip says: 'Sort lowest to highest.' The 'APPLIED STEPS' pane shows the step 'Sorted Rows'.

Note: Same option will be available on the drop down menu of the selected column.

This screenshot shows the 'Sort Ascending' step applied to a column named 'Changed Type'. The column contains character values from A to I. The 'Sort Ascending' option is highlighted with a green box, and a tooltip says: 'Sort lowest to highest.' The 'APPLIED STEPS' pane shows the step 'Sorted Rows'.

- **Split Column:** Whatever the column is there, that column will be split into multiple columns. In this section, we have below options.

The screenshot shows the Power Query Editor interface. In the top ribbon, the 'Home' tab is selected. On the far right, there's a 'Split Column' button with a dropdown arrow. A context menu has popped up, listing several options under the heading 'Split Column by Delimiter'. The 'By Delimiter' option is highlighted with a green box. Other options listed include 'By Positions', 'By Lowercase to Uppercase', 'By Uppercase to Lowercase', 'By Digit to Non-Digit', and 'By Non-Digit to Digit'. Below the menu, a tooltip provides a brief description: 'Split values in the selected column based on the specified delimiter.'

- By Delimiter
- By Number of Characters
- By Positions
- By Lowercase to Uppercase
- By Uppercase to Lowercase
- By Digit to Non-Digit
- By Non-Digit to Digit

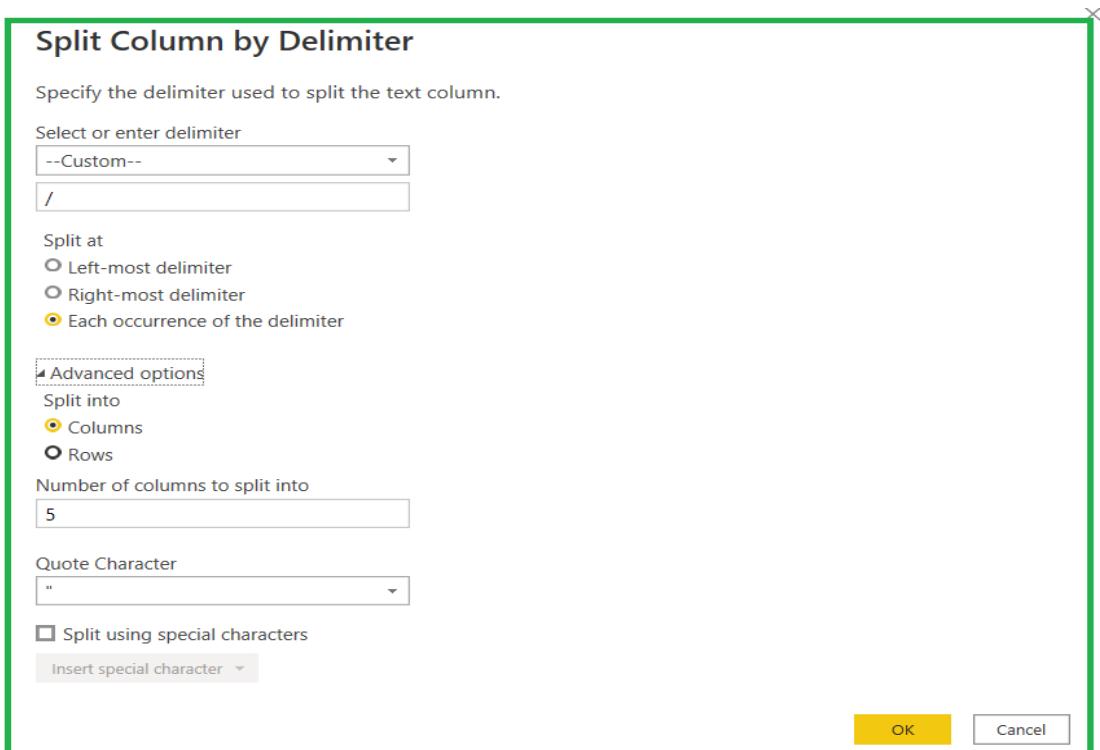
By Delimiter: Delimiter is nothing but a separator. Based on the delimiter we can split the column. It is also called as special character.

Take an address column for example

Address - Copy
8-81-2/Arundalpet/VIJAYAWADA/522566/AP
6-521-A/Arundal pet/Guntur/522601/AP
321-4a/Maharani veedhi/Marripalem/530021/AP
1-1A/Yellareddi guda/Hyderabad/508254/TN

In the above column we have delimiter "/". Based on that delimiter it will split that column into multiple columns. Select the column and click on the "By delimiter" option in "Split columns". Then it will pop up a window.

This screenshot shows the Power Query Editor with the 'Address' column selected. The 'Split Column' button is highlighted with a green box. A 'Split Column by Delimiter' dialog box is open, showing various options: 'By Positions' (selected), 'By Lowercase to Uppercase', 'By Uppercase to Lowercase', 'By Digit to Non-Digit', and 'By Non-Digit to Digit'. The 'Query Settings' pane on the right shows the step name as 'SPLIT COLUMNS'. The 'APPLIED STEPS' pane at the bottom lists the applied step as 'Split Column by Delimiter'.



Select or Enter Delimiter: Need to select delimiter from the drop down menu or you can give custom option to mention your own delimiter. For which delimiter we need to split, that you can mention here.

Split at: Here we need to mention from where we need to split. That means left most delimiter, Right most delimiter, each occurrence of the delimiter

Left most delimiter: Split from left side onwards wherever the first delimiter is there, up to that delimiter. It has to split one column remaining data should come in second column.

A ^B _C Address.1	A ^B _C Address.2	A ^B _C Address - Copy
8-81-2	Arundalpet/VIJAYAWADA/522566/AP	8-81-2/Arundalpet/VIJAYAWADA/522566/AP
6-521-A	Arundal pet/Guntur/522601/AP	6-521-A/Arundal pet/Guntur/522601/AP
321-4a	Maharani veedhi/Marripalem/530021/AP	321-4a/Maharani veedhi/Marripalem/530021/AP
1-1A	Yellareddi guda/Hyderabad/508254/TN	1-1A/Yellareddi guda/Hyderabad/508254/TN

Right most delimiter: Split From right side onwards where ever the first delimiter is there, up to that delimiter it has to split as one column, remaining data should come in second column.

A ^B _C Address	A ^B _C Address - Copy.1	A ^B _C Address - Copy.2
1 8-81-2/Arundalpet/VIJAYAWADA/522566/AP	8-81-2/Arundalpet/VIJAYAWADA/522566/AP	AP
2 6-521-A/Arundal pet/Guntur/522601/AP	6-521-A/Arundal pet/Guntur/522601	AP
3 321-4a/Maharani veedhi/Marripalem/530021/AP	321-4a/Maharani veedhi/Marripalem/530	AP
4 1-1A/Yellareddi guda/Hyderabad/508254/TN	1-1A/Yellareddi guda/Hyderabad/508254	TN

Each occurrence of the delimiter: For each and every delimiter occurrence a separate column should come and rest of the data to be presented in another column.

A ^B Address	A ^B Address - Copy.1	A ^B Address - Copy.2	A ^B Address - Copy.3	A ^B Address - Copy.4	A ^B Add
1 8-81-2/Arundalpet/VIJAYAWADA/522...	8-81-2	Arundalpet	VIJAYAWADA	522566	AP
2 6-521-A/Arundal pet/Guntur/522601/...	6-521-A	Arundal pet	Guntur	522601	AP
3 321-4a/Maharani veedhi/Marripalem...	321-4a	Maharani veedhi	Marripalem	530021	AP
4 1-1A/Yellareddi guda/Hyderabad/508...	1-1A	Yellareddi guda	Hyderabad	508254	TN

Advanced options: We have to split the column into multiple columns or multiple rows

Ex) Splitting column in to multiple rows:

The screenshot shows the 'Split Column by Delimiter' dialog box. On the left, there's a preview of the data with 20 rows. On the right, under the 'Delimited' tab, the 'Delimited by' dropdown is set to 'Comma'. Under 'Advanced options', the 'Split into' dropdown is set to 'Rows', which is highlighted with a green box. Other options like 'Left-most delimiter', 'Right-most delimiter', and 'Each occurrence of the delimiter' are also present.

No of columns to split into: How many columns we require as a result that number should mention here.

By Number Of Characters: Column will be split into multiple columns or rows based on number of characters. If you click on this option, it will ask you the number of characters. Again we need to provide an option called “split”.

Once, as far left as possible: It will consider from the left side based on no of characters

Once, as far right as possible: It will consider from the right side based on no of characters

Repeatedly: Based on the no of characters, repeatedly it will split the column

The screenshot shows a software interface with four columns of address data. The first column is labeled 'Address' and contains four rows of address details. The subsequent three columns are labeled 'Address - Copy.1', 'Address - Copy.2', and 'Address - Copy.3'. A modal dialog titled 'Split Column by Number of Characters' is overlaid on the interface. It asks for the 'Number of characters' (set to 15) and provides options for splitting: 'Once, as far left as possible' (selected), 'Once, as far right as possible', and 'Repeatedly'. There are also 'Advanced options' and 'OK/Cancel' buttons.

Ex) No of characters: 15

For every 15 characters separate column will be split

Note: Here also we can split column into multiple rows

By Positions: In this section, we need to mention for which positions we want to split the column.

Ex)

The screenshot shows a software interface with a table having four columns. The first column is labeled 'Column1' and the subsequent three are labeled 'Column1 - Copy.1', 'Column1 - Copy.2', and 'Column1 - Copy.3'. The data in the columns is as follows:

Column1	Column1 - Copy.1	Column1 - Copy.2	Column1 - Copy.3
8-81/ben2 circle/vijayawada/520010/AP	8-81/ben	z	circle/vijayawada/520010/AP
6-521-A/Arundal pet/Guntur/522601/AP	6-521-A/	Ar	undal pet/Guntur/522601/AP
321-4a/Maharani veedhi/Marripalem/Visakhapatnam/530020..	321-4a/M	ah	maharani veedhi/Marripalem/Visakhapatnam/530020/AP
1-1A/Yellareddi guda/Hyderabad/Telangana/508254/TN	1-1A/Yel	la	reddi guda/Hyderabad/Telangana/508254/TN

A modal dialog titled 'Split Column by Positions' is open, asking for 'Positions' (0, 8, 10) and featuring 'OK/Cancel' buttons.

First column: $8-0 = 8$ (From left side 8 characters)

Second column: $10-8 = 2$ (2 characters from the left side)

Note: Interval between first two position values will be consider as first column, again interval between second and third position values will be consider as second column Like this column will be split. Rest of the characters comes in final column.

By Lowercase to Uppercase: It will split the column up to the lower case letters are present from the left side. Once the lower case letters are over then it will split up to that part, rest of the characters displayed as separate column. If you want to split column based on this option it should have first character as lower case letter. Otherwise it will not split properly.

Ex)

Queries [42]

The screenshot shows the Power Query Editor interface. On the left, there's a tree view with 'Dataset' selected. The main area displays a table with four columns: 'Dataset', 'Dataset - Copy.1', 'Dataset - Copy.2', and 'Dataset - Copy.3'. The 'Dataset' column contains three rows of data: 'faddFFFFAaaaa111', 'ABCDefghijklmnop', and 'abcdEFGHijklMNOP'. The 'Dataset - Copy.1' column contains 'fadd', 'ABCDefgh', and 'abcd'. The 'Dataset - Copy.2' column contains 'FFFFAaaaa111', 'ijklmnop', and 'EFGHijkl'. The 'Dataset - Copy.3' column contains 'null', 'null', and 'MNOP'. The formula bar at the top shows the formula: `= Table.SplitColumn(#"Duplicated Column", "Dataset - Copy", Splitter.SplitTextByCharacterTransition({"A".."Z"}, {"A".."Z"}))`.

Dataset	Dataset - Copy.1	Dataset - Copy.2	Dataset - Copy.3
faddFFFFAaaaa111	fadd	FFFFAaaaa111	null
ABCDefghijklmnop	ABCDefgh	ijklmnop	null
abcdEFGHijklMNOP	abcd	EFGHijkl	MNOP

By Uppercase to Lowercase: It will split the column up to the upper case letters are present from the left side. Once the upper case letters are over then it will split up to that part, rest of the characters displayed as separate column.

If you want to split column based on this option it should have first character as upper case letter. Otherwise it will not split properly.

Ex)

Queries [42]

The screenshot shows the Power Query Editor interface. On the left, there's a tree view with 'Dataset' selected. The main area displays a table with four columns: 'Dataset', 'Dataset - Copy.1', 'Dataset - Copy.2', and 'Dataset - Copy.3'. The 'Dataset' column contains three rows of data: 'faddFFFFAaaaa111', 'ABCDefghijklmnop', and 'abcdEFGHijklMNOP'. The 'Dataset - Copy.1' column contains 'fadd', 'ABCDefgh', and 'abcd'. The 'Dataset - Copy.2' column contains 'FFFFAaaaa111', 'ijklmnop', and 'EFGHijkl'. The 'Dataset - Copy.3' column contains 'null', 'null', and 'MNOP'. The formula bar at the top shows the formula: `= Table.SplitColumn(#"Duplicated Column", "Dataset - Copy", Splitter.SplitTextByCharacterTransition({"A".."Z"}, {"a".."z"}))`.

Dataset	Dataset - Copy.1	Dataset - Copy.2	Dataset - Copy.3
faddFFFFAaaaa111	fadd	FFFFAaaaa111	null
ABCDefghijklmnop	ABC	efghijkl	mnop
abcdEFGHijklMNOP	abcd	EFGHijkl	MNOP

By Digit to Non Digit: It will split the column up to the digit values are present. Once non digit value arrives then there onwards separate column will be split. Starting character must and should be a digit, otherwise it will not give proper result.

Ex)

Queries [43]

The screenshot shows the Power Query Editor interface. On the left, there's a tree view with 'Digit' selected. The main area displays a table with two columns: 'Digit' and 'Digit - Copy.1'. The 'Digit' column contains two rows of data: '123aaaAAA' and 'AAA01222'. The 'Digit - Copy.1' column contains '123' and 'AAA01222'. The formula bar at the top shows the formula: `= Table.SplitColumn(#"Duplicated Column", "Digit - Copy", Splitter.SplitTextByCharacterTransition({"0".."9"}, {c => not Char.IsDigit(c)}))`.

Digit	Digit - Copy.1
123aaaAAA	123
AAA01222	AAA01222

By Non Digit to Digit: It will split the column up to the non digit values are present. Once digit value arrives then there onwards separate column will be split. Starting character must and should be a non digit, otherwise it will not give proper result.

Ex)

- **Group By:** Summarizing the data is called group by. That means it will group the rows based on the selected columns.

For summarizing data it will use aggregations functions like sum, max, min, average, count, Count Distinct rows etc.

For example, if we want “**Segment**” wise total sales on the “**Financials**” data set then we can use “**Group By**” option here.

Select the “**Financials**” data set, select “**Segment**” column and click on “**Group By**” option.

Basic: Here we can specify the column to group by, Name you can provide for new column, Operation you can specify(sum, max, min, count etc), On which column you need to do grouping that column you have to mention here.

Advanced: In this section, we can give do group by operation on multiple columns. If you click on this you will have the below screen.

Group By

Specify the columns to group by and one or more outputs.

Basic Advanced

Segment

Country

Add grouping

New column name

Count

Operation

Count Rows

Column

Sum

Sum

Sales

Add aggregation

OK

Cancel

For example, “**Country and Segment**” wise Total sales and no of rows we can do group by at a time. If you specify all the details and click on “**Ok**” it will give you the below result.

If you want to add another column to do group you can click on “**Add grouping**” option.

If you want to add another aggregation on the same columns you can click on “**Add aggregation**” option

If you want to add “**N**” no of columns for grouping and if you want to do multiple aggregations on these columns then we can do here.

The screenshot shows the Power BI Data Editor interface. On the left, there's a navigation pane with various queries listed under 'Queries [43]'. One query is selected and highlighted with a green border, labeled 'Financials'. The main area displays a table with four columns: 'Segment' (containing values like Government, Midmarket, Channel Partners, etc.), 'Country' (containing values like Canada, Germany, France, Mexico, United States of America), 'Count' (containing values like 20, 20, 20, 20, 20), and 'Sum' (containing values like 10741290.32, 11452895.94, 593802.073, 301344.75, 511136.4). The entire table is also highlighted with a green border.

Segment	Country	Count	Sum
Government	Canada	20	10741290.32
Government	Germany	20	11452895.94
Midmarket	France	20	593802.073
Midmarket	Germany	20	301344.75
Midmarket	Mexico	20	511136.4
Channel Partners	Canada	20	491184.14
Government	France	20	12127782.73
Channel Partners	Germany	20	336425.88
Enterprise	Canada	20	3967491.25
Small Business	Mexico	20	7096356
Midmarket	United States of America	20	465385.875
Government	Mexico	20	9791599.38
Government	United States of America	20	8390746.11
Channel Partners	United States of America	20	386534.18
Midmarket	Canada	20	510213.975
Enterprise	France	20	3890890.625
Enterprise	United States of America	20	4350605
Small Business	Canada	20	9175849
Small Business	France	20	7389006.5
Enterprise	Germany	20	4086826.25
Channel Partners	France	20	372090.36
Small Business	United States of America	20	11456559
Enterprise	Mexico	20	3315881.25
Small Business	Germany	20	7327648
Channel Partners	Mexico	20	3342291.06

- **Data type:** If you want to change the data type of any column you can use this option. Every column has its data type on the left corner of column header. You can change by clicking on that.

The screenshot shows the Power BI Data Editor interface. On the left, the 'Queries [43]' pane is open, with 'financials' selected. In the main area, a table is displayed with columns 'Segment' and 'Country'. A context menu is open over the 'Count' column, which has a yellow background. The menu is titled 'Data Type: Whole Number' and lists various data types: Decimal Number, Fixed decimal number, Whole Number, Percentage, Date/time, Date, Time, Date/Time/Timezone, Duration, Text, True/False, and Binary. The 'Whole Number' option is highlighted.

Table Data:

Segment	Country	Count
Government	Canada	60
Channel Partners	Canada	20
Enterprise	Canada	20
Midmarket	Canada	20
Small Business	Canada	20

This screenshot shows the same setup as above, but the context menu is now over the 'Sum' column, which has a yellow background. The menu is titled 'Data Type: Sum' and lists the same data types. The 'Whole Number' option is also present here.

Table Data:

Segment	Country	Sum
Government	Canada	60
Channel Partners	Canada	20
Enterprise	Canada	20
Midmarket	Canada	20
Small Business	Canada	20

Note: Character data type accepts other data type values but other data types will not accept character data type values.

- **Use First Row as Header:** It will convert first row values as column headers. It has two options.
 - Use first row as headers
 - Use headers as first row

Use First Row as Headers: promotes the first row as column headers for the selected table.

Ex)

The screenshot shows the Power BI Data Editor with the 'Example (2)' query selected in the 'Queries [43]' pane. The main area displays a table with columns 'A' and 'X'. The first row ('1') contains the values '1' and 'A'. A context menu is open over the 'A' cell in the first row, with the 'Data Type: Text' option selected. The menu also includes 'Use First Row as Headers' and 'Use Headers as First Row'. A tooltip at the bottom right of the menu says: 'Promote the first row of this table into column headers.'

Table Data:

A	X
1	A
2	B
3	C
4	D
5	F
6	G

The screenshot shows the Power BI Data Editor interface with the 'Transform' tab selected. In the ribbon, the 'Transform' tab is highlighted. On the far right of the ribbon, there is a dropdown menu labeled 'Data Type: Whole Number' with a green box around it. Below the ribbon, the 'Queries [43]' pane shows several items, and the main area displays a table with columns labeled 'A' and 'B'. The 'A' column contains values 1 through 5, and the 'B' column contains values B through G. The 'Transform' tab has various buttons like 'Choose Columns', 'Remove Columns', 'Keep Rows', 'Remove Rows', 'Reduce Rows', 'Sort', 'Split Column', 'Group By', and 'Replace Values'. The 'Replace Values' button is also highlighted with a green box.

Use Headers as First Row: Demotes the columns as first row values for the selected table.

Ex)

This screenshot shows the Power BI Data Editor with the 'Transform' tab selected. The 'Use Headers as First Row' button in the ribbon is highlighted with a green box. A tooltip below the button says 'Demote the headers of this table to the first row.' The main area shows a table with columns 'A' and 'X'. The 'A' column has values 1 through 6, and the 'X' column has values A through G. The 'Transform' tab includes buttons for 'Data Type Text', 'Merge Queries', 'Append Queries', 'Combine File', 'Combine', and 'AI Insights'.

This screenshot shows the Power BI Data Editor with the 'Transform' tab selected. The 'Use Headers as First Row' button in the ribbon is highlighted with a green box. A tooltip below the button says 'Demote the headers of this table to the first row.' The main area shows a table with columns 'Column1' and 'Column2'. The 'Column1' column has values 1 through 7, and the 'Column2' column has values X through G. The 'Transform' tab includes buttons for 'Data Type Text', 'Merge Queries', 'Append Queries', 'Combine File', 'Combine', and 'AI Insights'.

➤ **Replace Values:** Replacing existing values with new value for the selected column or you can directly select the cell value to replace with new value.

For Example, I want to replace value “4” with value “5” in the Column1. Then here we go.

This screenshot shows the Power BI Data Editor with the 'Replace Values' dialog box open. The 'Replace Values' button in the ribbon is highlighted with a green box. The dialog box has fields for 'Value To Find' (set to '4') and 'Replace With' (set to '5'). There is also an 'Advanced options' link and 'OK' and 'Cancel' buttons at the bottom. The main area shows a table with columns 'Column1' and 'Column2', where the 'Column1' values are being replaced from 4 to 5.



Result:

Column1	Column2
A	X
1	A
2	B
3	C
5	D
2	F
3	G

- **Merge Queries:** If you want to get the columns from another data set, you have to use Merge queries. By performing different kinds of joins we can merge the columns. In this section, we have two options.

Merge Queries: If you want to bring other table columns for the selected table then we can use this option

Merge Queries as New: If you want to create new data set by merging columns from one table to another table. Both table columns will be present in the new data set.

We can't merge multiple tables at a time; you can merge two tables by using common column. If you want to merge multiple tables, you have to merge two tables first; then merge with other table... like this, you have to do.

- **JOINS in Merge Queries:** We have different kind of join operations are there. Joins means merging columns from multiple tables. For this, join requires a common column.

- ★ Left Outer Join
- ★ Right Outer Join
- ★ Full Outer Join
- ★ Inner Join
- ★ Left Anti
- ★ Right Anti

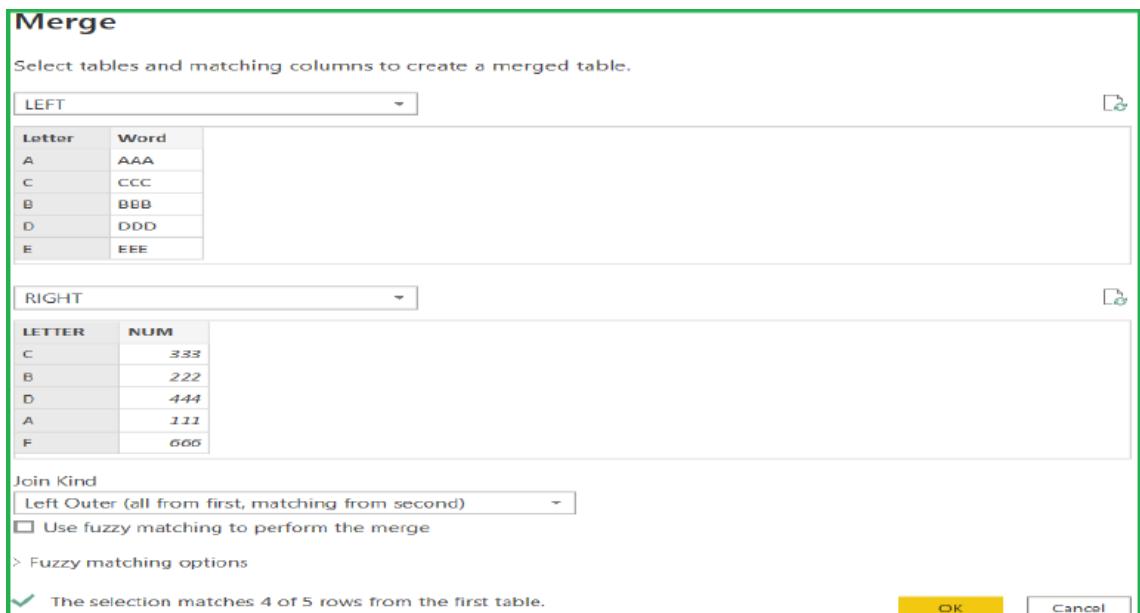
Suppose I am taking below tables for performing all the above join operations.

Left Table

Right Table

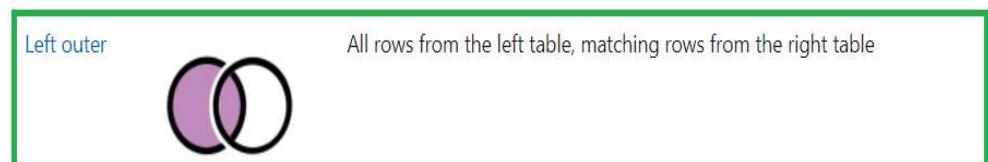
A ^B _C Letter	A ^B _C Word	A ^B _C LETTER	1 ² ₃ NUM
A	AAA	C	333
C	CCC	B	222
B	BBB	D	444
D	DDD	A	111
E	EEE	F	666

Select any table and click on “Merge Queries as New” option it will pop up another window. There you have to select Left Table, Right Table, Common Column and Join Kind. Then join will be performed.



1. **Left Outer Join:** Display all the records from left side table and matching records from right side table. (Unmatched records from Left table and matched records from right table)

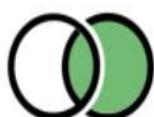
A ^B _C Letter	A ^B _C Word	1 ² ₃ NUM
A	AAA	111
C	CCC	333
B	BBB	222
D	DDD	444
E	EEE	null



- 2. Right Outer Join:** Display all the records from right side table and matching records from left side table.(Unmatched records from right table and matched records from left table)

A ^B _C Letter	A ^B _C Word	1 ² ₃ RIGHT.NUM
A	AAA	111
C	CCC	333
B	BBB	222
D	DDD	444
null	null	666

Right outer



All rows from the right table, matching rows from the left table

- 3. Full Outer Join:** Display all the records from both tables (Matched and Unmatched records)

A ^B _C Letter	A ^B _C Word	1 ² ₃ NUM
A	AAA	111
C	CCC	333
B	BBB	222
D	DDD	444
null	null	666
E	EEE	null

Full outer



All rows from both tables

- 4. Inner Join:** Display all the matching records from both tables (Matched)

A ^B _C Letter	A ^B _C Word	1 ² ₃ NUM
A	AAA	111
C	CCC	333
B	BBB	222
D	DDD	444

Inner

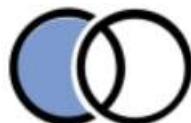


Only matching rows from both tables

5. **Left Anti:** Display the records which are there in left side table but not in right side table.

A ^B _C Letter	A ^B _C Word	1 ² ₃ NUM
E	EEE	null

Left anti

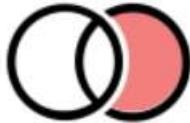


Only rows from the left table

6. **Right Anti:** Display the records which are there in right side table but not in left side table.

A ^B _C Letter	A ^B _C Word	1 ² ₃ NUM
null	null	666

Right anti



Only rows from the right table

- **Append Queries:** Combining rows of the multiple data sets is called as “**Append Queries**”. Here we can combine multiple table rows at a time. In this section, there are set of rules to follow.

- Column names should be same in all data sets
- Data types should be same in all data sets

In this section also, we have two options for appending multiple tables.

Append Queries: If you want to combine rows of another table for the selected table then we can use this option(Selected table itself)

Append Queries as New: If you want to create new data set by combining rows of multiple tables, we can use this option.

For example, I am taking below 3 tables and I want to append all the rows of these tables. Select any table and click on “**Append Queries as New**”

Append1

1 ² ₃ Id	A ^B _C Name
1 AA	
2 BB	
3 CC	
4 DD	

Append2

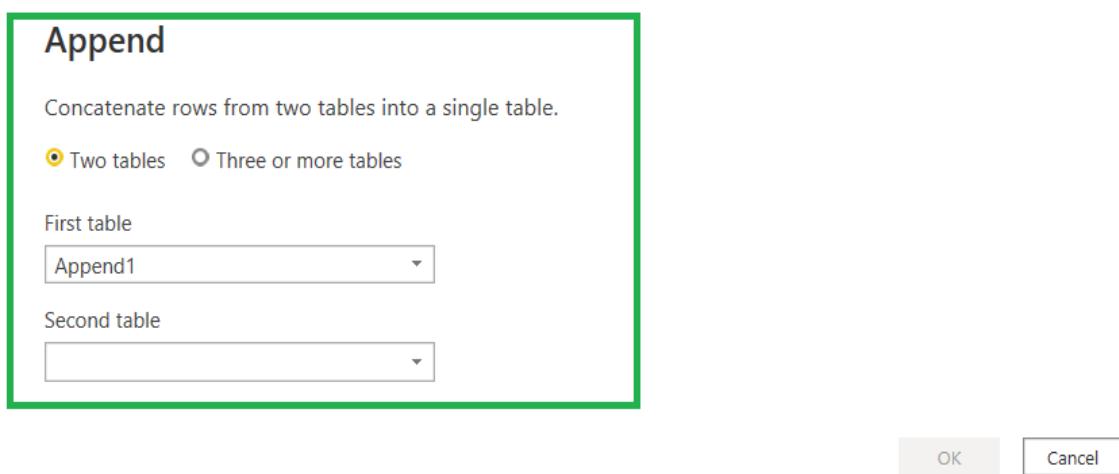
1 ² ₃ Id	A ^B _C Name
5 FF	
6 GG	

Append3

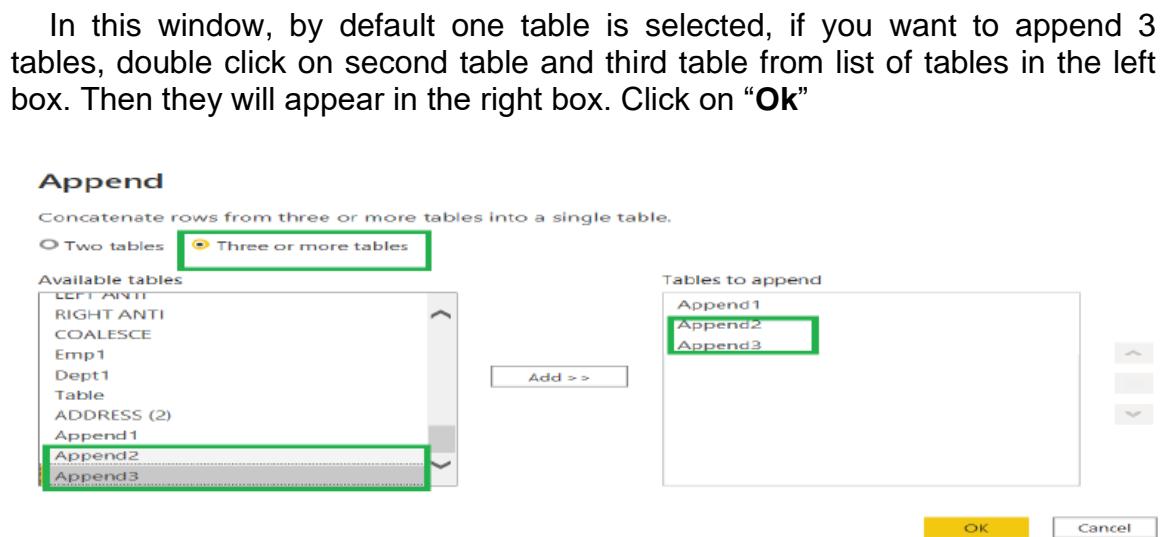
1 ² ₃ Id	A ^B _C Name
7 ZZ	

Select the “Append1” data set and click on “Append Queries as New” option then it will pop a new window. There you can select either “Two tables” option or “Three or more tables” option to append.

By default, it will be in “Two tables” option. If you want more than two tables to combine then select the “Three or more tables” option.



Here, I want to combine 3 tables “Append1”, “Append2”, “Append3”. Then I will select “Three or more tables” option. Once selected; it will pop up another window. Click on “Ok”



Then the result will be appeared in the new data set. See the below screen shot.

The screenshot shows the Power BI Data Editor interface. In the top ribbon, the 'Transform' tab is selected. A green box highlights the 'Append Queries as New' button in the 'Transform' group. To the right, a tooltip says: 'Append this query with other queries in this file to create a new query.' Below the ribbon, the 'Queries [47]' pane shows several tables: 'NAME', 'Append1', 'Append2', 'Append3', and 'Append4' (which is selected). The main workspace displays a table named 'Table.Combine([Append1, Append2, Append3])' with columns 'Id' and 'Name'. The data is as follows:

	Name
1	AA
2	BB
3	CC
4	DD
5	FF
6	GG
7	ZZ

Note: No of columns might be different in all the tables, even though “**Append Queries**” will work.

If the column names and data types are different in multiple tables, then “**Append queries**” option will work.

Append1

1 ² Id	A ^B C Name
1	AA
2	BB
3	CC
4	DD

Append2

1 ² Id1	A ^B C Name
5	FF
6	GG

Append3

1 ² Id2	A ^B C Name
7	ZZ

1 ² Id	A ^B C Name	1 ² Id1	1 ² Id2
1	AA		null
2	BB		null
3	CC		null
4	DD		null
null	FF		5
null	GG		6
null	ZZ		7

If you create any table from “**Enter Data**” option; by default it will consider any value as text data type.

Once you click on “**ok**” it will have both “**Source**” and “**Change Type**” option will be added in the “**Applied Steps**” section.

In “**Excel**” data also same will be happened. Change type step will assign data type for the data based on the data we have in the table.

We should not delete “**Change Type**” option. If you change it, sub-sequent applied steps will be affected.

❖ Transform Tab:

- **Transpose:** It will treat the table rows as columns and columns as rows. Select the table that you want to apply transpose. Then click on “Transpose” option. See the below screen shots.

The screenshot shows the Power BI interface with the 'Transform' tab selected. In the ribbon, the 'Transpose' icon is highlighted. A table with three columns ('Column1', 'Column2', 'Column3') and three rows is displayed. The first row ('Name', 'id', 'Sal') is highlighted.

Result:

The screenshot shows the Power BI interface with the 'Transform' tab selected. The 'Transpose' icon is still highlighted. The same table is shown, but the rows have been rearranged into columns ('Column1', 'Column2', 'Column3'). The second row ('XYZ', '123', '20000') is highlighted.

- **Reverse Rows:** Whatever column you have selected in the data set, that column values (rows) will be reversed. Select the column from any table, and then click on “Reverse Rows” option. Then rows will be reversed based on that selected column.

Ex)

The screenshot shows the Power BI interface with the 'Transform' tab selected. The 'Reverse Rows' icon is highlighted. A table with two columns ('A' and 'X') and seven rows is displayed. The first row ('1', 'A') is highlighted.

- **Count Rows:** Returns the number of rows for the selected table. Select the data set and click on “Count Rows” option. Then you can see the total number of rows as a result. This option will be used for checking for the purpose.

Ex) I am selecting “Example” table and click on “Count Rows” option. Then Power BI automatically counts the number of rows in this table and returns the value. It will be visible as a result in place of table in the data sets category.

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'Text Column' group, the 'Count Rows' button is highlighted with a green box. Below the ribbon, the 'Queries [48]' pane is visible, with 'Example' selected. The main area displays a table with columns A and X, containing the values 1 through 7 respectively. The entire table is highlighted with a green box.

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'Text Column' group, the 'RowCount' button is highlighted with a green box. Below the ribbon, the 'Queries [48]' pane is visible, with 'Example' selected. The main area displays a table with a single column labeled '7'. The entire table is highlighted with a green box.

- **Detect Data type:** In a table, we have multiple columns, and finding the data type of each column is difficult.

To overcome this, we have to select all the columns at a time and click on “**Detect Data Types**”. Then, Power BI automatically returns the data type for each and every column in that table.

Note: To select all columns at a time, we have to press “**CTRL+A**” buttons in the keyboard.

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'Text Column' group, the 'Detect Data Type' button is highlighted with a green box. Below the ribbon, the 'Queries [48]' pane is visible, with 'Example' selected. The main area displays a table with columns A and X, containing the values 1 through 7 respectively. The entire table is highlighted with a green box. On the right side, the 'APPLIED STEPS' pane shows a step named 'Changed Type'.

- **Rename:** If you want to rename the column, you have to select the column and click on “Rename”.

Then it will be converted as an editable mode, there you can change. Or else you can directly double click on the “**Column header**”.

Or else you can select the column and right click on that column, there you can see rename option. You have already seen in “Enter Data” option.

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'Data Type' section, the 'Replace Values' button is highlighted with a green box. Below it, the 'Rename' button is also highlighted with a green box. A tooltip window titled 'Rename' appears, stating 'Change the name of the currently selected column.' To the right, a table view shows a single column 'A' with values from 1 to 7 and letters A through G. The 'Example' query is selected in the left pane.

The screenshot shows a context menu for a selected column 'A'. The 'Rename...' option is highlighted with a green box. The menu includes options like Copy, Remove, Replace Values..., Split Column, Unpivot Columns, and others. The 'Example' query is selected in the left pane, and a table view is visible on the right.

- **Replace Values:** In Home tab, you will see the same option. Additionally it will have “Replace Errors” in this section. Replace values will replace a selected value with new value.

Replace Errors means, Replace all errors in the selected column with specified value. Select the table and click on “**Replace Errors**” then it will pop a new window to ask new value.

The screenshot shows the Power BI desktop interface with the 'Transform' tab selected. In the ribbon, under 'Data Type', the 'Replace Values' dropdown is open, showing 'Replace Errors' as the second option. A tooltip for 'Replace Errors' states: 'Replace all errors in the currently selected columns with the specified value.' Below the ribbon, the 'Queries [48]' pane shows a list of transformations, with 'Example' highlighted. The main area displays a table with columns 'A' and 'X'. Row 5 of column 'A' contains the value 'Error'. This row is selected, and the tooltip for 'Replace Errors' is visible.

This screenshot shows the 'Replace Errors' dialog box. It has a text input field labeled 'Value' containing the number '3'. The 'OK' button is highlighted with a green border, while the 'Cancel' button is greyed out. The background shows the Power BI interface with the 'Transform' tab selected and the 'Example' transformation applied to the table.

Result:

The screenshot shows the Power BI interface with the 'Transform' tab selected. The 'Example' transformation is applied to the table, changing the value 'Error' in row 5 of column 'A' to '3'. The 'Applied Steps' pane on the right shows the step 'Replaced Errors'.

Note: In Home Tab, only one option in “Replace Values” but in “Transform Data” two options are there.

- Replace values
- Replace Errors

➤ **Fill:** It will fill the cell values to the neighbouring empty cells in the currently selected column. In this Section we have two options

- Down
- Up

Down: Wherever the null values are there in the column, those are filled with the neighboring up value.

Ex)

Result

The screenshot shows the Power BI desktop interface with the 'Transform' tab selected. In the ribbon, under the 'Data Type' section, the 'Fill' button is highlighted. A tooltip for the 'Down' button states: 'Fill down cell values to neighboring empty cells in the currently selected columns.' Below the ribbon, the 'Queries [49]' pane shows various steps, with 'Fill' selected. The main area displays a table with columns 'Id' and 'Value'. Rows 1 through 7 have values 1, null, 3, null, null, 5, and 6 respectively. Row 5 is highlighted with a green box.

Up: Wherever the null values are there in the column, those are filled with the neighbouring down value.

Ex)

Result

This screenshot is similar to the previous one but with the 'Up' button highlighted in the ribbon's 'Fill' dropdown. The tooltip for 'Up' says: 'Fill up cell values to neighboring empty cells in the currently selected columns.' The table data remains the same, but row 2 now contains the value 3, and rows 4 and 5 now contain the value 5, demonstrating the 'fill up' operation.

- **Pivot Column:** It will convert rows into columns based on the selected column, nested columns will not support here. That means it will not apply on multiple columns.

Note: Here rows will be converted as columns and shows the count of values in that column.

The screenshot shows the 'Transform' tab selected in Power BI. The 'Pivot Column' button in the ribbon is highlighted. A tooltip for 'Pivot Column' says: 'Pivot the selected column into multiple columns based on the distinct values in the column.' The 'Properties' pane on the right shows 'Name' set to 'Numeric'. The main area shows a table with a single column 'L2.x' containing values 1, 2, 1, 2, 3, 1, 1. This column is being pivoted into multiple columns based on its distinct values (1, 2, 3).

➤ **Unpivot Columns**: It will convert columns into rows. We have three options in this section.

- **Unpivot columns**: It will apply unpivot on selected columns
- **Unpivot other columns**: It will apply unpivot on other than selected columns
- **Unpivot only selected columns**: It will apply unpivot on elected columns

If you want to convert columns into rows, you have to select more than one column. After selection, click on “**Unpivot Columns**” then automatically selected columns will be converted into rows. These represented as “**Attribute-value**” pair.

Ex)

Result:

	Attribute	Value
1	Column1	Name
2	Column2	XYZ
3	Column1	id
4	Column2	123
5	Column1	Sal
6	Column2	20000

➤ **Move**: If you want to move the column from current position to left or right or to the beginning or to the end then we can use this option. In this section, we have four options.

- Left
- Right
- To beginning
- To end

Left: Selected column will be moved to the left side

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Move' dropdown menu, the 'Left' option is highlighted with a green box. Below the ribbon, a table named 'Table1' is displayed with four columns: 'Char', 'Value', 'Symbol', and 'Type'. The 'Char' column contains values A through J. The 'Value' column contains numerical values. The 'Symbol' column contains symbols like \$, %, etc. The 'Type' column contains text. The 'Char' column is currently selected.

Result:

The screenshot shows the result of moving the 'Char' column to the left. The table now has four columns: 'Char', 'Value', 'Symbol', and 'Type'. The 'Char' column is positioned at the far left, followed by 'Value', 'Symbol', and 'Type'. The data remains the same as in the original table.

Right: Selected column will be moved to the right side

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Move' dropdown menu, the 'Right' option is highlighted with a green box. Below the ribbon, a table named 'Table1' is displayed with four columns: 'Char', 'Value', 'Symbol', and 'Type'. The 'Char' column contains values A through J. The 'Value' column contains numerical values. The 'Symbol' column contains symbols like \$, %, etc. The 'Type' column contains text. The 'Char' column is currently selected.

Result:

The screenshot shows the result of moving the 'Char' column to the right. The table now has four columns: 'Value', 'Symbol', 'Type', and 'Char'. The 'Value' column is at the far left, followed by 'Symbol', 'Type', and 'Char'. The data remains the same as in the original table.

To Beginning: Selected column will be moved to the starting position of the table.

The screenshot shows the Power BI Data Editor interface. A context menu is open over a column named "Symbol". The menu path "Move > To Beginning" is highlighted with a red box. The table contains the following data:

	Symbol	Char	Value
1	!	A	55
2	@	B	12
3	#	C	1
4	\$	D	3
5	%	D	5
6	^	G	2
7	&	H	55
8	*	J	12

Result:

The screenshot shows the Power BI Data Editor interface. The "Symbol" column has been moved to the first position. The formula bar shows the updated code: `= Table.ReorderColumns(#"Changed Type", {"Symbol", "Numeric", "Char", "Value"})`. The table now looks like this:

	Symbol	Char	Value
1	!	A	55
2	@	B	12
3	#	C	1
4	\$	D	3
5	%	D	5
6	^	G	2
7	&	H	55
8	*	J	12

To End: Selected column will be moved to the end position of the table.

The screenshot shows the Power BI Data Editor interface. A context menu is open over a column named "Symbol". The menu path "Move > To End" is highlighted with a red box. The table contains the following data:

	Symbol	Char	Value
1	!	A	55
2	@	B	12
3	#	C	1
4	\$	D	3
5	%	D	5
6	^	G	2
7	&	H	55
8	*	J	12

Result:

The screenshot shows the Power BI Data Editor interface. The "Symbol" column has been moved to the last position. The formula bar shows the updated code: `= Table.ReorderColumns(#"Changed Type", {"Numeric", "Char", "Value", "Symbol"})`. The table now looks like this:

	Numeric	Char	Value	Symbol
1		A	55	!
2		B	12	@
3		C	1	#
4		D	3	\$
5		D	5	%
6		G	2	^
7		H	55	&
8		J	12	*

Note: We can directly drag the column from one place to another place in the table.

- **Convert to list:** Currently selected column will be converted into list; remaining columns will be removed from the table.

Select the column and click on “Convert to list” option. Then you can see only one column as list.

Result:

- **Format:** It will change case of the text or cleanse the text. If you select any column in a table, in that column each and every cell value will be converted other case. We have below options in “Format”.

- Lower Case
- Upper Case
- Capitalize Each Word
- Trim
- Clean
- Add prefix
- Add suffix

Lower Case: Selected column values will be converted into Lower Case letters

Result:

	Lower	Upper	Sentence	Space
1	abril	Mara	United States	Space
2	hashimoto	Philip	freat Britain UK	12222
3	gent	Kathleen	france	ABC 12A
4	hanner	Nereida	United States	22F
5	magwood	Gaston	United States	23223323
6	brumm	Etta	IONDON	223232
7	hurn	Earlean	india	AFDFD
8	melgar	Vincenza	united arab Emirates	ADF
9	weiland	Fallon	INDIA	
10	winward	Arcelia	aruna chal pradesh	DF
11	bouska	Franklyn	china	FDDDFDF
12	unknow	Sherron	france	DFD

Upper Case: Selected column values will be converted into Upper Case letters

Result:

	Lower	Upper	Sentence	Space
1	abril	MARA	United States	Space
2	hashimoto	PHILIP	freat Britain UK	12222
3	gent	KATHLEEN	france	ABC 12A
4	hanner	NEREIDA	United States	22F
5	magwood	GASTON	United States	23223323
6	brumm	ETTA	IONDON	223232
7	hurn	EARLEAN	india	AFDFD
8	melgar	VINCENZA	united arab Emirates	ADF
9	weiland	FALLON	INDIA	
10	winward	ARCELIA	aruna chal pradesh	DF
11	bouska	FRANKLYN	china	FDDDFDF
12	unknow	SHERRON	france	DFD

Capitalize Each Word: Convert the first letter of each word in the selected columns into an upper case letter.

Screenshot of Power BI Data Editor showing the 'Format' ribbon tab selected. A context menu is open over the 'Sentence' column, with the 'Capitalize Each Word' option highlighted. The tooltip for 'Capitalize Each Word' states: "Convert the first letter of each word in the selected columns into an uppercase letter." The table contains 12 rows of sentence fragments.

ABC Sentence	ABC Lower
United States	abril
Great Britain UK	hashimoto
France	gent
United States	hanner
United States	magwood
London	brumm
India	hurn
United Arab Emirates	melgar
India	weland
Aruna Chal Pradesh	winward
China	bouska
France	unknow

Result:

Screenshot of Power BI Data Editor showing the result of applying the 'Capitalize Each Word' transformation. The 'Format' ribbon tab is still selected. The 'Space' column has been added and populated with the original values from the 'Sentence' column. The 'Lower' and 'Upper' columns now contain capitalized versions of the words.

ABC Sentence	ABC Lower	ABC Upper	ABC Space
United States	abril	MARA	Space
Great Britain UK	hashimoto	PHILIP	12222
France	gent	KATHLEEN	ABC 12A
United States	hanner	NEREIDA	22F
United States	magwood	GASTON	23223323
London	brumm	ETTA	223232
India	hurn	EARLEAN	AFDFO
United Arab Emirates	melgar	VINCENZA	ADF
India	weland	FALLON	
Aruna Chal Pradesh	winward	ARCELIA	DF
China	bouska	FRANKLYN	FDDFDF
France	unknow	SHERRON	DFD

Trim: Remove leading and trailing whitespaces from each cell in the selected columns

Screenshot of Power BI Data Editor showing the 'Format' ribbon tab selected. A context menu is open over the 'Space' column, with the 'Trim' option highlighted. The tooltip for 'Trim' states: "Remove leading and trailing whitespaces from each cell in the selected columns." The table contains 12 rows of space-related values.

ABC Space	ABC Sentence	ABC Lower	ABC Upper
Space	United States	United States	
12222	Great Britain UK	Great Britain UK	
ABC 12A	France	France	
22F	United States	United States	
23223323	United States	United States	
223232	London	London	
AFDFO	India	India	
ADF	United Arab Emirates	United Arab Emirates	
DF	India	India	
FDDFDF	Aruna Chal Pradesh	Aruna Chal Pradesh	
DFD	China	China	
DFD	France	France	

Result:

Screenshot of Power BI Data Editor showing the result of applying the 'Trim' transformation. The 'Format' ribbon tab is still selected. The 'Space' column has been removed, and the 'Lower' and 'Upper' columns now contain the trimmed versions of the words from the original 'Sentence' column.

ABC Space	ABC Sentence	ABC Lower	ABC Upper
Space	United States	abril	MARA
12222	Great Britain UK	hashimoto	PHILIP
ABC 12A	France	gent	KATHLEEN
22F	United States	hanner	NEREIDA
23223323	United States	magwood	GASTON
223232	London	brumm	ETTA
AFDFO	India	hurn	EARLEAN
ADF	United Arab Emirates	melgar	VINCENZA
DF	India	weland	FALLON
FDDFDF	Aruna Chal Pradesh	winward	ARCELIA
DFD	China	bouska	FRANKLYN
DFD	France	unknow	SHERRON

Clean: Remove non-printable characters in the selected columns

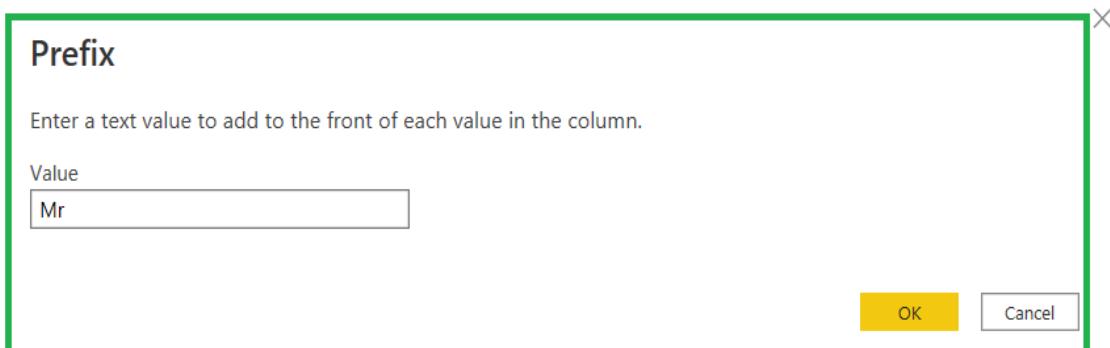
Ex)



Add Prefix: Adds a specified text to the front of each value in the selected column. Select the column and click on “Add Prefix”.

Then it will pop a window to ask which text we need to put as a prefix, fill there, automatically it will be applied to each cell value in the selected column.

The screenshot shows the Power BI Data Editor interface. A context menu is open over the 'Lower' column, with 'Format' selected. In the 'Format' menu, 'Add Prefix' is highlighted with a green box. A tooltip for 'Add Prefix' appears, stating: 'Adds a specified text value to the front of each value in the selected column.' The main table view shows columns 'Lower', 'Space', and 'Upper' with various data rows.



Result:

The screenshot shows the Power BI Data Editor after applying the 'Add Prefix' operation. The 'Lower' column now contains the prefix 'Mr' followed by the original values. The other columns remain unchanged.

	Lower	Space	Sentence	Upper
1	Mr Abril	Space	United States	Mara
2	Mr Hashimoto	12222	Great Britain UK	Philip
3	Mr Gent	ABC 12A	France	Kathleen
4	Mr Hanner	22F	United States	Nereida
5	Mr Magwood	23223323	United States	Gaston
6	Mr Brumm	223232	London	Etta
7	Mr Hurn	AFFDF	India	Earlean
8	Mr Melgar	ADF	United Arab Emirates	Vincenza
9	Mr Welland	~	India	Fallon
10	Mr Winward	DF	Aruna Chal Pradesh	Arcelia
11	Mr Bouska	FDDFDF	China	Franklyn
12	Mr Unknow	DFD	France	Sherron

Add Suffix: Adds a specified text to the end of each value in the selected column. Select the column and click on “Add Suffix”.

Then it will pop a window to ask which text we need to put as a suffix, fill there, automatically it will be applied to each cell value in the selected column.

The screenshot shows the Power BI Data Editor interface. A context menu for the 'Upper' column is open, with the 'Format' option selected. In the 'Format' menu, the 'Add Suffix' option is highlighted. A pop-up dialog box titled 'Suffix' is displayed, asking for a value to add to the end of each value in the column. The input field contains 'Mr'. At the bottom right of the dialog are 'OK' and 'Cancel' buttons, with 'OK' being the active button.

Result:

	Upper	Lower	Space	Sentence
1	Mara	Mr Abril	Space	United States
2	Philip	Mr Hashimoto	12222	Great Britain UK
3	Kathleen	Mr Gent	ABC 12A	France
4	Nereida	Mr Hanner	22F	United States
5	Gaston	Mr Magwood	23223323	United States
6	Etta	Mr Brumm	223232	London
7	Earlean	Mr Horn	AFDFD	India
8	Vincenza	Mr Melgar	ADF	United Arab Emirates
9	Fallon	Mr Welland	~	India
10	Arcelia	Mr Winward	DF	Aruna Chal Pradesh
11	Franklyn	Mr Bouska	FDDFDF	China
12	Sherron	Mr Unknown	DFD	France

- **Merge Columns:** To combine the multiple columns into single column we have to use merge columns.

Merge columns require at least two columns. If one column is selected then this option will be in disable mode.

Here we need to pass separator to differentiate the merged column. You can select any separator from the list or you can give custom separator also. Finally you can give name of the merged column.

Note: Here we can only combine multiple columns of a same data set. We can't combine columns of different data sets into one column here.

Ex) I have selected two columns and clicked on “Merge Columns” option. Then pop will be opened, there you have to provide separator (Delimiter) and name of the merged column.

The screenshot shows the Power BI 'Transform' ribbon tab selected. In the 'Text Column' section, the 'Merge Columns' button is highlighted with a green box. A tooltip for 'Merge Columns' is displayed, stating: 'Concatenate the currently selected columns into one column.' Below the ribbon, the Power BI interface shows a table with two columns: 'Lower' and 'Upper'. The 'Lower' column contains names like Abril, Hashimoto, Gent, Hanner, Magwood, Brumm, Hurn, Melgar, Weiland, Winward, Bouska, and Unknown. The 'Upper' column contains Mara, Philip, Kathleen, Nereida, Gaston, Etta, Earlean, Vincenza, Fallon, Arcelia, Franklyn, and Sherron. To the right of the table, there is a vertical list of country names: United States, Great Britain, UK, France, United States, United States, London, India, United Arab Emirates, India, Aruna Chal Pradesh, China, and France.

Here I have given “|” as delimiter in “Custom” option then i have given column name as “**Merged Column**”

Merge Columns

Choose how to merge the selected columns.

Separator

--Custom--

New column name (optional)

Merged Column

OK

Cancel

Result:

AB_C Merged Column
Abril Mara
Hashimoto Philip
Gent Kathleen
Hanner Nereida
Magwood Gaston
Brumm Etta
Hurn Earlean
Melgar Vincenza
Weiland Fallon
Winward Arcelia
Bouska Franklyn
Unknow Sherron

- **Extract:** It will extract characters from text values from the selected column. In this section we have lot of options.

- Length
- First Characters
- Last Characters
- Range
- Text Before Delimiter
- Text After Delimiter

○ Text Between Delimiters

Length: It will return the length of the values for each cell in the selected column. Select the column name and click on “Length” Option.

The screenshot shows the Power Query Editor interface. The 'Transform' tab is selected. A context menu is open over a column named 'Column1'. The 'Length' option is highlighted with a green box. The tooltip for 'Length' states: 'Return the length of the text in the selected columns.' Other options like 'First Characters', 'Last Characters', 'Range', 'Text Before Delimiter', 'Text After Delimiter', and 'Text Between Delimiters' are also listed in the menu.

Result:

The screenshot shows the Power Query Editor after applying the 'Length' function. The 'Column1' column now contains numerical values representing the length of the strings in the original column. The data is as follows:

Row	Value
1	37
2	36
3	57
4	50

First Characters: Returns a specified number of characters from the start of each value in the selected column.

The screenshot shows the Power Query Editor interface. The 'Transform' tab is selected. A context menu is open over a column named 'Column1'. The 'First Characters' option is highlighted with a green box. The tooltip for 'First Characters' states: 'Return a specified number of characters from the start of each value in this column.' Other options like 'Last Characters', 'Range', 'Text Before Delimiter', 'Text After Delimiter', and 'Text Between Delimiters' are also listed in the menu.

Select a column and click on “First Characters” option, it will pop up a window. Here we need to pass “Count”. Based on this option “Extract” will be happened.



Result:

A B C	Column1
8-81	
6-52	
321-	
1-1A	

Last Characters: Returns a specified number of characters from the end of each value in the selected column.

The screenshot shows the Power Query ribbon with the 'Transform' tab selected. In the 'Text' section of the ribbon, the 'Extract' icon is highlighted, and its dropdown menu is open, showing the 'Last Characters' option. A tooltip for 'Last Characters' is displayed, stating: 'Return a specified number of characters from the end of each value in this column.' Below the ribbon, the Power Query editor window shows a table named 'Column1' with four rows of data: '8-81/benz circle/vijayawada/520010/AP', '6-521-A/Arundal pet/Guntur/522601/AP', '321-4a/Maharani veedhi/Marripalem/Visakhapatnam/5...', and '1-1A/Yellareddi guda/Hyderabad/Telangana/508254/TN'. The 'Queries [54]' pane on the left lists several examples, with 'ADDRESS' selected.

Select a column and click on “**Last Characters**” option, it will pop up a window. Here we need to pass “**Count**”. Based on this option “Extract” will be happened.



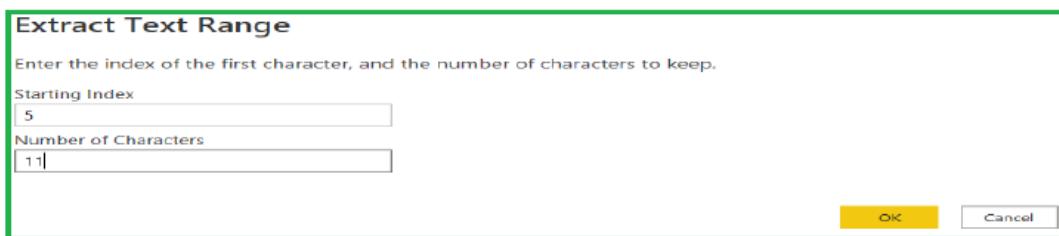
Result:

A B C	Column1
	/520010/AP
	/522601/AP
	/530020/AP
	/508254/TN

Range: Return a specified number of characters from each value in the selected column, starting at specified index.

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Text' section of the ribbon, the 'Extract' dropdown is open, and the 'Range' option is highlighted with a green box. A tooltip for 'Range' is displayed, stating: 'Return a specified number of characters from each value in this column, starting at a specified index.' Below the ribbon, the 'Queries [54]' pane shows an example query named 'ADDRESS'. The main area displays a table named 'Column1' with four rows of address data.

Select a column and click on “**Range**” option, it will pop up a window. Here we need to pass “**Starting index**” and “**Number of Characters**”.



Result:

The screenshot shows the transformed data in 'Column1'. The original data contained full addresses. After applying the 'Range' extract operation with a starting index of 5 and 11 characters, the resulting data is: 'benz circle', '-A/Arundal', 'a/Maharani', and 'Yellareddi', which are the street names extracted from the addresses.

Text Before Delimiter: Returns the text that occurs before a delimiter. By default it will scan from the left side.

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Text' section of the ribbon, the 'Extract' dropdown is open, and the 'Text Before Delimiter' option is highlighted with a green box. A tooltip for 'Text Before Delimiter' is displayed, stating: 'Return the text that occurs before a delimiter.' Below the ribbon, the 'Queries [54]' pane shows an example query named 'ADDRESS'. The main area displays a table named 'Column1' with four rows of address data.



Select column and click on “**Text Before Delimiter**”. Then it will pop up a window. There we have to pass “**Delimiter**”. You can extract the text from right side also by changing the “**Scan for the delimiter**” option in “**Advanced options**”. You can skip the number of delimiters there if you want. Click on “**OK**”

Result:

Column1
8-81
6-521-A
321-4a
1-1A

Text After Delimiter: Returns the text that occurs after a delimiter. By default it will scan from the left side. You can change it to right side if you want.

The screenshot shows the Power BI desktop interface with the 'Transform' tab selected. In the 'Text' ribbon group, the 'Extract' dropdown is open, and 'Text After Delimiter' is highlighted. Below the ribbon, a table named 'Column1' contains four rows of address data. To the right of the table, the 'Text After Delimiter' dialog box is visible, showing the configuration for extracting text after the delimiter '/'.

Select column and click on “**Text After Delimiter**”. Then it will pop up a window. There we have to pass “**Delimiter**”. You can extract the text from right side also by changing the “**Scan for the delimiter**” option in “**Advanced options**”. You can skip the number of delimiters there if you want. Click on “**OK**”



Result:

Column1
benz circle/vijayawada/520010/AP
Arundal pet/Guntur/522601/AP
Maharani veedhi/Marripalem/Visakhapatnam/530020/AP
Yellareddi guda/Hyderabad/Telangana/508254/TN

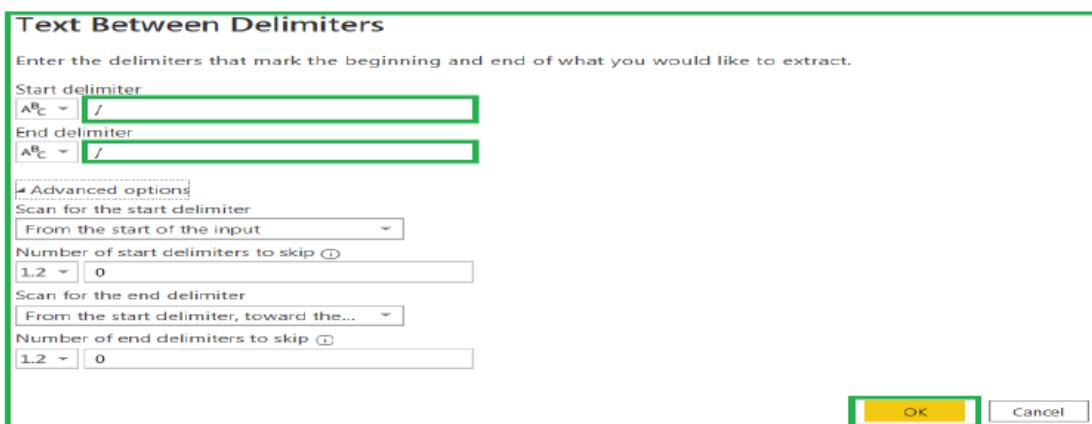
Text Between Delimiters: Returns the text that occurs between two delimiters. Here we need two mention two delimiters in the pop up window.

The screenshot shows the Power BI 'Transform' ribbon tab selected. In the 'Text' section of the ribbon, the 'Extract' dropdown is open, with the option 'Text Between Delimiters' highlighted. Below the ribbon, a list of queries is shown, and a preview window displays four rows of address data from 'Column1'. A tooltip for 'Text Between Delimiters' is visible, stating: 'Return the text that occurs between two delimiters.'

Select column and click on “**Text Between Delimiters**”. Then it will pop up a window. There we have to pass “**Delimiter**”.

You can extract the text from right side also by changing the “**Scan for the start delimiter**” option in “**Advanced options**”.

You can skip the number of delimiters for first delimiter or second also there if you want. Click on “**OK**”



Result:

Column1
benz circle
Arundal pet
Maharani veedhi
Yellareddi guda

➤ **Statistics:** It will perform statistical operations on the numeric or decimal columns. In this section, we have below options.

- Sum
- Minimum
- Maximum
- Median
- Average
- Standard Deviation
- Count Values

- Count Distinct Values

Sum: It will return the sum of all the values in the currently selected columns.
Select the column and click on “**Sum**” option.

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Text Column' section of the ribbon, the 'Sum' button is highlighted with a green box. A tooltip for 'Sum' is displayed, stating: 'Return the sum of all the values in the currently selected column.' Below the ribbon, a table view shows a single column named 'Numeric' with 8 rows of data. The 'Sum' button is also highlighted in the formula bar above the table.

Result:

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Text Column' section of the ribbon, the 'Sum' button is highlighted with a green box. A tooltip for 'Sum' is displayed, stating: 'Return the sum of all the values in the currently selected column.' Below the ribbon, a table view shows a single column named 'Numeric' with 8 rows of data. The 'Sum' button is also highlighted in the formula bar above the table. The result of the sum operation is displayed as 19 in the formula bar.

Minimum: It will return the minimum of all the values in the currently selected column.

Select the column and click on “**Minimum**” option.

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Text Column' section of the ribbon, the 'Minimum' button is highlighted with a green box. A tooltip for 'Minimum' is displayed, stating: 'Return the minimum of all the values in the currently selected column.' Below the ribbon, a table view shows a single column named 'Numeric' with 8 rows of data. The 'Minimum' button is also highlighted in the formula bar above the table.

Result:

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Text Column' section of the ribbon, the 'Minimum' button is highlighted with a green box. A tooltip for 'Minimum' is displayed, stating: 'Return the minimum of all the values in the currently selected column.' Below the ribbon, a table view shows a single column named 'Numeric' with 8 rows of data. The 'Minimum' button is also highlighted in the formula bar above the table. The result of the minimum operation is displayed as 1 in the formula bar.

Maximum: It will return the maximum of all the values in the currently selected column.

Select the column and click on “**Maximum**” option.

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. A context menu is open over a column named 'Value' in a table named 'Table.TransformColumnTypes'. The menu is titled 'Maximum' and includes options like Sum, Minimum, Median, Average, Standard Deviation, Count Values, and Count Distinct Values. The 'Maximum' option is highlighted.

Result:

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. The formula bar displays the expression: `= List.Max(#"Changed Type"[Numeric])`. The result of the formula, which is the maximum value from the 'Value' column, is shown in the preview pane as '5'.

Median: Returns the median of the values in the currently selected column.

Ex) Median

1, 3, 3, **6**, 7, 8, 9
Median = **6**

1, 2, 3, **4, 5**, 6, 8, 9
Median = $(4 + 5) \div 2$
= **4.5**

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. A context menu is open over a column named 'Value' in a table named 'Table.TransformColumnTypes'. The menu is titled 'Median' and includes options like Sum, Minimum, Maximum, Average, Standard Deviation, Count Values, and Count Distinct Values. The 'Median' option is highlighted.

Result:

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. The formula bar displays the expression: `= List.Median(#"Changed Type"[Numeric])`. The result of the formula, which is the median value from the 'Value' column, is shown in the preview pane as '2'.

Average: Returns the average of all the values in the currently selected column

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the formula bar, the expression `= Table.TransformColumnTypes(Source,{{"Numeric", Int64.Type}})` is entered. A context menu is open over the 'Value' column, with the 'Average' option highlighted. The tooltip for 'Average' states: 'Return the average of all the values in the currently selected column.'

Result:

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the formula bar, the expression `= List.Average(#"Changed Type"[Numeric])` is entered. The result '2.375' is displayed in the preview pane.

Standard Deviation: Returns the standard deviation of values in the selected column

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the formula bar, the expression `= Table.TransformColumnTypes(Source,{{"Numeric", Int64.Type}})` is entered. A context menu is open over the 'Value' column, with the 'Standard Deviation' option highlighted. The tooltip for 'Standard Deviation' states: 'Return the standard deviation of all the values in the currently selected column.'

Result:

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the formula bar, the expression `= List.StandardDeviation(#"Changed Type"[Numeric])` is entered. The result '1.5059406173077154' is displayed in the preview pane.

Count Values: Return the number of non-null values in the currently selected column.

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the formula bar, the expression `= Table.TransformColumnTypes(Source,{{"Numeric", Int64.Type}})` is entered. A context menu is open over the 'Value' column, with the 'Count Values' option highlighted. The tooltip for 'Count Values' states: 'Return the number of non-null values in the currently selected column.'

Result:

Queries [54]

- EMP
- 123 Numeric**
- Emp List

7

Count Distinct Values: Returns the number of unique and non-null values in the currently selected column.

File Home Transform Add Column View Tools Help

Queries [54]

- EMP
- 123 Numeric**
- Emp List
- Example (2)
- Segmentwise Data (Midmarket)
- Qlist
- EMPINFO
- Fill
- FINANCIALS2

Any Column Text Column

123 Numeric Char Value

	Char	Value
1	A	55 1
2	B	12 @
3	C	3 #
4	D	8 \$
5	E	5 %
6	null	2 ^
7	F	55 &
8	G	12 *

Sum Minimum Median Average Standard Deviation Count values Count Distinct Values

10² Trigonometry Rounding Information Date & Time Date Time Duration

Result:

Queries [54]

- EMP
- 123 Numeric**
- Emp List

4

- **Standard:** It will perform basic mathematical operations on the numeric or decimal columns. In this section, we have below options.

- Add
- Multiply
- Subtract
- Divide
- Integer-Divide
- Modulo
- Percentage
- Percent of

Add: Adds a specified value to each number in the selected column. Select Dataset, select column and click on “Add” in “Standard”. Then it will pop up a window. There you have to mention the “Value”. Click on “OK”

File Home Transform Add Column View Tools Help

Queries [54]

- EMP
- 123 Numeric**
- Emp List
- Example (2)
- Segmentwise Data (Midmarket)
- Qlist
- EMPINFO
- Fill
- FINANCIALS2

Any Column Text Column

123 Numeric Char Value

	Char	Value
1	A	55 1
2	B	12 @
3	C	3 #
4	D	8 \$
5	E	5 %
6	null	2 ^
7	F	55 &
8	G	12 *

Add Multiply Subtract Divide Integer-Divide Modulo Percentage Percent Of

10² Trigonometry Rounding Information Date & Time Date Time Duration

Result:

1.2 Numeric
5
5
6
5
7
8
6
9

Multiply: Multiply each value in the selected column with specified value. Select Dataset, select column and click on “Multiply” in “Standard”. Then it will pop up a window. There you have to mention the “Value”. Click on “OK”

The screenshot shows the Power BI desktop interface. The 'Transform' ribbon tab is selected. A context menu is open over a column named 'Numeric'. The 'Multiply' option is highlighted with a green box. A tooltip for 'Multiply' is visible, stating 'Multiplies each number in the selected column by a specified value.'

Multiply

Enter a number by which to multiply each value in the column.

Value

5

OK

Cancel

Result:

1.2 Numeric
5
5
10
5
15
20
10
25

Subtract: Subtract specified value from each number in the selected column. Select Dataset, select column and click on “Subtract” in “Standard”. Then it will pop up a window. There you have to mention the “Value”. Click on “OK”

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the formula bar, there is a complex DAX formula involving 'Table.TransformColumns'. A context menu is open over the 'Value' column, with the 'Subtract' option highlighted. The 'Subtract' option is described as 'Subtracts a specified value from each number in the selected column.'

Subtract

Enter a number to subtract from each value in the column.

Value

OK

Cancel

Result:

1.2 Numeric
1
1
2
1
3
4
2
5

Divide: Divides each value in the selected column by the specified value.

Select Dataset, select column and click on “Divide” in “Standard”. Then it will pop up a window. There you have to mention the “Value”. Click on “OK”

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the formula bar, there is a complex DAX formula involving 'Table.TransformColumns'. A context menu is open over the 'Value' column, with the 'Divide' option highlighted. The 'Divide' option is described as 'Divides each number in the selected column by a specified value.'

Divide

Enter a number by which to divide each value in the column.

Value

OK

Cancel

Result:

1.2 Numeric
1
1
2
1
3
4
2
5

Integer-Divide: Integer-divides each number in the selected column by a specified value.

Select Dataset, select column and click on “**Integer-Divide**” in “**Standard**”. Then it will pop up a window. There you have to mention the “**Value**”. Click on “**OK**”

Integer-Divide

Enter a number by which to integer-divide each value in the column.

Value

OK

Cancel

Result:

1.2 Numeric
1
1
2
1
3
5
2
6

Modulo: Calculates the remainder of dividing each number in the selected column with specified value.

Select Dataset, select column and click on “**Modulo**” in “**Standard**”. Then it will pop up a window. There you have to mention the “**Value**”. Click on “**OK**”

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'Text Column' section of the ribbon, the 'Modulo' function is highlighted with a green box. A tooltip for 'Modulo' is displayed, stating: 'Calculates the remainder of dividing each number in the selected column by a specified value.'

Modulo

Enter a number from which to find the remainder for each value in the column.

Value

4

OK

Cancel

Result:

1.2 Numeric
1
1
2
1
3
0
2
1

Percentage: Calculate a specified percentage of the values in the selected column.

Select Dataset, select column and click on “Percentage” in “Standard”. Then it will pop up a window. There you have to mention the “Value”. Click on “OK”

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'Text Column' section of the ribbon, the 'Percentage' function is highlighted with a green box. A tooltip for 'Percentage' is displayed, stating: 'Calculate a specified percentage of the values in the selected column.'

Percentage

Enter a percentage to apply to each value in the column.

Value

10

OK

Cancel

Result:

1.2 Numeric
0.5
0.5
1
0.5
1.5
2
1
2.5

Percent of: Calculate the values in the selected column as a percentage of a specified value.

Select Dataset, select column and click on “Percent of” in “Standard”. Then it will pop up a window. There you have to mention the “Value”. Click on “OK”

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'More' section of the ribbon, there is a dropdown menu with various mathematical operations. The 'Percent Of' option is highlighted with a green box, and a tooltip below it explains: 'Calculate the values in the selected column as a percentage of a specified value.'

Percent Of

Enter a value to find each value in the selected column as a percentage of it.

Value

OK

Cancel

Result:

1.2 Numeric
50
50
100
50
150
200
100
250

➤ **Scientific:** It will perform scientific operations on the selected columns. In this section, we have below options.

- **Absolute:** Returns absolute values of the selected column
- **Power:** Calculates the power of each value in the selected column

- **Square**: Calculates the square of each number in the selected column.
- **Cube**: Calculates the cube of each number in the selected column.
- **Power**: Calculates the power of each number in the selected column with the specified value.
- o **Square Root**: Calculates square root of each value in the selected column
- o **Exponent**: Calculates exponent of the each number in the selected column
- o **Logarithm**:
 - Base-10: Calculates the base-10 logarithm of each number in the selected column.
 - Natural: Calculates the natural logarithm of each number in the selected column.
- o **Factorial**: Calculates the factorial of each number in the selected column.

Square: Returns the square of the each number in the selected column.

The screenshot shows the Power BI desktop application interface. The 'Transform' ribbon tab is selected. In the 'Power' section of the ribbon, the 'Square' option is highlighted with a green box. A tooltip for 'Square' is visible, stating: 'Return the square of numbers in the selected columns.' Below the ribbon, a table named 'Table.TransformColumns(#Added To Column', {{"Numeric", each _ - 4, type} is displayed. The 'Numeric' column is selected, indicated by a green box around its name in the query list on the left. The table data shows various numerical values.

Result:

Numeric
1
1
4
1
9
16
4
25

Cube: Returns the cube of the each number in the selected column.

The screenshot shows the Power BI Data Editor interface. The 'Transform' tab is active. A context menu is open over a column named '1.2 Numeric'. The 'Power' option is highlighted, and its tooltip is visible: 'Return the cube of numbers in the selected columns.'

Result:

1.2 Numeric
1
1
8
1
27
64
8
125

Power: Returns the numbers in the selected columns raised to a specified power.

The screenshot shows the Power BI Data Editor interface. The 'Transform' tab is active. A context menu is open over a column named '1.2 Numeric'. The 'Power' option is highlighted, and its tooltip is visible: 'Return the numbers in the selected columns raised to a specified power.'

Power

Specify the power to raise to.

Power

OK

Cancel

Result:

1.2 Numeric
1
1
8
1
27
64
8
125

➤ **Trigonometry:** It will perform trigonometric operations in the selected columns. In this section, we have below options.

- **Sine:** Returns the sine of each number in the selected column
- **Cosine:** Returns the cosine of each number in the selected column
- **Tangent:** Returns the tangent of each number in the selected column
- **Arcsine:** Returns the arcsine of each number in the selected column
- **Arccosine:** Returns the arccosine of each number in the selected column
- **Arctangent:** Returns the arctangent of each number in the selected column

➤ **Rounding:** It will perform rounding on numbers in the selected columns. In this section, we have below options.

- **Round Up**
- **Round Down**
- **Round**

Round Up: Round the numbers in the specified column to the next nearest integer value.

12 Round	12 Round - Copy
1.52	1.52
4.5678	4.5678
3.21456	3.21456
5.556	5.556
8.4555	8.4555

 The '12 Round' column contains the original values, and the '12 Round - Copy' column contains the rounded values."/>

Result:

12 Round - Copy
2
5
4
6
9

Round Down: Round the numbers in the specified column to the previous integer value.

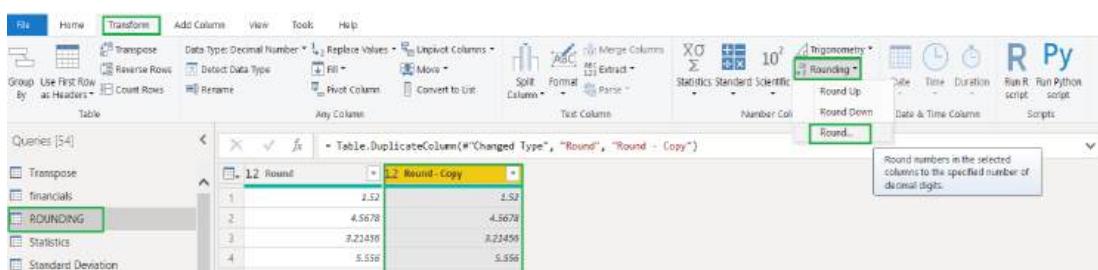
12 Round	12 Round - Copy
1.52	1.52
4.5678	4.5678
3.21456	3.21456
5.556	5.556
8.4555	8.4555

 The '12 Round' column contains the original values, and the '12 Round - Copy' column contains the rounded values."/>

Result:

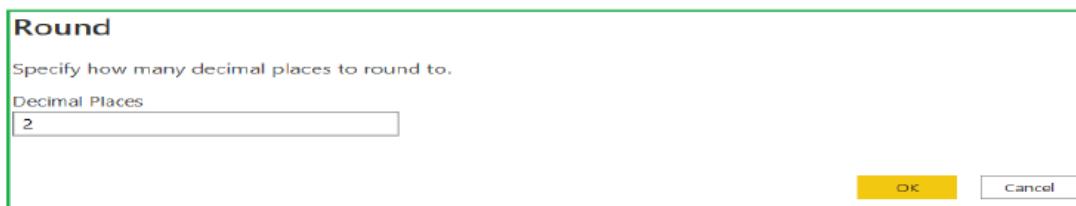
1
4
3
5
8

Round: Round the numbers in the specified column to the specified number of decimal digits. For rounding, it requires decimal column.



The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'Number' section of the ribbon, the 'Rounding' button is highlighted with a green box. A tooltip window is open over the 'Round...' button, stating: 'Round numbers in the selected columns to the specified number of decimal digits.'

Select the decimal column and click on “**Rounding**” option, then select “**Round**”. It will ask you the number of decimal places.



Result:

L2 Round - Copy
1.52
4.57
3.21
5.56
8.46

Ex 1) 4.5687

Here I have given “**Decimal places**” as 2. That means it should return up to two decimal places after rounding. But in this example, we have 4 decimal places. If you apply round on this value below result will come.

Ans) 4.57

Based on the 3rd decimal place 2nd decimal place will be rounded. Here 3rd decimal place is 8; which is greater than 4. Then 1 is added to the 2nd decimal place.

Ex 2) 4.5646

Here 3rd decimal place is 4; which is not greater than 4; then 0 is added will be added to the 2nd decimal place.

Note: Based on “**Decimal places**” argument only rounding will be done. Suppose if we give 2. It will check the 3rd decimal place. If 3rd decimal place is greater than 4; then 1 is added to the 2nd decimal place and display the result. If 3rd decimal place is less than or equal to 4 then 0 will be added to the 2nd decimal place.

- **Information:** It will extract the information about the column. Here we have three options.

- Is Even
- Is Odd
- Sign

Is Even: It will return “True” for even number for each value in the selected column. If not it will return “False”. (A number is divisible by 2 is called as even number)

Result:

	Numeric
1	FALSE
2	FALSE
3	TRUE
4	FALSE
5	FALSE
6	TRUE
7	TRUE
8	FALSE

- Ex 1) 4----→ True
2) 5----→ False

Is Odd: It will return “True” for odd number for each value in the selected column. If not it will return “False”. (A number is not divisible by 2 is called as odd number)

Result:

Char	Value	Symbol
5	55	!
6	12	@
7	2	#
8	3	\$
9	5	%
0	2	*
H	55	&
I	12	*

Ex 1) 5 ---> True

2) 4 ---> False

Sign: It will return the sign of numbers in the selected column.

- **Date:** It will extract date values part or format date values from the date column. It accepts either a date column or date time column.

In this section, we have lot of options.

- Age
- Date Only
- Parse
- Year
 - Year
 - Start of Year
 - End of Year
- Month
 - Month
 - Start of Month
 - End of Month
 - Days in Month
 - Name of Month
- Quarter
 - Quarter of Year
 - Start of Quarter
 - End of Quarter
- Week
 - Week of Year
 - Week of Month

- Start of Week
- End of Week
- o Day
 - Day
 - Day of Week
 - Day of Year
 - Start of Day
 - End of Day
 - Name of Day
- o Combine Date and Time
- o Earliest
- o Latest

Note: If you want to enter date values in power bi, you have to mention “YYYY-MM-DD” format only. Otherwise power bi will not understand properly.

Age: Returns the duration between the current local time and the values in the selected column.

Select “Date” column and click on drop down menu of “Date” option, then select “Age” option.

The screenshot shows the Power BI ribbon with the 'Transform' tab selected. In the 'Date' dropdown menu, the 'Age' option is highlighted with a green box. A tooltip for 'Age' is displayed, stating: 'Return the duration between the current local time and the values in the selected columns.' Below the dropdown, there are several other options: Month, Quarter, Week, Day, Combine Data and Time, Earliest, and Latest.

Result:

Date
-28.00:00:00
373.00:00:00
1156.00:00:00
814.00:00:00
539.00:00:00

Date Only: Extract the date part form the selected date/time columns.

Select “DateTime” column and click on drop down menu of “Date” option, then select “Date Only” option.

The screenshot shows the Power BI 'Transform' ribbon with the 'Date' dropdown menu open. The 'Date Only' option is highlighted with a green border. A tooltip explains: 'Extract the date component from the Date/Time values in the selected columns.'

Result:

The table 'Merged' contains the following data:

	Merged
1	11-04-2022
2	08-03-2022
3	01-04-2022
4	11-07-2019
5	08-09-2018
6	10-08-2021

Parse: Returns a date value parsed from the text in the selected columns. This is not important.

Year:

Year: It will extract the year part from the date/time values in the selected columns

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Year” option.

The screenshot shows the Power BI 'Transform' ribbon with the 'Date' dropdown menu open. The 'Year' option is highlighted with a green border. A tooltip explains: 'Extract the year component from the Date/Time values in the selected columns.'

Result:

The table 'Merged' contains the following data:

	Merged
1	2022
2	2022
3	2022
4	2019
5	2018
6	2021

Start of Year: It will extract first date of the year corresponding to each date/time value in the selected columns

The screenshot shows the Power BI interface with the 'Transform' tab selected. In the 'Queries [5]' pane, 'Merged' is selected. In the 'Merged' table preview, the 'Date' column contains dates from 2017 to 2021. On the ribbon, under the 'Date' dropdown, 'Start of Year' is highlighted. The 'Properties' pane on the right shows the 'Name' as 'DateTime Table' and the description as 'Return the first day of the year corresponding to each Date/Time value in the selected columns.'

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Start of Year” option.

Result:

Merged
01-01-2022
01-01-2022
01-01-2022
01-01-2019
01-01-2018
01-01-2021

- **End of Year:** It will extract last date of the year corresponding to each date/time value in the selected columns

The screenshot shows the Power BI interface with the 'Transform' tab selected. In the 'Queries [5]' pane, 'Merged' is selected. In the 'Merged' table preview, the 'Date' column contains dates from 2017 to 2021. On the ribbon, under the 'Date' dropdown, 'End of Year' is highlighted. The 'Properties' pane on the right shows the 'Name' as 'DateTime Table' and the description as 'Return the last day of the year corresponding to each Date/Time value in the selected columns.'

Select “Date/Time” column and click on drop down menu of “Date” option, then select “End of Year” option.

Result:

Merged
31-12-2022
31-12-2022
31-12-2022
31-12-2019
31-12-2018
31-12-2021

Month:

- **Month:** Extract the month part corresponding to each value in the selected date/time column.

The screenshot shows the Power BI desktop interface with the 'Transform' tab selected. In the center, there's a table named 'Merged' with several rows of date/time data. On the right, the 'Date' section of the Power BI ribbon is open, showing a dropdown menu for 'Month'. Other options like 'Year', 'Quarter', 'Week', 'Day', 'Start of Month', 'End of Month', 'Days in Month', and 'Name of Month' are also visible. A tooltip for 'Month' indicates it extracts the month component from the date/time values in the selected column.

Select “**Date/Time**” column and click on drop down menu of “**Date**” option, then select “**Month**” option.

Result:

The screenshot shows the 'Merged' table with the following data:

	Merged
1	11-04-2022
2	08-09-2022
3	01-04-2022
4	11-07-2022
5	09-09-2018
6	10-08-2021

-Start of Month: Returns the first day of the month corresponding to each date/time values in the selected column.

Select “**Date/Time**” column and click on drop down menu of “**Date**” option, then select “**Start of Month**” option.

The screenshot shows the Power BI desktop interface with the 'Transform' tab selected. In the center, there's a table named 'Merged' with several rows of date/time data. On the right, the 'Date' section of the Power BI ribbon is open, showing a dropdown menu for 'Start of Month'. Other options like 'Month', 'Year', 'Quarter', 'Week', 'Day', 'End of Month', 'Days in Month', and 'Name of Month' are also visible. A tooltip for 'Start of Month' indicates it returns the first day of the month corresponding to each Date/Time value in the selected column.

Result:

The screenshot shows the 'Merged' table with the following data:

	Merged
1	01-04-2022
2	01-03-2022
3	01-04-2022
4	01-07-2019
5	01-09-2018
6	01-08-2021

- End of Month:** Returns the last day of the month corresponding to each date/time values in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “End of Month” option.

Result:

Merged
30-04-2022
31-03-2022
30-04-2022
31-07-2019
30-09-2018
31-08-2021

- **Days in Month:** Returns the number of days in the month corresponding to each date/time values in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Days in Month” option.

Result:

Merged
30
31
30
31
30
31

- **Name of Month:** Returns the name of the month that corresponding to each date/time value in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Name of Month” option.

The screenshot shows the Power BI Data Editor interface. A context menu is open over a column named 'Merged'. The 'Date' dropdown menu is expanded, with 'Month' highlighted. Other options like 'Year', 'Quarter', 'Week', 'Day', and 'Name of Month' are also visible. The 'Name of Month' option is also highlighted with a green box. The 'APPLIED STEPS' pane on the right shows the step 'Calculated End of Month'.

Result:

Merged
April
March
April
July
September
August

Quarter:

- **Quarter of Year:** Returns the quarter corresponding to each date/time values in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Quarter of Year” option.

The screenshot shows the Power BI Data Editor interface. A context menu is open over a column named 'Merged'. The 'Date' dropdown menu is expanded, with 'Quarter of Year' highlighted. Other options like 'Year', 'Month', 'Quarter', 'Week', 'Day', and 'End of Quarter' are also visible. The 'End of Quarter' option is also highlighted with a green box. The 'APPLIED STEPS' pane on the right shows the step 'Calculated End of Month'.

Result:

Merged
2
1
2
3
3
3

- **Start of Quarter:** Returns the start day of quarter corresponding to each date/time values in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Start of Quarter” option.

The screenshot shows the Power BI desktop interface with the 'Transform' tab selected. In the center, there's a table named 'Merged'. On the right, the 'Date' column's dropdown menu is open, with 'End of Quarter' highlighted. A tooltip explains that it returns the start of the quarter corresponding to each Date/Time value.

Result:

Merged
01-04-2022
01-01-2022
01-04-2022
01-07-2019
01-07-2018
01-07-2021

- **End of Quarter:** Returns the end day of quarter corresponding to each date/time values in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “End of Quarter” option.

This screenshot is similar to the one above, showing the 'Transform' tab and the 'Merged' table. The 'Date' column's dropdown menu is open, and 'End of Quarter' is selected. A tooltip provides the same explanation as before.

Result:

Merged
30-06-2022
31-03-2022
30-06-2022
30-09-2019
30-09-2018
30-09-2021

Week:

- **Week of Year:** Returns the week of the year corresponding to each date/time value in the selected column.

Select “**Date/Time**” column and click on drop down menu of “**Date**” option, then select “**Week of Year**” option

The screenshot shows the Power BI interface in the 'Transform' tab. A dropdown menu under the 'Date' button is open, with 'Week' highlighted. The 'Week of Year' option is also visible in the list. The 'Merged' table in the main area contains several date/time values. The 'Properties' pane on the right shows the 'Name' as 'DateTime Table'.

Result:

The screenshot shows the 'Merged' table with a new column labeled 'Week of Year'. The values in this column are 16, 11, 14, 28, 36, and 33, corresponding to the dates in the original table.

Note: There are total 52 weeks in a year. If we use “**Week of Year**” option then it will calculate the no of weeks corresponding to the value we have in the selected column.

- **Week of Month:** Returns the week of the month corresponding to each date/time value in the selected column.

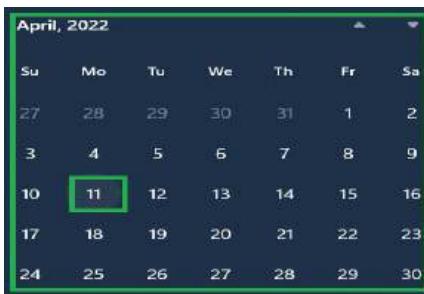
Select “**Date/Time**” column and click on drop down menu of “**Date**” option, then select “**Week of Month**” option

The screenshot shows the Power BI interface in the 'Transform' tab. A dropdown menu under the 'Date' button is open, with 'Week of Month' highlighted. The 'Week' option is also visible in the list. The 'Merged' table in the main area contains several date/time values. The 'Properties' pane on the right shows the 'Name' as 'DateTime Table'.

Result:

	1 ² Merged	
3		
2		
1		
2		
2		
2		

Ex) 11-04-2022 → Here this date comes in the 3rd week of that month. See below screen shot.



- **Start of Week:** Returns the start date of the week corresponding to each date/time value in the selected column.

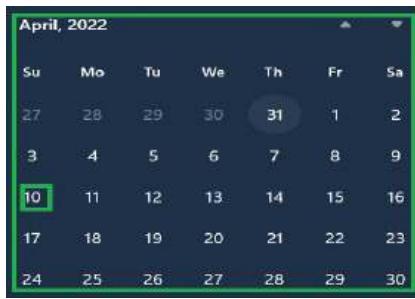
Select “Date/Time” column and click on drop down menu of “Date” option, then select “Start of Week” option

The screenshot shows the Power BI 'Transform Data' editor. In the 'Applied Steps' pane, a step is selected with the description: 'Return the start of the week corresponding to each Date/Time value in the selected column.' The 'Date' dropdown in the step configuration is set to 'Start of Week'. The 'Merged' column in the preview pane now displays the start date for each row, such as 10-04-2022, 06-03-2022, etc.

Result:

	1 ² Merged	
	10-04-2022	
	06-03-2022	
	27-03-2022	
	07-07-2019	
	02-09-2018	
	08-08-2021	

Ex) 11-04-2022 → Start of the week for this date is 10-04-2022. See the below screenshot.



- **End of Week**: Returns the end date of the week corresponding to the each date/time value in the selected column.

Select “**Date/Time**” column and click on drop down menu of “**Date**” option, then select “**End of Week**” option

The screenshot shows the Power BI Data Editor interface. On the left, there's a navigation pane with 'Queries [57]' and various table options like 'Time Table', 'Date Table', and 'DateTimeTable'. The 'DateTimeTable' is currently selected. In the main area, there's a table named 'Merged' with six rows containing dates: 11-04-2022, 08-04-2022, 05-04-2022, 12-07-2019, 08-09-2018, and 10-08-2021. On the right, the 'Transform' ribbon tab is active. A dropdown menu is open under the 'Date' button, with 'Week' selected. Below it, 'End of Week' is also highlighted with a green box. A tooltip at the bottom right of the dropdown menu states: 'Return the end of the week corresponding to each Date/Time value in the selected column.'

Result:

Merged
16-04-2022
12-03-2022
02-04-2022
13-07-2019
08-09-2018
14-08-2021

Day:

- **Day**: Extract the day part from the date/time values in the selected column

Select “**Date/Time**” column and click on drop down menu of “**Date**” option, then select “**Day**” option

This screenshot is similar to the previous one, showing the Power BI Data Editor with the 'DateTimeTable' selected. The 'Merged' table now contains only the day component of the dates: 16, 12, 02, 13, 08, and 14. The 'Transform' ribbon tab is still active, and the Date dropdown menu is open, with 'Day' selected. A tooltip at the bottom right of the dropdown menu states: 'Extract the day component from the Date/Time values in the selected columns.'

Result:

123 Merged
11
8
1
11
8
10

- **Day of Week:** Returns the day of the week corresponding to each date/time value in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Day of Week” option

The screenshot shows the Power BI interface with the 'Transform Data' tab selected. In the 'Applied Steps' pane on the right, a step named 'Day of Week' is highlighted. The description for this step is: 'Return the day of the week corresponding to the Date/Time values in the selected columns.' Below the description, there are options for 'Day', 'Day of Year', 'Start of Day', 'End of Day', and 'Name of Day'. The 'Day' option is currently selected.

Result:

123 Merged
1
2
5
4
6
2

Ex) 11-04-2022 → this date is the starting day in that week. See below screenshot.



Note:

Here 11-04-2022 → 1(Index start from “0” in that week. That means it will give 10-04-2022→0)

- **Day of Year:** Returns the day of the year corresponding to each date/time value in the selected column.

Select “**Date/Time**” column and click on drop down menu of “**Date**” option, then select “**Day of Year**” option

The screenshot shows the Power BI Data Editor interface. The 'Transform' tab is active at the top. In the center, there's a table named 'Merged' with one column containing dates. To the right, the 'Date' dropdown menu is open, and the 'Day of Year' option is selected. A tooltip for 'Day of Year' is visible, stating: 'Returns the day of the year corresponding to each Date/Time value in the selected column.' The 'APPLIED STEPS' pane on the right shows the step 'Extracted Date'.

Result:

123 Merged
101
67
91
192
251
222

Ex) 11-04-2022→101

Note: In a year we have only 365 days. For leap year it is 366 days. Out of this number “Day of Year” option will return day number of that particular year.

- **Start of Day:** Returns the start of the day corresponding to each date/time value in the selected column.

Select “**Date/Time**” column and click on drop down menu of “**Date**” option, then select “**Start of Day**” option

Ex) 11-04-2022 01:20:30

For the above example “Start of the Day” option will return result as below.

Result: 11-04-2022 00:00:00.

Note: Actually every day starts at “**00:00:00**”. So if you want to find start of day you have to take “**Date Time**” column. Then click on “**Start of Day**” option. It will return result as current date with “**00:00:00**” timing.

The screenshot shows the Power BI Data Editor interface with the 'Transform' tab selected. On the left, there's a sidebar with various data source icons. In the center, a table named 'Merged' is displayed with several rows of date/time data. On the right, a context menu is open over the 'Date' column, specifically for the first row. The menu has sections for 'Date', 'Time', 'Duration', and 'Number'. Under the 'Date' section, 'Day' is selected. Below it, a sub-menu for 'Current Date and Time' includes 'End of Day', which is also highlighted with a green box. The 'Applied Steps' pane on the right shows the step 'Changed Type'.

Result:

Merged	
	11-04-2022 00:00:00
	08-03-2022 00:00:00
	01-04-2022 00:00:00
	11-07-2019 00:00:00
	08-09-2018 00:00:00
	10-08-2021 00:00:00
	10-08-2021 00:00:00

End of Day: Returns the end of the day corresponding to each date/time value in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “End of Day” option

Ex) 11-04-2022 01:20:30

For the above example “End of the Day” option will return result as below.

Result: 12-04-2022 00:00:00.

Note: Actually every day starts at “00:00:00” and ends at “00:00:00”. End of the current day is nothing but start of the next day. It will return result as next day date with “00:00:00” timing.

This screenshot is identical to the one above, showing the Power BI Data Editor with the 'Transform' tab selected. The 'Merged' table is shown with its data. The context menu for the 'Date' column is open again, with 'End of Day' highlighted. The 'Applied Steps' pane shows the step 'Changed Type'.

Result:

Merged
12-04-2022 00:00:00
09-03-2022 00:00:00
02-04-2022 00:00:00
12-07-2019 00:00:00
09-09-2018 00:00:00
11-08-2021 00:00:00
11-08-2021 00:00:00

- **Name of Day:** Returns the end of the day corresponding to each date/time value in the selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Name of” option

The screenshot shows the Power BI interface with the 'Transform' tab selected. In the 'Applied Steps' pane, a step named 'Name of Day' is highlighted. The tooltip for this step states: 'Return the name of the day corresponding to each Date/Time value in the selected column.' The 'Date' dropdown in the 'Data' section is set to 'Day'.

Return:

Merged
Monday
Tuesday
Friday
Thursday
Saturday
Tuesday

Combine Date and Time: Merge the selected columns into a new column containing both date and time data from the selected columns.

Select both “Date” column and “Time” column and click on drop down of “Date” option in the “Transform”. Then click on “Combine Date and Time” option. Both columns will be combined and shown as a single column.

The screenshot shows the Power BI interface with the 'Transform' tab selected. In the 'Applied Steps' pane, a step named 'Combine Date and Time' is highlighted. The tooltip for this step states: 'Merge the selected columns into a new column containing both Date and Time data from the selected columns.' The 'Date' dropdown in the 'Data' section is set to 'Day'.

Result:

ABC	Merged
1	11-04-2022 01:20:30
2	08-03-2022 04:15:22
3	01-04-2022 05:22:38
4	11-07-2019 07:11:40
5	08-09-2018 09:18:20
6	10-08-2021 02:19:40

Earliest: Returns the earliest date value in the currently selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Earliest” option.

Queries [57]

- Time Table
- Date Table
- DateTime Table**
- Financials
- ROUNDING
- Statistics
- Standard Deviation
- Merge Columns
- EMP

= Table.TransformColumns(#"Merged Date and Time", {("Merged", DateTime.Date, type date)})

Earliest

Result:

Queries [57]

- Time Table
- Date Table
- DateTime Table**

= List.Min(#"Extracted Date"[Merged])

08-09-2018

Latest: Returns the latest date value in the currently selected column.

Select “Date/Time” column and click on drop down menu of “Date” option, then select “Latest” option.

Queries [57]

- Time Table
- Date Table
- DateTime Table**
- Financials
- ROUNDING
- Statistics
- Standard Deviation
- Merge Columns
- EMP
- Numeric
- Temp List

= Table.TransformColumns(#"Merged Date and Time", {("Merged", DateTime.Date, type date)})

Latest

Result:

Queries [57]

- Time Table
- Date Table
- DateTime Table**

= List.Max(#"Extracted Date"[Merged])

11-04-2022

- **Time**: It will extract time values part or format of time values from the time column. It accepts either a time column or date time column.

In this section, we have lot of options.

- Time Only
- Local Time
- Parse
- Hour
 - Hour
 - Start of Hour
 - End of Hour
- Minute
- Second
- Combine Date and Time
- Earliest
- Latest

Time Only: Extract the time part from the date/time column values in the selected column

The screenshot shows the Power BI desktop interface with the 'Transform' ribbon tab selected. In the 'Data' section of the ribbon, there is a dropdown menu with several options: 'Date', 'Time', 'Duration', 'Rounding', 'Trigonometry', 'Number', and 'Time Only'. The 'Time Only' option is highlighted with a green box. A tooltip for 'Time Only' is displayed, stating: 'Extract the time component from the Date/Time values in the selected column.' Below the ribbon, the 'Queries [5]' pane shows a 'DateTimeTable' query. The 'Merged' table in the main canvas contains the following data:

	Merged
1	11-04-2022 01:20:30
2	08-08-2022 04:15:22
3	01-04-2022 05:22:38
4	11-07-2019 07:11:40
5	08-09-2019 09:18:20
6	20-08-2021 02:19:40
7	10-08-2022 01:10:15

Result:

The screenshot shows the 'Merged' table in the Power BI canvas. The table contains the following time values:

Merged
01:20:30
04:15:22
05:22:38
07:11:40
09:18:20
02:19:40
01:10:15

Local Time: It will change all date/time/zone values in the selected columns to local time. This is not required much.

Parse: Return a time value parsed from the text in the selected column. This is not required much.

Hour:

- **Hour**: Returns the hour part corresponding to each date/time values in the selected columns.

The screenshot shows the Power BI Data Editor interface. The ribbon is at the top with 'Transform' selected. The 'Queries [57]' pane on the left lists several tables, with 'Datetime Table' currently selected. The main area shows a table named 'Merged' with 7 rows of data. The 'Time' column contains dates like '11-04-2022 01:28:30'. A context menu is open over the 'Time' column, with 'Start of Hour' highlighted. The 'Properties' pane on the right shows the function details: 'Name: Start of Hour', 'Description: Returns the hour corresponding to each Date/Time value in the selected columns.', and 'Applied Steps' showing 'Source' and 'Changed Type'.

Result:

123 Merged	
1	
4	
5	
7	
9	
2	
1	

- **Start of Hour**: Returns the start of the hour corresponding to each time value in the selected columns.

This screenshot is identical to the one above, showing the Power BI Data Editor with the 'Start of Hour' function applied to the 'Time' column of the 'Merged' table. The results show the start of the hour for each date, resulting in values like '01:00:00' for April 11, 2022.

Result:

123 Merged	
	11-04-2022 01:00:00
	08-03-2022 04:00:00
	01-04-2022 05:00:00
	11-07-2019 07:00:00
	08-09-2018 09:00:00
	10-08-2021 02:00:00
	10-08-2021 01:00:00

- **End of Hour**: Returns the end of the hour corresponding to each time value in the selected columns.

Minute: Returns the minute value corresponding to each date/time value in the currently selected column.

Result:

Merged
11-04-2022 02:00:00
08-03-2022 05:00:00
01-04-2022 06:00:00
11-07-2019 08:00:00
08-09-2018 10:00:00
10-08-2021 03:00:00
10-08-2021 02:00:00

Minute: Returns the minute value corresponding to each date/time value in the currently selected column.

Second: Returns the seconds corresponding to each Date/Time value in the selected columns.

Result:

Merged
20
15
22
11
18
19
10

Second: Returns the second value corresponding to each date/time value in the currently selected column.

Second: Returns the second corresponding to each Date-Time value in the selected columns.

Result:

L2 Merged
30
22
38
40
20
40
15

Combine Date and Time: It is already discussed in “Date” option in the transform data.

Earliest: Return the earliest time value in the currently selected column.

The screenshot shows the Power BI 'Transform' tab selected. In the formula bar, the query is defined as `= Table.TransformColumnTypes(Source,{{"Date", type time}})`. The 'EARLIER' function is highlighted in the formula bar. The 'EARLIER' function properties pane is open, showing the description: "Returns the earliest time value in the currently selected column." The 'APPLIED STEPS' section indicates the source is 'Changed Type' and the type is 'Merged Type'.

Result:

Queries [57]				
Time Table			01:20:30	
Date Table				

Latest: the latest time value in the currently selected column.

The screenshot shows the Power BI 'Transform' tab selected. In the formula bar, the query is defined as `= Table.TransformColumnTypes(Source,{{"Date", type time}})`. The 'LATEST' function is highlighted in the formula bar. The 'LATEST' function properties pane is open, showing the description: "Returns the latest time value in the currently selected column." The 'APPLIED STEPS' section indicates the source is 'Changed Type' and the type is 'Merged Type'.

Result:

Queries [57]				
Time Table			11:11:40	
Date Table				

❖ Add Column Tab:

- **Column from Examples:** Whatever the column we have chosen from the table, based on that column data type new column will be generated.

It will show you the output values options directly by the Power BI. We have to select the required outputs from the list of output values.

Def: Use examples to create new column in the selected table. If you select date column

Note: Power BI shows the default options for each column differently based on the data types of it.

If you take date column as example then date and time related default values will be visible; from there itself we have to select values and create new column.

In this section we have two options.

- **From all Columns**
- **From Selection**

From all Columns: New column will be derived based on all the columns. We can choose options of all the columns existing in the table and derive a new column.

From Selection: Based on the selected columns new column will be derived. In this way, options will be strict to the selected columns only.

For example, I am taking dates related data set. Select data set, select column and click on drop down menu of “**Column form Examples**” option in “**Add columns**” tab. Then click on “**From Selection**” option.

Date	Time
11-04-2022	01:20:30
08-03-2022	04:15:22
01-04-2022	05:22:38
11-07-2019	07:11:40
08-09-2018	09:18:20
10-08-2021	02:19:40
10-08-2021	01:10:15

Then again it will show you the pop up like below. Double click on “**first cell**” in “**column1**” at the right side of the screen.

Then you will see the list of default format options for date column. Select any one and press “Enter” button in the key board.

Same type of values will be visible for corresponding column values. Then click on “**OK**”. A new column will be created with the selected values.

Suppose I am selecting “Month Name from Date” option; then all the corresponding rows column values will be filled with corresponding month names. So Column name will be visible as “Month Name”.

The screenshot shows the 'Add Column From Examples' dialog. In the 'Column' dropdown, 'Month Name from Date' is selected. The preview pane shows a table with columns 'Date' and 'Time'. The 'Date' column contains dates like '11-04-2022', '08-03-2022', etc., and the 'Time' column contains times like '01:20:30', '04:15:22', etc. On the right, the 'Query Settings' pane lists various date-related functions, with 'MonthName' highlighted.

The screenshot shows the 'Add Column From Examples' dialog. In the 'Column' dropdown, 'Month Name' is selected. The preview pane shows a table with columns 'Date' and 'Time'. The 'Date' column contains dates like '11-04-2022', '08-03-2022', etc., and the 'Time' column contains times like '01:20:30', '04:15:22', etc. On the right, the 'Query Settings' pane lists various date-related functions, with 'MonthName' highlighted.

Result:

The screenshot shows the Power BI Data View. A new column 'Month Name' has been added to the table. The 'Month Name' column contains the following values: April, March, April, July, September, August, August. The entire table is highlighted with a green border.

If you want to create new column again based on the “Month Name” column; select the column click on “From Selection” option from the “Column form Examples” option.

Then it will show you the text related data type default options. Select the required option and click on “OK”. Again new column will be created. See below screenshots.

The screenshot shows the Power BI ribbon. The 'Add Column' tab is selected. Under the 'From Selection' button, a tooltip says: "Use examples and the current selection to create a new column in this table. (Ctrl+Shift+E)". Below the ribbon, the Power BI Data View shows the same table with the 'Month Name' column populated. The 'Month Name' column is highlighted with a green border.

Add Column From Examples
Enter sample values to create a new column (Ctrl+Enter to apply).

	Date	Time	Month Name
1	11-04-2022	01:20:30	April
2	08-03-2022	04:15:22	March
3	01-04-2022	05:22:38	April
4	11-07-2019	07:11:40	July
5	08-09-2018	09:18:20	September
6	10-08-2021	02:19:40	August
7	10-08-2021	01:10:15	August

Column1

April (Month Name)
5 (Length of Month Name)

OK Cancel

Add Column From Examples
Enter sample values to create a new column (Ctrl+Enter to apply).
Transform: Text.Length([Month Name])

	Date	Time	Month Name
1	11-04-2022	01:20:30	April
2	08-03-2022	04:15:22	March
3	01-04-2022	05:22:38	April
4	11-07-2019	07:11:40	July
5	08-09-2018	09:18:20	September
6	10-08-2021	02:19:40	August
7	10-08-2021	01:10:15	August

Length
5
5
5
4
9
6
6

OK Cancel

Result:

	Date	Time	Month Name	Length
1	11-04-2022	01:20:30	April	5
2	08-03-2022	04:15:22	March	5
3	01-04-2022	05:22:38	April	5
4	11-07-2019	07:11:40	July	4
5	08-09-2018	09:18:20	September	9
6	10-08-2021	02:19:40	August	6
7	10-08-2021	01:10:15	August	6

Note: If you want to combine columns by using “**Column form Examples**” that we can do here with “**From all Columns**” option.

Suppose I want to combine Month Name and Length columns using delimiter “_”.

Go to “**Column from Examples**” and select “**From all columns**” option. Then you can see the blow window.

Add Column From Examples
Enter sample values to create a new column (Ctrl+Enter to apply).

	Date	Time	Month Name	Length
1	11-04-2022	01:20:30	April	5
2	08-03-2022	04:15:22	March	5
3	01-04-2022	05:22:38	April	5
4	11-07-2019	07:11:40	July	4
5	08-09-2018	09:18:20	September	9
6	10-08-2021	02:19:40	August	6
7	10-08-2021	01:10:15	August	6

Column1

April_5

OK Cancel

Note: We should type first cell value of first column, and type delimiter “-“ then type second column first cell value manually. Then only combine will be happened.

Once value supplied click on “Enter” button in the keyboard and then click on “OK”.

	Date	Time	Month Name	Length	Merged
1	11-04-2022	01:20:30	April	5	April-5
2	08-03-2022	04:15:22	March	5	March-5
3	01-04-2022	05:22:38	April	5	April-5
4	11-07-2019	07:11:40	July	4	July-4
5	08-09-2018	09:18:20	September	9	September-9
6	10-08-2021	02:19:40	August	6	August-6
7	10-08-2021	01:10:15	August	6	August-6

Result:

	Date	Time	Month Name	Length	Merged
1	11-04-2022	01:20:30	April	5	April-5
2	08-03-2022	04:15:22	March	5	March-5
3	01-04-2022	05:22:38	April	5	April-5
4	11-07-2019	07:11:40	July	4	July-4
5	08-09-2018	09:18:20	September	9	September-9
6	10-08-2021	02:19:40	August	6	August-6
7	10-08-2021	01:10:15	August	6	August-6

- **Custom Column:** By using this option, we can create our own column based on requirement using custom formulas. We can display week no, month name, we can combine columns, year number, day number etc.

go to “Add column” tab, select the data set in “Power Query Editor” and Click on “Custom column”.

Custom Column
Add a column that is computed from the other columns.

New column name:

Custom column formula:

Available columns:

- Date
- Time
- Month Name
- Length
- Merged

Learn about Power Query formulas

✓ No syntax errors have been detected.

In this window we have to write formulas for generating new column.

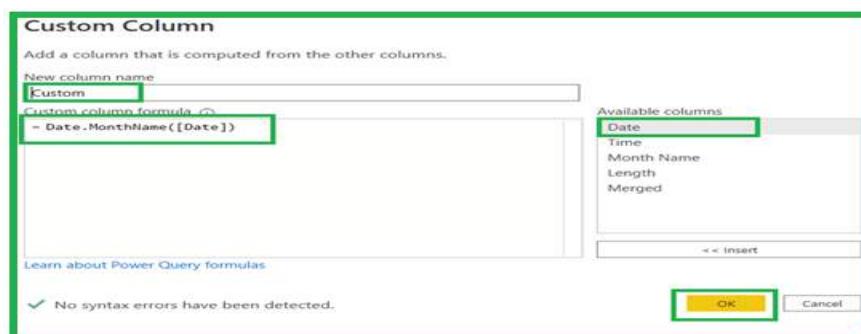
Suppose I want to display the “Month Name” of the particular date column, for that we have to write some formulas. All these formulas are M-language formulas.

If you want to see the M-language formulas you have to click on below link. There you can find lot of formulas.

Link: <https://docs.microsoft.com/en-us/powerquery-m/power-query-m-function-reference>

Or else click on “Learn about Power Query formulas” then choose “comprehensive function reference content set”

If I want to write date related function you have to write for example “**Date.MonthName**” to display month name. One function written, then open the bracket “(“, and double click on “**Month Name**” column, then close the bracket ”)”. Finally click on “OK”. Custom column will be created.



Result:

	Date	Time	Month Name	Length	Merged	Custom
1	11-04-2022	01:20:30	April	5	April-5	April
2	08-03-2022	04:15:22	March	5	March-5	March
3	01-04-2022	05:22:38	April	5	April-5	April
4	11-07-2019	07:11:40	July	4	July-4	July
5	08-09-2018	09:18:20	September	9	September-9	September
6	10-08-2021	02:19:40	August	6	August-6	August
7	10-08-2021	01:10:15	August	6	August-6	August

If you want to combine columns, you can do in the “**custom column**” option as well. Click on “**Custom Column**”; then write formula as below

If you want to combine “**Month Name**” and “**Custom**” column you should write the formula as “[Month Name]&”:&[Custom]”. Here “:” is called delimiter between two columns.

Custom Column

Add a column that is computed from the other columns.

New column name

Custom.1

Custom column formula ⓘ

= [Month Name]&":&[Custom]

Available columns

Date

Time

Month Name

Length

Merged

Custom

<< Insert

[Learn about Power Query formulas](#)

✓ No syntax errors have been detected.

OK

Cancel

Result:

Time	Month Name	Length	Merged	Custom	Custom.1
01:20:30	April		5 April-5	April	April:April
04:15:22	March		5 March-5	March	March:March
05:22:38	April		5 April-5	April	April:April
07:11:40	July		4 July-4	July	July:July
09:18:20	September		9 September-9	September	September:September
02:19:40	August		6 August-6	August	August:August
01:10:15	August		6 August-6	August	August:August

Note: Here if we want to combine multiple columns, all the columns character type of columns. We have combined both “**text**” data type columns. We can’t combine columns of “**Text**” and “**Numeric**”. If you try to combine them; it will though an error.

Custom Column

Add a column that is computed from the other columns.

New column name

Custom.2

Custom column formula ⓘ

= [Month Name]&":&[Length]

Available columns

Date

Time

Month Name

Length

Merged

Custom

Custom.1

<< Insert

[Learn about Power Query formulas](#)

✓ No syntax errors have been detected.

OK

Cancel

Result:

Month Name	Length	Merged	Custom	Custom.1	Custom.2
April	5	April-5	April	April:April	Error
March	5	March-5	March	March:March	Error
April	5	April-5	April	April:April	Error
July	4	July-4	July	July:July	Error
September	9	September-9	September	September:September	Error
August	6	August-6	August	August:August	Error
August	6	August-6	August	August:August	Error

Note: Whatever the options you have applied for creating a new column in “Add Column” tab, then “New column” will be created in the end of the table.

If you do this operation in other tabs, new column will be generated at the place of both columns”.

- **Conditional Column:** Based on condition, new column will be generated on the selected column.

If you want to create new column based on condition you have to select the data set, click on “Conditional Column”. Then it will show you below screen.

Add Conditional Column

Add a conditional column that is computed from the other columns or values.

New column name: Custom.2

Column Name	Operator	Value	Output
If Day Name	equals	ABC 123 Sunday	Then ABC 123 01
Else If Day Name	equals	ABC 123 Monday	Then ABC 123 02
Else If Day Name	equals	ABC 123 Tuesday	Then ABC 123 03
Else If Day Name	equals	ABC 123 Wednesday	Then ABC 123 04
Else If Day Name	equals	ABC 123 Thursday	Then ABC 123 05
Else If Day Name	equals	ABC 123 Friday	Then ABC 123 06
... [More]			
Add Clause			
Else		ABC 123 07	

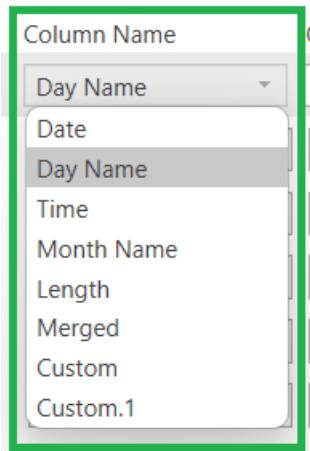
OK Cancel

In this you can write the “column name” and you can write conditions, you can use different operators. Here I have used “equals” operator, you can type “value” here to come as “output” or else you can pass by “parameter”.

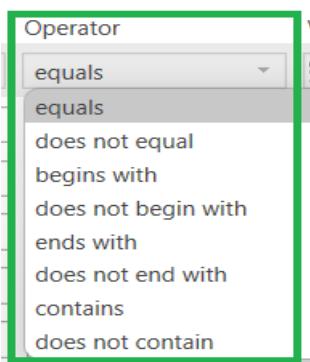
If you want to add new condition you have to click on “Add Clause” option.

Add Clause

You can change Column name by clicking on drop down menu.



You can change operator by click on drop down menu.



If you want to delete or move up or move down any condition then you have to click on dots. (...) on the right side of the condition and you can do the operation.

Add Conditional Column

Add a conditional column that is computed from the other columns or values.

New column name: Custom.2

	Column Name	Operator	Value	Output
If:	Day Name	equals	ABC = Sunday	ABC = 01
Else If:	Day Name	equals	ABC = Monday	ABC = 02
Else If:	Day Name	equals	ABC = Tuesday	ABC = 03
Else If:	Day Name	equals	ABC = Wednesday	ABC = 04
Else If:	Day Name	equals	ABC = Thursday	ABC = 05
Else If:	Day Name	equals	ABC = Friday	ABC = 06
Add Clause				
Else:		ABC =	07	

On the right side of the interface, there is a vertical toolbar with buttons for 'Delete', 'Move Up', and 'Move Down'.

Ex) suppose if you have "N" values then you have to write "N-1" "if-Else" conditions and for the "Nth Value" you can write the output directly in "Else" part.

I have written conditions for the entire week like

If "Day Name" = Sunday, then output should come as "01" and so on.....Else "07"

Result:

ABC_Merged	ABC_Custom	ABC_Custom.1	ABC_Custom.2
April-5	April	April:April	2
March-5	March	March:March	3
April-5	April	April:April	6
July-4	July	July:July	5
September-9	September	September:September	7
August-6	August	August:August	3
August-6	August	August:August	3

Index Column: By using this option, sequential values will be generated for each row in the table in a separate column. In this we have three options.

- From 0
- From 1
- Custom

From 0: If you select this option index values will be generated from 0 to the last record. Suppose if we have 10 records, index values will be 0 to 9.

Select the data set and click on “Index Column” then “From 0” then automatically index column will be created and index starts from “0”.

Result:

ABC_Merged	ABC_Custom	ABC_Custom.1	ABC_Custom.2	Index
April-5	April	April:April	2	0
March-5	March	March:March	3	1
April-5	April	April:April	6	2
July-4	July	July:July	5	3
September-9	September	September:September	7	4
August-6	August	August:August	3	5
August-6	August	August:August	3	6

From 1: If you select this option index values will be generated from 1 to the last record. Suppose if we have 10 records, index values will be 1 to 10.

Select the data set and click on “Index Column” then “From 1” then automatically index column will be created and index starts from “1”.

Result:

ABC_Merged	ABC_Custom	ABC_Custom.1	ABC_Custom.2	Index.1
April-5	April	April:April	2	0
March-5	March	March:March	3	1
April-5	April	April:April	6	2
July-4	July	July:July	5	3
September-9	September	September:September	7	4
August-6	August	August:August	3	5
August-6	August	August:August	3	6

Custom: If you select this option, index values will be generated based on the two arguments. One is “**Starting index**” and another one is “**increment**”

Select the data set and click on “**Index Column**” then “**Custom**” then new window will be opened. There you have to give “**starting index**” and “**increment**” values.

Ex) if you give “**starting index**” ad “2” then index values generated from 2. i.e starting value in index column will be 2 And if you give “**increment**” as 2; then the next value in index column will be $2+2=4$. See the below screenshot.

The screenshot shows the Power BI desktop interface. The 'File', 'Home', and 'Transform' tabs are visible at the top. The 'Add Column' tab is currently selected. In the 'Transform' ribbon, the 'Index Column' button is highlighted. A tooltip appears above the 'Index Column' button, stating: "Create a new column with an index starting at a specified value and with a specified increment." Below the ribbon, the 'Queries [57]' pane is open, showing various data sources like 'Transpose', 'Time Table', 'Date Table', 'DateTime Table' (which is selected), 'financials', 'ROUNDING', and 'Statistics'. The main workspace shows a table with columns 'Index' and 'Month-Year'. The 'Index' column has values 1 through 7, and the 'Month-Year' column has values corresponding to April, March, April, July, September, August, and August respectively. At the bottom, the 'Add Index Column' dialog box is displayed, prompting the user to 'Add an index column with a specified starting index and increment.' It contains two input fields: 'Starting Index' with the value '2' and 'Increment' with the value '2'. There are 'OK' and 'Cancel' buttons at the bottom right of the dialog.

Result:

The screenshot shows the Power BI data grid. A new column titled 'Index.2' has been added, containing the following index values: 2, 4, 6, 8, 10, 12, and 14. The column header 'Index.2' is highlighted with a green border.

- **Duplicate Column:** If you want to create a duplicate column for the selected column; you can do that with this option.

Select the “**Dataset**” and then select a “**column**” then click on “**Duplicate column**”. Automatically copy of that column will be created. It will be visible as last column in that data set.

Or else you can right click on select column, there you will see “**Duplicate Column**” option. You can click on that. Automatically duplicate column will be created.

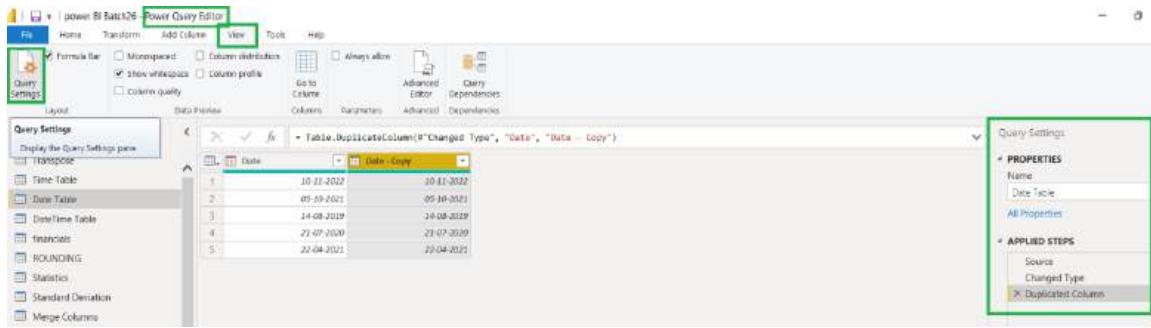
Result:

Date	Date - Copy
10-11-2022	10-11-2022
05-10-2021	05-10-2021
14-08-2019	14-08-2019
21-07-2020	21-07-2020
22-04-2021	22-04-2021

❖ View Tab:

- **Query Settings:** By using this option, you can make “**Query Settings**” pane visible or disable in the power query editor.
Query settings pane will be visible on the right side of the power query editor.

If you click on this option; “**Query Settings**” pane will be disabled. If you want it back, again you need to click on the same option.

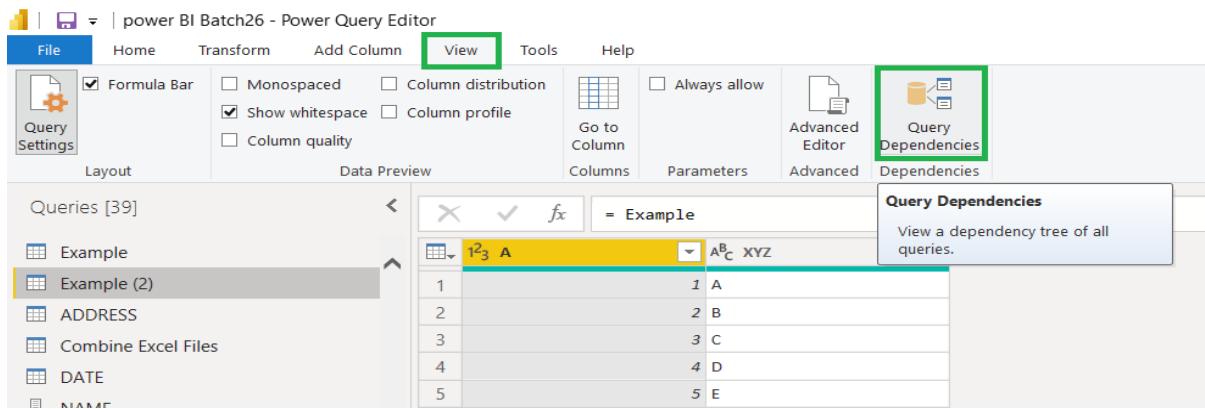


- **Formula Bar:** Some times; formula bar will be disabled unfortunately, to make it visible you have to do tick mark on “Formula Bar” check box in “View” tab. Then again formula bar will be appeared.

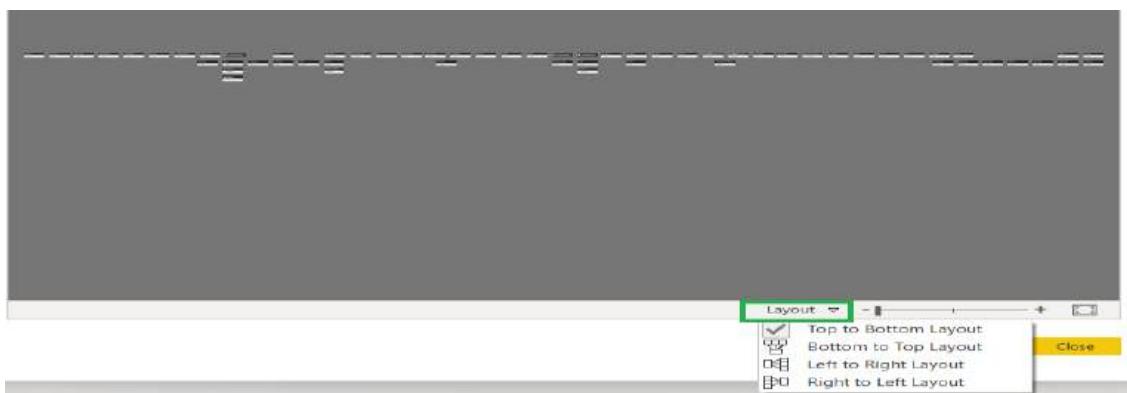
If you select “Formula Bar”, result will be look likes below.

- **Query Dependencies:** If you want to see the relationships between the data sets, then you have to use this option.

Go to “View” tab -> Click on “Query Dependencies”. Then you can see the relationship between parent table and child table.



Once click on “Query Dependencies”, then you can see the below screen. If you want you can zoom in or zoom out by dragging the pointer in the bottom of the screen. You can change the “Layout” for your convenience.



See the below screen shot to identify the reference data set.



- **Column Distribution:** It will show you the column value distribution in data preview.

Select the data set and click on “**Column Distribution**” option in “View” tab. Then it will show you how many distinct values are there, and how many unique values are there in each column on the top of the column values. See below.

Column distribution
Show column value distribution in data preview.

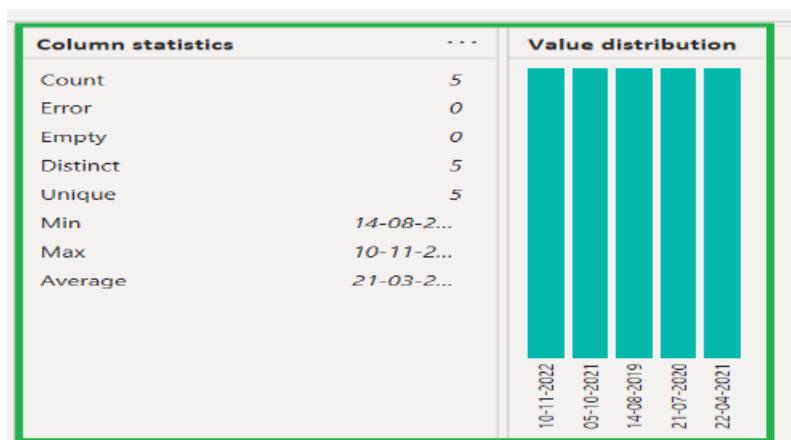
	Date - Copy
1	10-11-2022
2	05-10-2021
3	14-08-2019
4	21-07-2020
5	22-04-2021

➤ **Column Profile:** It will show you the column profile in details pane.

Select the data set and click on “**Column Profile**” option in “**View**” tab. Then you can see the profile details of the columns at the bottom of the table.

Column profile
`= Table.DuplicateColumn(#"Changed Type", "Date", "Date - Copy")`

	Date	Date - Copy
1	10-11-2022	10-11-2022
2	05-10-2021	05-10-2021
3	14-08-2019	14-08-2019
4	21-07-2020	21-07-2020
5	22-04-2021	22-04-2021



It will show you count, Error, Empty, and Distinct, Unique, Min, Max and average values. In Value Distribution; you can see the number of records present in the table.

➤ **Column Quality:** It will show you the column quality details in data preview.

Select the “Data set” and click on “Column Quality” option in “View” tab. Then you can see the column quality details of the columns in the data preview.

Date	Date - Copy
10-11-2022	10-11-2022
05-10-2021	05-10-2021
14-08-2019	14-08-2019
21-07-2020	21-07-2020
22-04-2021	22-04-2021

Result:

Column	Valid	Error	Empty
Date	100%	0%	0%
Date - Copy	100%	0%	0%

It will show you Number of valid records, errors, Empty records in percentage values.

DAX FUNCTIONS

DAX stands for “**Data Analytics Expressions**”. It is a collection of functions, operators, and constants that can be used in a formula, or expression, to calculate and return one or more values.

In transform data, we can forecast the output before running of a report, but if you want the output at the time of running a report, then we can use “**DAX Functions**”.

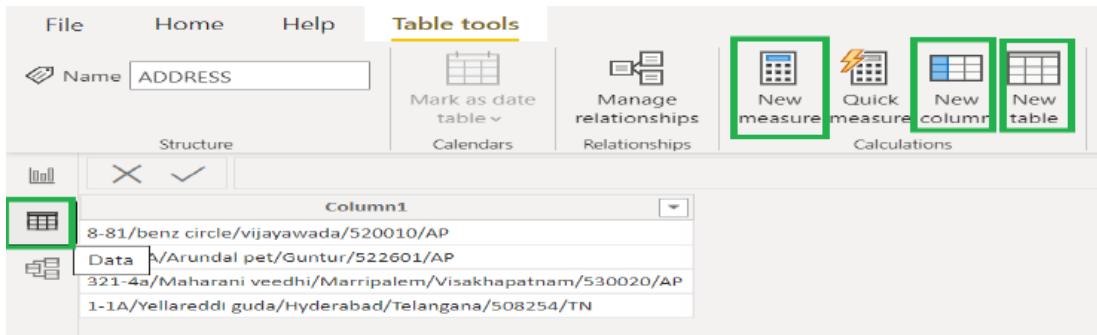
By using these functions we can generate calculated columns, measures and calculated measures. We have lot of functions to do this.

Click on the below link to see the different types of “**DAX Reference Functions**”

Link: <https://learn.microsoft.com/en-us/dax/>

These functions only will be applied in three options. Those are “**New Measure**”, “**New Column**” and “**New Table**”.

We can see these options in Power BI desktop at **Model**, and **Data Tab**, right click on “**Data Set**” and “**Modeling**” etc.



Wherever you see these options, there you can apply “**Dax Functions**”. No need to go for particular tab.

Dax functions will return multiple types of outputs i.e single value, or multiple values or entire table.

Note1: Whatever the dax functions returns single value as output, those functions will be applied on “**New Measure**” only.

Note2: If you want to perform dax function on row level calculations, those functions will be applied on “**New Column**” or “**Calculated Column**”.

Note3: If you want to perform dax function on table level calculations, those functions will be applied on “**New Table**”.

Functions in DAX:

Different types of DAX functions are as follows. Each of which has different functionality.

- **Aggregation functions**
- **Date and time functions**
- **Text functions**
- **Logical functions**
- **Math and Trig functions**
- **Other functions**
- **Relationship functions**
- **Table manipulation functions**
- **Statistical functions**

- Filter functions
- Time Intelligence Functions

Aggregation Functions

Aggregation functions calculate a value such as sum, average, minimum or maximum for all rows in a column or table as defined by the expression.

Note: All aggregation functions returns a single value.

- **To Create Measure:** If you want to create a measure, first you have to select particular “data set” and click on “New Measure”. Otherwise measure will be created in different data set.

The screenshot shows the Power BI Data Editor interface. The 'Table tools' ribbon is selected, and the 'New measure' button is highlighted with a green box. A data table is displayed with four columns: NUMBER, VALUE, DATE, and RATE. The Fields pane on the right shows a measure named 'AGGREGATION' with its details: Storage mode: Import, Last refresh: 9/14/2022, 7:17:44 PM. The measure is listed under the 'AGGREGATION' folder.

Once you select data set and click on “New Measure” then it will show you formula bar.

If you want to write a formula for average you have to just put the name of the measure before “=” symbol and then write “**Average(<column>)**”

Note: Here <column> means, on which column you want to create measure.

Note: Once formula entered, then click on “**tick mark**” or just press “**Enter**” button.

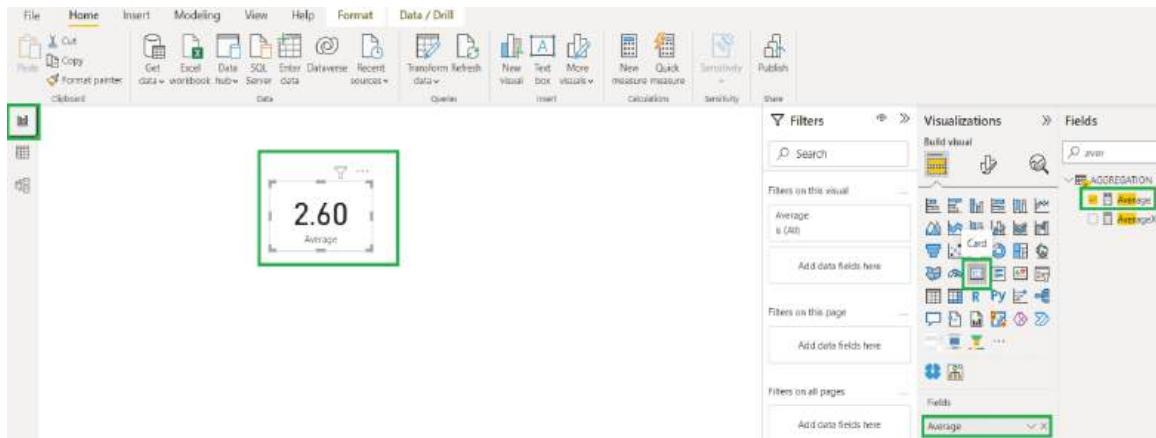
The screenshot shows the Power BI formula bar. The 'Measure tools' ribbon is selected, and the 'New measure' button is highlighted with a green box. The formula bar displays the formula 'AVERAGE([Column])'. Below the formula bar, a data table is shown with columns NUMBER, VALUE, DATE, and RATE. The Fields pane on the right shows the 'Average' measure under the 'AGGREGATION' folder.

Once measure created, then you have to see the result of that measure right.

To see the result of the measure:

Note1: If you want to see the measure value then it can be done in visual format only. For that you have to go to the “Report” tab and select “Data set” on the right side pane. Expand that data set there you can see “Measure”.

Note2: Take “Card” Visual from the default visuals and Drag the measure from the data set to the “fields” section under the visualizations pane.



❖ **AVERAGE:** Returns average of all the numbers in a column.

Syntax: [AVERAGE\(<column>\)](#)

Average = Sum of values/No of values.

Ex) Table Name: **Aggregation**

NUMBER	VALUE	DATE	RATE
2	1	11-05-2022	5
3	2	10-09-2021	2
1	4	25-02-2020	
2	4	12-04-1998	
5	2	08-07-2021	5

I want to calculate average on “NUMBER” column in the table “Aggregation”. So

I have to write average formula in “New Measure” is as follow.

Formula: Average = [AVERAGE\('Aggregation'\[NUMBER\]\)](#)

Output:

2.60

Average

Note: We can apply this function either in decimal value or numeric value

- ❖ **AVERAGEA:** Returns average of all the values in a column. Here it can access Boolean values also.

The **AverageA** function takes a column and averages the numbers in it, but also handles non-numeric data types according to the following rules:

- Values that evaluate to TRUE count as 1.
- Values that evaluate to FALSE count as 0 (zero).
- Values that contain non-numeric text count as 0 (zero).
- Empty text ("") counts as 0 (zero).

Example

The following example returns the average of non-blank cells in the referenced column, given the following table. If you used the AVERAGE function, the mean would be 21/2; with the AVERAGEA function, the result is 22/5.

Transaction ID	Amount	Result
0000123	1	Counts as 1
0000124	20	Counts as 20
0000125	n/a	Counts as 0
0000126		Counts as 0
0000126	TRUE	Counts as 1

DAX

- `AVERAGEA([Amount])`

- ❖ **AVERAGEX:** Calculates the average of a set of expressions evaluated over a table

Syntax: AVERAGEX(<table>,<expression>)

Formula: AverageX =
AVERAGEX('Aggregation','Aggregation'[NUMBER]*'Aggregation'[VALUE])

Output:

6.00

AverageX

Note: Here table should be mentioned in single quotes. For example, 'Aggregation'

Difference between Average and AverageX:

- ✓ Average function takes only single argument, where as AverageX takes two arguments.
 - ✓ Average function calculates result on the already existing column, but in AverageX it will consider result based on the second argument.
 - ✓ If you pass second argument in AverageX function as column name then Average and AverageX will work in same way.
- **COUNT:** Counts the number of rows in the specified column that contain non-blank values.

Syntax: `COUNT(<column>)`

Dax Function: `Count = count(Aggregation[VALUE])`

Output:



Note: It will support all the data types like numeric, decimal and text.

- **COUNTX:** Counts the number of rows that contain a non-blank value or an expression that evaluates to a non-blank value, when evaluating an expression over a table.

Syntax: `COUNTX(<table>,<expression>)`

Dax Function: `CountX = COUNTX(Aggregation,Aggregation[NUMBER]+Aggregation[VALUE])`

Output:



- **COUNTROWS:** The COUNTROWS function counts the number of rows in the specified table, or in a table defined by an expression.

Syntax: `COUNTROWS([<table>])`

Dax Function: `Countrows = COUNTROWS(Aggregation)`

Output:

5

Countrows

- **DISTINCTCOUNT:** Counts the number of distinct values in a column.

Syntax: `DISTINCTCOUNT(<column>)`

Dax Function: `DistinctCount = DISTINCTCOUNT(Aggregation[VALUE])`
Output:

3

DistinctCount

- **DISTINCTCOUNTNOBLANK:** Counts the number of distinct values in a column.

Syntax: `DISTINCTCOUNTNOBLANK(<column>)`

Dax Function: `DistinctCountNoBlank = DISTINCTCOUNTNOBLANK(Aggregation[RATE])`

Output:

3

DistinctCountNoBla...

Note: Distinct count functions counts the blank value. But count function doesn't consider blank values.

- **COUNTBLANK:** Counts the number of blank cells in a column.

Syntax: `COUNTBLANK(<column>)`

Dax Function: `CountBlank = COUNTBLANK(Aggregation[RATE])`

Output:

2

CountBlank

- **SUM:** Adds all the numbers in a column.

Syntax: `SUM(<column>)`

Dax Function: `Sum = sum(Aggregation[VALUE])`

Output:

13

Sum

- **SUMX:** Returns the sum of an expression evaluated for each row in a table.

Syntax: SUMX(<table>, <expression>)

Dax Function:

SumX = `sumx(Aggregation,Aggregation[VALUE]+Aggregation[NUMBER])`

Output:

26

SumX

- **MAX:** Returns the largest value in a column, or between two scalar expressions.

Syntax:

`MAX(<column>)`

`MAX(<expression1>, <expression2>)`

Dax Function: Max = `Max(Aggregation[NUMBER])`

Output:

5

Max

- **MAXX:** Evaluates an expression for each row of a table and returns the largest value.

Syntax: MAXX(<table>,<expression>)

Dax Function:

MaxX = `MAXX(Aggregation,Aggregation[VALUE]*Aggregation[NUMBER])`

Output:

10

MaxX

- **MIN**: Returns the smallest value in a column, or between two scalar expressions.

Syntax:

MIN(<column>)
MIN(<expression1>, <expression2>)

Dax Function:

MIN = **MIN**(Aggregation[VALUE])

Output:



- **MINX**: Returns the smallest value that results from evaluating an expression for each row of a table.

Syntax: **MINX**(<table>, < expression>)

Dax Function:

MinX = **minx**(Aggregation,Aggregation[VALUE]*Aggregation[NUMBER])

Output:

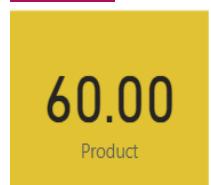


- **PRODUCT**: Returns the product of the numbers in a column.

Syntax: **PRODUCT**(<column>)

Dax Function: Product = **PRODUCT**('Aggregation'[NUMBER])

Output:



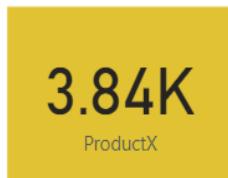
- **PRODUCTX**: Returns the product of an expression evaluated for each row in a table.

Syntax: **PRODUCTX**(<table>, <expression>)

Dax Function:

```
ProductX =  
PRODUCTX('Aggregation','Aggregation'[NUMBER]*'Aggregation'[VALUE])
```

Output:

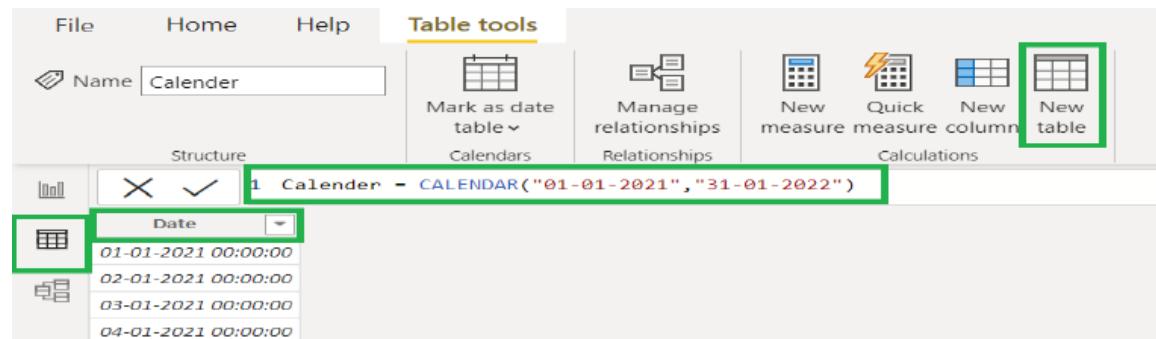


Date and Time Functions

These functions help you to create calculations based on dates and time.

- **CALENDAR**: To generate all the dates between “Start date” and “End date” we can use calendar function. It will generate a new calculated data set with single column named “Date”

Note: This function will be applied on “New Table” option only, because it will generate a new calculated table.



Syntax: CALENDAR(<start_date>, <end_date>)

Dax Function:

```
Calendar1 = CALENDAR("2020-01-01","2022-12-31")
```

Note: Here date values and character values should be written in double quotes only.

Output:

1 Calendar1 = CALENDAR("2020-01-01","2022-12-31")

Date

- Sort ascending
- Sort descending
- Clear sort
- Clear filter
- Clear all filters
- Date filters

- (Select all)
- 01-01-2020 00:00:00
- 02-01-2020 00:00:00
- 03-01-2020 00:00:00
- 04-01-2020 00:00:00
- 05-01-2020 00:00:00
- 06-01-2020 00:00:00
- 07-01-2020 00:00:00
- 08-01-2020 00:00:00
- 09-01-2020 00:00:00
- 10-01-2020 00:00:00
- 11-01-2020 00:00:00
- 12-01-2020 00:00:00
- 13-01-2020 00:00:00
- 14-01-2020 00:00:00
- 15-01-2020 00:00:00
- 16-01-2020 00:00:00
- 17-01-2020 00:00:00
- 18-01-2020 00:00:00
- 19-01-2020 00:00:00
- 20-01-2020 00:00:00
- 21-01-2020 00:00:00
- 22-01-2020 00:00:00
- 23-01-2020 00:00:00
- 24-01-2020 00:00:00

If you want to see all the dates in between the start date and end date then you have to go to “Data” tab and click on “Date”.

But if you want some more dates then go to “Report” tab and click on “Table” visual and drag the “Date” column in the “Columns” option.

Filters

Visualizations

Fields

calendar1

Calendar

Date

Year

Quarter

Month

Day

Columns

Date

Year

Quarter

Month

Day

Remove field

New quick measure

Show items with no data

Date

Date is Date

More options

Date

31-12-2022 00:00:00

30-12-2022 00:00:00

29-12-2022 00:00:00

28-12-2022 00:00:00

27-12-2022 00:00:00

26-12-2022 00:00:00

25-12-2022 00:00:00

24-12-2022 00:00:00

23-12-2022 00:00:00

22-12-2022 00:00:00

21-12-2022 00:00:00

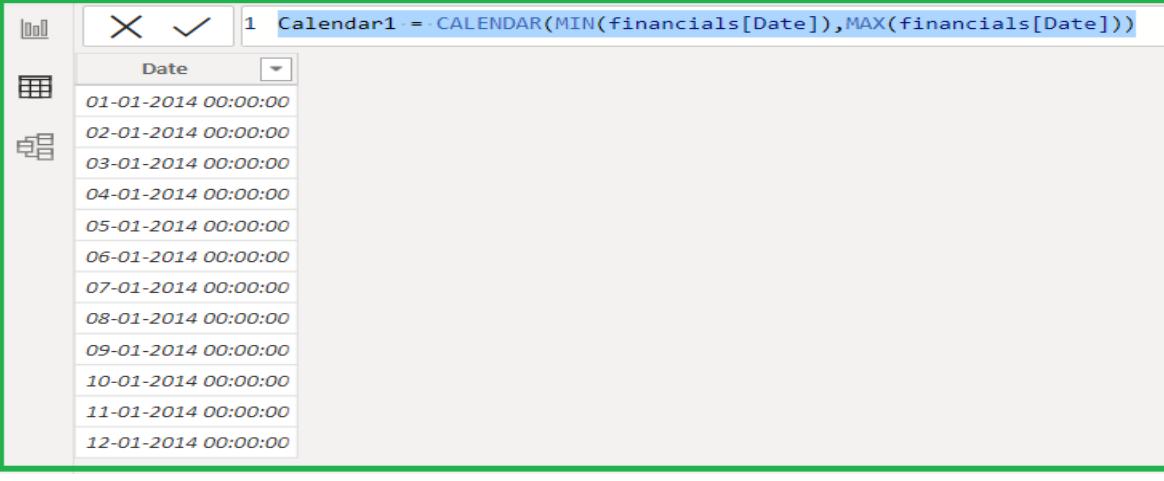
20-12-2022 00:00:00

Note: You can give start date and end date either in static way or dynamic way. Suppose, I want to generate Calendar based on “**Date**” column in “**Financials**” data set.

For start date I will take “**MIN**” value of that column and for end date I will take “**MAX**” value of that column.

Dax Function:

Calendar1 = CALENDAR(MIN(financials[Date]),MAX(financials[Date]))



Date
01-01-2014 00:00:00
02-01-2014 00:00:00
03-01-2014 00:00:00
04-01-2014 00:00:00
05-01-2014 00:00:00
06-01-2014 00:00:00
07-01-2014 00:00:00
08-01-2014 00:00:00
09-01-2014 00:00:00
10-01-2014 00:00:00
11-01-2014 00:00:00
12-01-2014 00:00:00

Rule: Start date must be less than or equal to end date but should not be greater than end date.

Note: You can take start date and end date from different data sets in the calendar function.

- **DATE:** Returns the specified date in “**datetime**” format. It will take three numeric values and returns a column with “**datetime**” format

Syntax: DATE(<year>, <month>, <day>)

Dax Function:

Date2 = Date(2022,10,18)

The screenshot shows the Power BI interface with the 'Column tools' tab selected. A new column named 'Date2' has been created, and its formula is displayed as '= Date(2022,10,18)'. The data preview shows 12 rows of data, each with the same date value: 18-10-2022 00:00:00.

Note: Here you have to give proper year, month and day values in the date function. Otherwise it will give default values

You can pass the arguments in Date function in a dynamic way as well. See the below syntax. Here I have taken **year**, **month** and **day** values from **date** column in Calendar data set.

Dax Function:

```
Date1 =
date(year('Calendar'[Date]),month('Calendar'[Date]),day('Calendar'[Date]))
```

Note: All other functions except calendar will be applied in “**New Column**” only.

By default “**Date**” function returns date value along with default time with **datetime** format.

If you want to change the data type you can change to other. See below.

The screenshot shows the Power BI interface with the 'Column tools' tab selected. The 'Data type' dropdown is open, showing various options: Whole number, Decimal number, Fixed decimal number, Date/time, Date, Time, Text, True/false, Binary, and others. The 'Date' option is highlighted with a green circle.

Note: We can change the data type for “**Calculated Column**” but we can’t change the data type for “**Measure**”

If you want you can change the format of the “Date” column. See below screenshot how we can change the format.

The screenshot shows the Power BI Column Tools pane. The 'Format' button is highlighted. A dropdown menu for 'Date formats' is open, listing various date format options. The option '14-03-2001 (dd-mm-yyyy)' is selected and highlighted with a green box. The table structure pane on the left shows a table with columns 'Date' and 'Date2'. The 'Date2' column has the formula `Date(2022,10,18)`.

Date	Date2
01-01-2014 00:00:00	18 October 2022
02-01-2014 00:00:00	18 October 2022
03-01-2014 00:00:00	18 October 2022
04-01-2014 00:00:00	18 October 2022
05-01-2014 00:00:00	18 October 2022
06-01-2014 00:00:00	18 October 2022
07-01-2014 00:00:00	18 October 2022
08-01-2014 00:00:00	18 October 2022
09-01-2014 00:00:00	18 October 2022
10-01-2014 00:00:00	18 October 2022
11-01-2014 00:00:00	18 October 2022
12-01-2014 00:00:00	18 October 2022

- **DATEDIFF:** Returns the difference between two dates based on interval boundaries such as Second, Minute, Hour, Day, Week, Month, Quarter and Year.

Syntax: `DATEDIFF(<Date1>, <Date2>, <Interval>)`

Dax Function:

`DATEDIFF('2018-04-01','2022-12-31',MONTH)`

Ex)

The screenshot shows the Power BI Data View pane. A calculated column named 'DateDiff1' is defined with the formula `Datediff("2018-04-01","2022-12-31",MONTH)`. The table structure pane on the left shows a table with columns 'Date', 'Date2', 'DateDiff', and 'DateDiff1'. The 'DateDiff1' column contains the value 56 for all rows.

Date	Date2	DateDiff	DateDiff1
01-01-2014	18-10-2022	3212	56
02-01-2014	18-10-2022	3211	56
03-01-2014	18-10-2022	3210	56
04-01-2014	18-10-2022	3209	56
05-01-2014	18-10-2022	3208	56
06-01-2014	18-10-2022	3207	56
07-01-2014	18-10-2022	3206	56
08-01-2014	18-10-2022	3205	56
09-01-2014	18-10-2022	3204	56
10-01-2014	18-10-2022	3203	56
11-01-2014	18-10-2022	3202	56
12-01-2014	18-10-2022	3201	56

Note: In this function we can give larger value as “**start date**” and “**smaller value**” as end date as well. But results will come along with negative sign “-“.

Date	Date2	DateDiff	DateDiff1
01-01-2014	18-10-2022	3212	-56
02-01-2014	18-10-2022	3211	-56
03-01-2014	18-10-2022	3210	-56
04-01-2014	18-10-2022	3209	-56
05-01-2014	18-10-2022	3208	-56
06-01-2014	18-10-2022	3207	-56
07-01-2014	18-10-2022	3206	-56
08-01-2014	18-10-2022	3205	-56
09-01-2014	18-10-2022	3204	-56
10-01-2014	18-10-2022	3203	-56
11-01-2014	18-10-2022	3202	-56
12-01-2014	18-10-2022	3201	-56

We can pass the column values as arguments in the “DATEDIFF” function dynamically.

Dax Function:

DateDiff = **DATEDIFF(Calendar1[Date],Calendar1[Date2],DAY)**

Note: Here we can calculate difference between two dates based on the 3rd argument. If you give “Day” as 3rd argument difference will be coming in days. If you give “Month” as 3rd argument difference will be coming in months etc.

Name			Column	DATEDIFF(Date1, Date2, Interval)	
Data type			Whole number	Returns the number of units (unit specified in Interval) between the input dates.	
Structure			Date	Date2	Column
	X	✓	1 DateDiff=DATEDIFF(Calendar1[Date],Calendar1[Date2],		
			Date	Date2	Column
			01-01-2014	18-10-2022	
			02-01-2014	18-10-2022	
			03-01-2014	18-10-2022	
			04-01-2014	18-10-2022	
			05-01-2014	18-10-2022	
			06-01-2014	18-10-2022	
			07-01-2014	18-10-2022	
			08-01-2014	18-10-2022	
			09-01-2014	18-10-2022	
			10-01-2014	18-10-2022	
			11-01-2014	18-10-2022	
			12-01-2014	18-10-2022	
Structure			Formatting		
X	✓	1 DateDiff = DATEDIFF(Calendar1[Date],Calendar1[Date2],DAY)	Date	Date2	DateDiff
			01-01-2014	18-10-2022	3212
			02-01-2014	18-10-2022	3211
			03-01-2014	18-10-2022	3210
			04-01-2014	18-10-2022	3209
			05-01-2014	18-10-2022	3208
			06-01-2014	18-10-2022	3207
			07-01-2014	18-10-2022	3206
			08-01-2014	18-10-2022	3205
			09-01-2014	18-10-2022	3204
			10-01-2014	18-10-2022	3203
			11-01-2014	18-10-2022	3202
			12-01-2014	18-10-2022	3201

- **DATEVALUE:** Converts a date in text format to a date in “datetime” format.

Syntax: DATEVALUE(date_text)

Dax Function:

DateValue1 = DATEVALUE(Calendar1[Date2])

Ex)

Date	Date2	Datediff	DateDiff1	DateValue1
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00

Date	Date2	Datediff	DateDiff1	DateValue1
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00

Note: We have lot of options to convert date in text data type to “**datetime**” data type by simply click on “**Data type**” option and change. But in sometimes this function is required to use in multiple functions within other functions.

- **DAY:** Returns the day of the month, a number from 1 to 31. That means it will extract day part from the date column.

Syntax: DAY(<date>)

Note: Here “**date**” in “**datetime**” format, or a text representation of date.

Dax Function:

Day = Day(Calendar1[Date])

Ex)

Date	Date2	Datediff	DateDiff1	DateValue1	Day
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00	1
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00	2
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00	3
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00	4
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00	5
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00	6
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00	7
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00	8
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00	9
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00	10
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00	11
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00	12

- **MONTH:** Returns the month as a number from 1 (January) to 12 (December). That means it will extract month part from the date column.

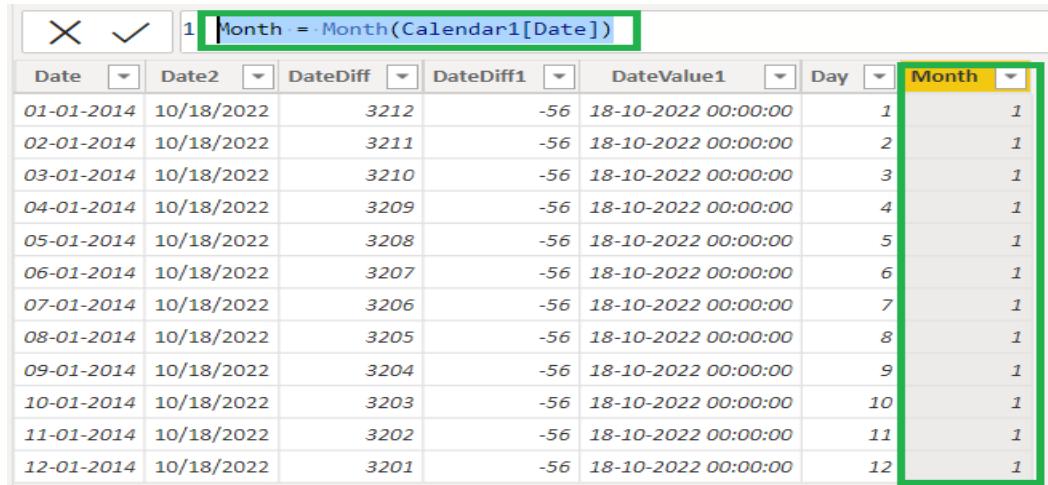
Syntax: MONTH(<datetime>)

Note: Here “**datetime**” column in “**datetime**” format, or a text format.

Dax Function:

Month = Month(Calendar1[Date])

Ex)



Date	Date2	DateDiff	DateDiff1	DateValue1	Day	Month
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00	1	1
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00	2	1
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00	3	1
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00	4	1
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00	5	1
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00	6	1
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00	7	1
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00	8	1
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00	9	1
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00	10	1
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00	11	1
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00	12	1

- **YEAR:** Returns the year of a date as a four digit integer in the range 1900-9999. That means it will extract year part from the **datetime** column.

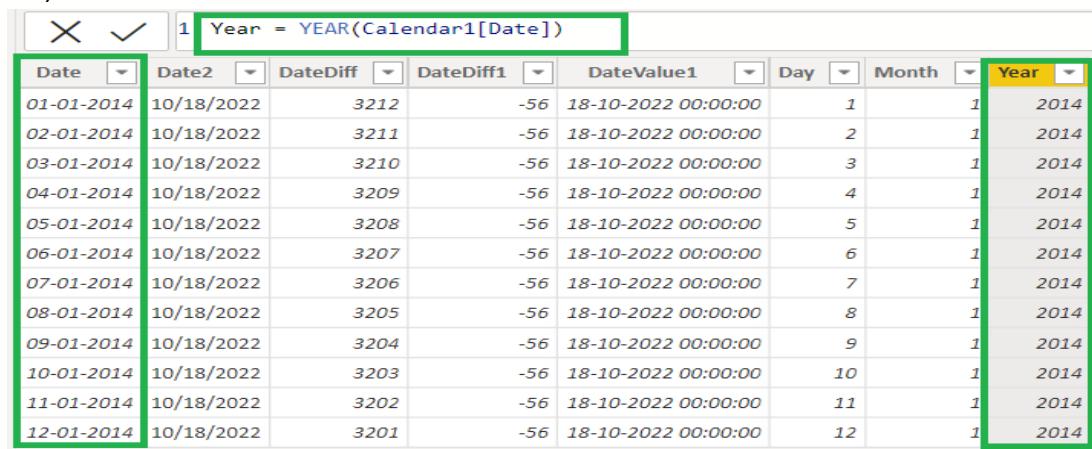
Syntax: YEAR(<date>)

Note: Here “**date**” column in “**datetime**” format, or a text format.

Dax Function:

Year = YEAR(Calendar1[Date])

Ex)



Date	Date2	DateDiff	DateDiff1	DateValue1	Day	Month	Year
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00	1	1	2014
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00	2	1	2014
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00	3	1	2014
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00	4	1	2014
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00	5	1	2014
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00	6	1	2014
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00	7	1	2014
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00	8	1	2014
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00	9	1	2014
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00	10	1	2014
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00	11	1	2014
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00	12	1	2014

- **EDATE:** Returns the date that is the indicated number of months before or after the start date. That means it will add the number of months to the existing date and display that date.

Syntax: EDATE(<start_date>, <months>)

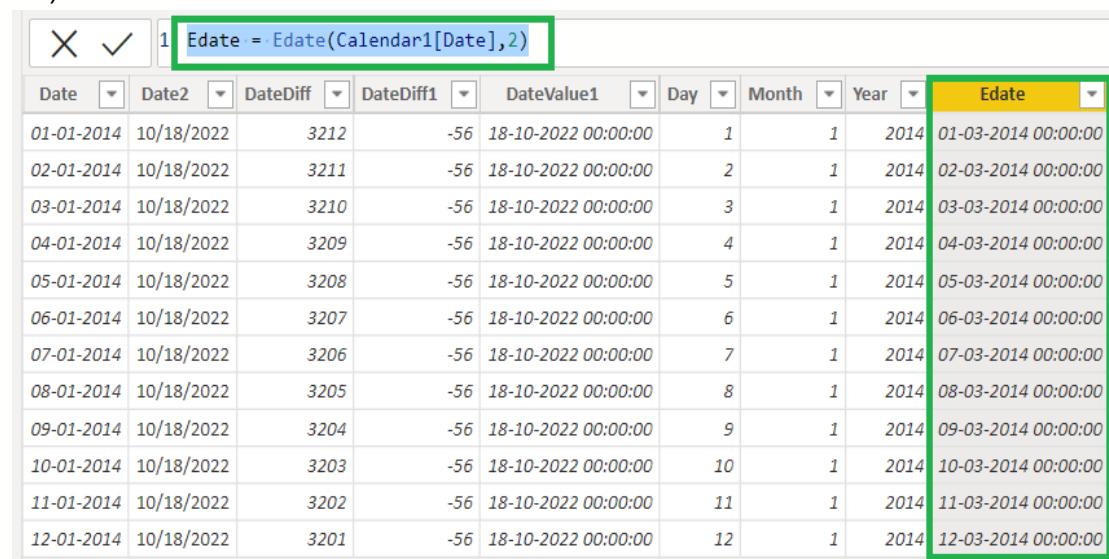
Note1: Here “**start_date**” in “**datetime**” format, or a text representation of date. “**months**” column is an integer that represents number of months before or after “**start_date**” column.

Note2: You can give either negative or positive values in the <months> argument.

Dax Function:

Edate = Edate(Calendar1[Date],2)

Ex)



Date	Date2	DateDiff	DateDiff1	DateValue1	Day	Month	Year	Edate
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00	1	1	2014	01-03-2014 00:00:00
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00	2	1	2014	02-03-2014 00:00:00
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00	3	1	2014	03-03-2014 00:00:00
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00	4	1	2014	04-03-2014 00:00:00
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00	5	1	2014	05-03-2014 00:00:00
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00	6	1	2014	06-03-2014 00:00:00
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00	7	1	2014	07-03-2014 00:00:00
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00	8	1	2014	08-03-2014 00:00:00
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00	9	1	2014	09-03-2014 00:00:00
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00	10	1	2014	10-03-2014 00:00:00
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00	11	1	2014	11-03-2014 00:00:00
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00	12	1	2014	12-03-2014 00:00:00

- **EOMONTH:** Returns the date in **datetime** format of the last day of the month, before or after a specified number of months.

Syntax: EOMONTH(<start_date>, <months>)

Note1: Here “**start_date**” in “**datetime**” format, or a text representation of date. “**months**” column is an integer that represents number of months before or after “**start_date**” column.

Note2: Here before means negative values, after means positive values.

Dax Function:

Eomonth = EOMONTH(Calendar1[Date],-2)

Ex)

1 Eomonth = EOMONTH(Calendar1[Date], -2)												
Date	Date2	DateDiff	DateDiff1	DateValue1	Day	Month	Year	Edate		Eomonth		
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00	1	1	2014	01-03-2014 00:00:00		30-11-2013 00:00:00		
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00	2	1	2014	02-03-2014 00:00:00		30-11-2013 00:00:00		
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00	3	1	2014	03-03-2014 00:00:00		30-11-2013 00:00:00		
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00	4	1	2014	04-03-2014 00:00:00		30-11-2013 00:00:00		
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00	5	1	2014	05-03-2014 00:00:00		30-11-2013 00:00:00		
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00	6	1	2014	06-03-2014 00:00:00		30-11-2013 00:00:00		
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00	7	1	2014	07-03-2014 00:00:00		30-11-2013 00:00:00		
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00	8	1	2014	08-03-2014 00:00:00		30-11-2013 00:00:00		
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00	9	1	2014	09-03-2014 00:00:00		30-11-2013 00:00:00		
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00	10	1	2014	10-03-2014 00:00:00		30-11-2013 00:00:00		
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00	11	1	2014	11-03-2014 00:00:00		30-11-2013 00:00:00		
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00	12	1	2014	12-03-2014 00:00:00		30-11-2013 00:00:00		

- **NOW:** Returns the current date and time in **datetime** format.

Syntax: NOW()

Note: The **NOW** function is useful when you need to display the current date and time on a worksheet or calculate a value based on the current date and time, and have that value updated each time you open the worksheet.

Dax Function:

now = now()

Ex)

1 now = now()	
Date	now
31-01-2014	19-10-2022 06:50:38
25-03-2015	19-10-2022 06:50:38
18-04-2016	19-10-2022 06:50:38
21-05-2017	19-10-2022 06:50:38
23-06-2018	19-10-2022 06:50:38
15-07-2019	19-10-2022 06:50:38
11-08-2020	19-10-2022 06:50:38
08-09-2021	19-10-2022 06:50:38
19-10-2022	19-10-2022 06:50:38

- **TODAY:** Returns the current date.

Syntax: TODAY()

Note: Here today function will display date along with default time.

Dax Function:

Today = TODAY()

Ex)

Date	now	Today
31-01-2014	19-10-2022 06:57:09	19-10-2022 00:00:00
25-03-2015	19-10-2022 06:57:09	19-10-2022 00:00:00
18-04-2016	19-10-2022 06:57:09	19-10-2022 00:00:00
21-05-2017	19-10-2022 06:57:09	19-10-2022 00:00:00
23-06-2018	19-10-2022 06:57:09	19-10-2022 00:00:00
15-07-2019	19-10-2022 06:57:09	19-10-2022 00:00:00
11-08-2020	19-10-2022 06:57:09	19-10-2022 00:00:00
08-09-2021	19-10-2022 06:57:09	19-10-2022 00:00:00
19-10-2022	19-10-2022 06:57:09	19-10-2022 00:00:00

- **HOUR:** Returns the hour as a number from 0 (12:00 A.M.) to 23 (11:00 P.M.).

Syntax: HOUR(<datetime>)

Note: Here datetime is **datetime** value, such as 16:48:00 or 4:48 PM.

Dax Function:

Hour = Hour(Calendar1[Now])

Ex)

Date	Date2	DateDiff	DateDiff1	DateValue1	Day	Month	Year	Edate	Emonth	Now	Today	Hour
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00	1	2014	01-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00	2	2014	02-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00	3	2014	03-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00	4	2014	04-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00	5	2014	05-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00	6	2014	06-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00	7	2014	07-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00	8	2014	08-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00	9	2014	09-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00	10	2014	10-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00	11	2014	11-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00	12	2014	12-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00		7

- **MINUTE:** Returns the minute as a number from 0 to 59, given a date and time value.

Syntax: MINUTE(<datetime>)

Note: Here datetime is **datetime** value, such as 16:48:00 or 4:48 PM.

Dax Function:

Minute = MINUTE(Calendar1[Now])

Ex)

Date	Date2	DateDiff	DateDiff1	DateValue1	Day	Month	Year	Edate	Emonth	Now	Today	Hour	Minute
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00	1	1	2014	01-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00	2	1	2014	02-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00	3	1	2014	03-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00	4	1	2014	04-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00	5	1	2014	05-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00	6	1	2014	06-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00	7	1	2014	07-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00	8	1	2014	08-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00	9	1	2014	09-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00	10	1	2014	10-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00	11	1	2014	11-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00	12	1	2014	12-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5

- **SECOND:** Returns the seconds of a time value, as a number from 0 to 59.

Syntax: SECOND(<time>)

Note: Here “time” column in “datetime” value, such as 16:48:23 or 4:48:47 PM.

Dax Function:

Second = SECOND(Calendar1[Now])

Ex)

Date	Date2	DateDiff	DateDiff1	DateValue1	Day	Month	Year	Edate	Emonth	Now	Today	Hour	Minute	Second
01-01-2014	10/18/2022	3212	-56	18-10-2022 00:00:00	1	1	2014	01-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
02-01-2014	10/18/2022	3211	-56	18-10-2022 00:00:00	2	1	2014	02-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
03-01-2014	10/18/2022	3210	-56	18-10-2022 00:00:00	3	1	2014	03-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
04-01-2014	10/18/2022	3209	-56	18-10-2022 00:00:00	4	1	2014	04-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
05-01-2014	10/18/2022	3208	-56	18-10-2022 00:00:00	5	1	2014	05-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
06-01-2014	10/18/2022	3207	-56	18-10-2022 00:00:00	6	1	2014	06-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
07-01-2014	10/18/2022	3206	-56	18-10-2022 00:00:00	7	1	2014	07-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
08-01-2014	10/18/2022	3205	-56	18-10-2022 00:00:00	8	1	2014	08-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
09-01-2014	10/18/2022	3204	-56	18-10-2022 00:00:00	9	1	2014	09-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
10-01-2014	10/18/2022	3203	-56	18-10-2022 00:00:00	10	1	2014	10-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
11-01-2014	10/18/2022	3202	-56	18-10-2022 00:00:00	11	1	2014	11-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18
12-01-2014	10/18/2022	3201	-56	18-10-2022 00:00:00	12	1	2014	12-03-2014 00:00:00	30-11-2013 00:00:00	19-10-2022 07:05:18	19-10-2022 00:00:00	7	5	18

- **TIME:** Converts hours, minutes, and seconds given as numbers to a time in datetime format.

Syntax: TIME(hour, minute, second)

Dax Function:

Time = Time(11,15,25)

Note: If you give like this, time will be displayed along with default date. It will be in datetime format. If you want you can change it to time format by selecting in data type option.

Ex)

Note: You can pass the HOUR, MINUTE, SECOND functions dynamically as arguments in the TIME function if you want.

- **NETWORKDAYS:** Returns the number of whole workdays between two dates (inclusive).

Syntax: NETWORKDAYS(<start_date>, <end_date>[, <weekend>, <holidays>])

Dax Function:

Networkday = NETWORKDAYS(Calendar1[Day],Calendar1[Now],1)

Date	Day	Month	Year	DayName	MonthName	YearName	Now	Today	Hour	Minute	Second	Networkday
01-01-2014	1	1	2014	Monday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32039
02-01-2014	2	1	2014	Tuesday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32039
03-01-2014	3	1	2014	Wednesday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32037
04-01-2014	4	1	2014	Thursday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32036
05-01-2014	5	1	2014	Friday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32035
06-01-2014	6	1	2014	Saturday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32034
07-01-2014	7	1	2014	Sunday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32034
08-01-2014	8	1	2014	Monday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32037
09-01-2014	9	1	2014	Tuesday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32038
10-01-2014	10	1	2014	Wednesday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32032
11-01-2014	11	1	2014	Thursday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32033
12-01-2014	12	1	2014	Friday	January	2014	19-10-2022 00:00:00	19-10-2022 00:00:00	8	7	36	32030

That means it will ignore the weekend and holidays and display all the working days.

Note:

Weekend number values indicate the following weekend days:

- 1 or omitted: Saturday, Sunday
- 2: Sunday, Monday
- 3: Monday, Tuesday
- 4: Tuesday, Wednesday
- 5: Wednesday, Thursday
- 6: Thursday, Friday
- 7: Friday, Saturday
- 11: Sunday only

- 12: Monday only
- 13: Tuesday only
- 14: Wednesday only
- 15: Thursday only
- 16: Friday only
- 17: Saturday only

- **QUARTER**: Returns the quarter as a number from 1 (January – March) to 4 (October – December).

Every year has 4 Quarters right. Based on the date you have, corresponding quarter will be displayed.

Syntax: QUARTER(<date>)

Dax Function:

Quarter = QUARTER(Random_Date[Date])

Ex)

Structure		Formatting	
X	✓	1	Quarter = QUARTER(Random_Date[Date])
Date	now	Today	Quarter
31-01-2014	19-10-2022 08:12:11	19-10-2022 00:00:00	1
25-03-2015	19-10-2022 08:12:11	19-10-2022 00:00:00	1
18-04-2016	19-10-2022 08:12:11	19-10-2022 00:00:00	2
21-05-2017	19-10-2022 08:12:11	19-10-2022 00:00:00	2
23-06-2018	19-10-2022 08:12:11	19-10-2022 00:00:00	2
15-07-2019	19-10-2022 08:12:11	19-10-2022 00:00:00	3
11-08-2020	19-10-2022 08:12:11	19-10-2022 00:00:00	3
08-09-2021	19-10-2022 08:12:11	19-10-2022 00:00:00	3
19-10-2022	19-10-2022 08:12:11	19-10-2022 00:00:00	4

- **TIMEVALUE**: Converts a time in **text** format to a time in **datetime** format.

Syntax: TIMEVALUE(time_text)

Dax Function:

TimeValue = TIMEVALUE('Time Table'[Time])

Ex)

Structure		Formatting	
Name	Time	\$ Format	Text
Data type	Text	\$ %	Auto
Date	Time	TimeValue	
01:20:30	11:15:25 AM	30-12-1899 11:15:25	
04:15:22	11:15:25 AM	30-12-1899 11:15:25	
05:22:38	11:15:25 AM	30-12-1899 11:15:25	
07:11:40	11:15:25 AM	30-12-1899 11:15:25	
09:18:20	11:15:25 AM	30-12-1899 11:15:25	
11:11:40	11:15:25 AM	30-12-1899 11:15:25	
02:19:40	11:15:25 AM	30-12-1899 11:15:25	

Structure		Formatting	
Name	TimeValue	\$ Format	*14-03-2001 13:30:...
Data type	Date/time	\$ %	Auto
Date	time	TimeValue	
01:20:30	11:15:25 AM	30-12-1899 11:15:25	
04:15:22	11:15:25 AM	30-12-1899 11:15:25	
05:22:38	11:15:25 AM	30-12-1899 11:15:25	
07:11:40	11:15:25 AM	30-12-1899 11:15:25	
09:18:20	11:15:25 AM	30-12-1899 11:15:25	
11:11:40	11:15:25 AM	30-12-1899 11:15:25	
02:19:40	11:15:25 AM	30-12-1899 11:15:25	

Note: You can change the datetime format to time format in Data type option in column tools

- **WEEKDAY:** Returns a number from 1 to 7 identifying the day of the week of a date. By default the day ranges from 1 (Sunday) to 7 (Saturday).

Syntax: `WEEKDAY(<date>, <return_type>)`

Dax Function:

`WeekDay = WEEKDAY(Randam_Date[Date],1)`

Ex)

Date	now	Today	WeekDay
31-01-2014	19-10-2022 08:29:08	19-10-2022 00:00:00	1
25-03-2015	19-10-2022 08:29:08	19-10-2022 00:00:00	2
18-04-2016	19-10-2022 08:29:08	19-10-2022 00:00:00	2
21-05-2017	19-10-2022 08:29:08	19-10-2022 00:00:00	2
23-06-2018	19-10-2022 08:29:08	19-10-2022 00:00:00	2
15-07-2019	19-10-2022 08:29:08	19-10-2022 00:00:00	3
11-08-2020	19-10-2022 08:29:08	19-10-2022 00:00:00	3
08-09-2021	19-10-2022 08:29:08	19-10-2022 00:00:00	3
19-10-2022	19-10-2022 08:29:08	19-10-2022 00:00:00	4

Date	now	Today	Quarter	WeekDay
31-01-2014	19-10-2022 08:30:52	19-10-2022 00:00:00	1	6
25-03-2015	19-10-2022 08:30:52	19-10-2022 00:00:00	1	4
18-04-2016	19-10-2022 08:30:52	19-10-2022 00:00:00	2	2
21-05-2017	19-10-2022 08:30:52	19-10-2022 00:00:00	2	1
23-06-2018	19-10-2022 08:30:52	19-10-2022 00:00:00	2	7
15-07-2019	19-10-2022 08:30:52	19-10-2022 00:00:00	3	2
11-08-2020	19-10-2022 08:30:52	19-10-2022 00:00:00	3	3
08-09-2021	19-10-2022 08:30:52	19-10-2022 00:00:00	3	4
19-10-2022	19-10-2022 08:30:52	19-10-2022 00:00:00	4	4

Note:

return_type indicates as below.

1, week begins on Sunday (1) and ends on Saturday (7).

2, week begins on Monday (1) and ends on Sunday (7).

3, week begins on Monday (0) and ends on Sunday (6).

- **WEEKNUM:** Returns the week number for the given date according to the **return_type** value.

Out of 52 weeks, return the week number of the corresponding date.

Syntax: `WEEKNUM(<date>[, <return_type>])`

Dax Function:

WeekNum = WEEKNUM(Randam_Date[Date],1)

Ex)

The screenshot shows the Power BI Data View interface. A new column named 'WeekNum' is being created. The formula bar at the top contains the formula 'WeekNum=WEEKNUM(Randam_Date[Date],1)'. The 'Properties' pane on the right shows the data type as 'Whole number' and the summary as 'Sum'. The table below displays 10 rows of data, each with a Date column value and a corresponding WeekNum value.

Date	now	Today	WeekNum
31-01-2014	19-10-2022 08:40:07	19-10-2022 00:00:00	1
25-03-2015	19-10-2022 08:40:07	19-10-2022 00:00:00	1
18-04-2016	19-10-2022 08:40:07	19-10-2022 00:00:00	2
21-05-2017	19-10-2022 08:40:07	19-10-2022 00:00:00	2
23-06-2018	19-10-2022 08:40:07	19-10-2022 00:00:00	2
15-07-2019	19-10-2022 08:40:07	19-10-2022 00:00:00	3
11-08-2020	19-10-2022 08:40:07	19-10-2022 00:00:00	3
08-09-2021	19-10-2022 08:40:07	19-10-2022 00:00:00	3
19-10-2022	19-10-2022 08:40:07	19-10-2022 00:00:00	4

The screenshot shows the Power BI Data View interface after the 'WeekNum' column has been created. The 'WeekNum' column now contains the values 5, 13, 17, 21, 25, 29, 33, 37, and 43 respectively for each row. The formula bar at the top still shows the original formula.

Date	now	Today	Quarter	WeekDay	WeekNum
31-01-2014	19-10-2022 08:42:15	19-10-2022 00:00:00	1	6	5
25-03-2015	19-10-2022 08:42:15	19-10-2022 00:00:00	1	4	13
18-04-2016	19-10-2022 08:42:15	19-10-2022 00:00:00	2	2	17
21-05-2017	19-10-2022 08:42:15	19-10-2022 00:00:00	2	1	21
23-06-2018	19-10-2022 08:42:15	19-10-2022 00:00:00	2	7	25
15-07-2019	19-10-2022 08:42:15	19-10-2022 00:00:00	3	2	29
11-08-2020	19-10-2022 08:42:15	19-10-2022 00:00:00	3	3	33
08-09-2021	19-10-2022 08:42:15	19-10-2022 00:00:00	3	4	37
19-10-2022	19-10-2022 08:42:15	19-10-2022 00:00:00	4	4	43

Text Functions

These functions basically applied on character string columns. We have lot of functions available in this category. These text functions will be applied in “New Column” option only.

- **COMBINEVALUES:** Joins two or more text strings into one text string or combining multiple columns into single column by using **delimiter** or **special character**.

Syntax:

COMBINEVALUES(<delimiter>, <expression>[, <expression>]...)

Dax Function:

CombineValues = COMBINEVALUES("|",'Text Data'[String1],'Text Data'[String2])

Here delimiter "|" have to give in double quotes, because it is special character.

Ex)

Structure			Formatting	Properties
X	✓	1	CombineValues = COMBINEVALUES(" ", 'Text Data'[String1], 'Text Data'[String2])	
String1	String2	String3	CombineValues	
AA	aaa	111	AA aaa	
BB	bbb	222	BB bbb	
CC	ccc	333	CC ccc	
DD	ddd	444	DD ddd	
EE	eee	555	EE eee	
FF	fff	666	FF fff	
GG	ggg	777	GG ggg	
HH	hhh	888	HH hhh	
II	iiii	999	II iiii	

Note: Here we can combine multiple columns of a single data set but not combine columns of the multiple data sets. We can combine up to 28 columns only.

- **CONCATENATE:** Joins two text strings into one text string. We can combine only two columns of a data set. We can't combine multiple columns here. We can't use delimiter here.

Syntax: CONCATENATE(<text1>, <text2>)

Dax Function:

Concatenate = CONCATENATE('Text Data'[String1], 'Text Data'[String2])

Ex)

Structure			Formatting	Properties
X	✓	1	Concatenate = CONCATENATE('Text Data'[String1], 'Text Data'[String2])	
String1	String2	String3	CombineValues	Concatenate
AA	aaa	111	AA aaa	AAaaaa
BB	bbb	222	BB bbb	BBbbbb
CC	ccc	333	CC ccc	CCcccc
DD	ddd	444	DD ddd	DDdddd
EE	eee	555	EE eee	EEeeee
FF	fff	666	FF fff	FFffff
GG	ggg	777	GG ggg	GGgggg
HH	hhh	888	HH hhh	HHhhhh
II	iiii	999	II iiii	IIiiii

Note: In concatenate function, If you want to combine two columns with delimiter then we have to write the dax function as below.

Dax Function:

Concatenate1 = `CONCATENATE('Text Data'[String1],CONCATENATE("|",'Text Data'[String2]))`

Ex)

Structure		Formatting		Properties	
X	✓	1	Concatenate1 = <code>CONCATENATE('Text Data'[String1],CONCATENATE(" ",'Text Data'[String2]))</code>	ConcatenateX	Concatenate1
String1	String2	String3	CombineValues	Concatenate	Concatenate1
AA	aaa	111	AA aaa	AAAAA	AA aaa
BB	bbb	222	BB bbb	BBBBB	BB bbb
CC	ccc	333	CC ccc	CCCCC	CC ccc
DD	ddd	444	DD ddd	DDDDD	DD ddd
EE	eee	555	EE eee	EEEEEE	EE eee
FF	fff	666	FF fff	FFFFFF	FF fff
GG	ggg	777	GG ggg	GGGGGG	GG ggg
HH	hhh	888	HH hhh	HHHHHH	HH hhh
II	iiii	999	II iiii	IIIIII	II iiii

Difference between combine values and concatenate:

- ✓ In combine values, we can combine multiple columns or strings with delimiter, but we can combine only two columns with concatenate.
- ✓ We can use delimiter in combine values, but we don't use delimiter in concatenate function

- **CONCATENATEX:** Concatenates the result of an expression evaluated for each row in a table.

That means it will combine the rows of a column in a table, display in single row and replicates the same value to all the corresponding rows.

Syntax: `CONCATENATEX(<table>, <expression>[, <delimiter> [, <orderBy_expression> [, <order>]]...])`

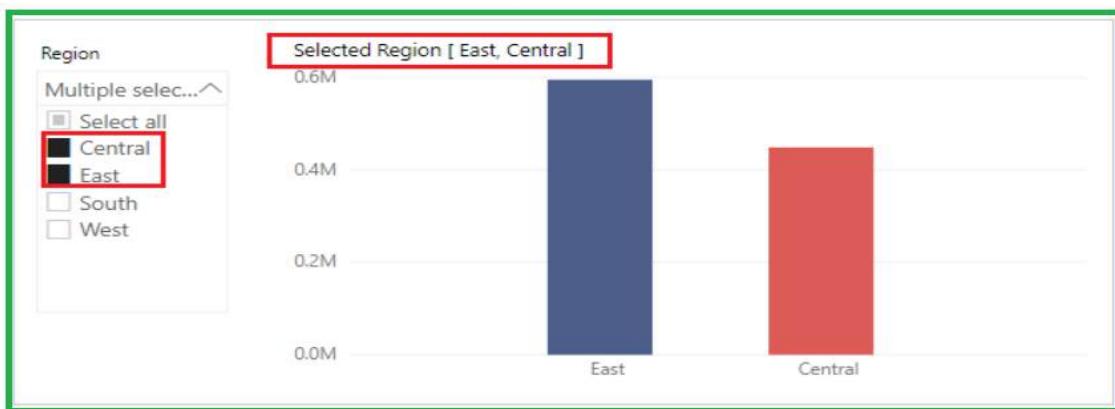
Dax Function:

ConcatenateX = `CONCATENATEX('Text Data','Text Data'[String1],":")`

Ex)

ConcatenateX = CONCATENATEX('Text Data', 'Text Data'[String1], ":")					
String1	String2	String3	CombineValues	Concatenate	ConcatenateX
AA	aaa	111	AA aaa	AAaaa	AA:BB:CC:DD:EE:FF:GG:HH:II
BB	bbb	222	BB bbb	BBbbb	AA:BB:CC:DD:EE:FF:GG:HH:II
CC	ccc	333	CC ccc	CCccc	AA:BB:CC:DD:EE:FF:GG:HH:II
DD	ddd	444	DD ddd	DDddd	AA:BB:CC:DD:EE:FF:GG:HH:II
EE	eee	555	EE eee	EEeee	AA:BB:CC:DD:EE:FF:GG:HH:II
FF	fff	666	FF fff	FFfff	AA:BB:CC:DD:EE:FF:GG:HH:II
GG	ggg	777	GG ggg	GGggg	AA:BB:CC:DD:EE:FF:GG:HH:II
HH	hhh	888	HH hhh	HHhhh	AA:BB:CC:DD:EE:FF:GG:HH:II
II	iiii	999	II iiii	IIiiii	AA:BB:CC:DD:EE:FF:GG:HH:II

Note: In slicer visual, sometimes need to put the combine of values as a title name



```
TitleTxt =
VAR GetValues= CONCATENATE(
VALUES(Orders[Region]), Orders[Region], ", ")
RETURN
"Selected Region [ " & GetValues & " ]"
```

- **EXACT:** Compares two text strings or two column values and returns TRUE if they are exactly the same, otherwise returns FALSE.

Note: EXACT is case-sensitive

Syntax: EXACT(<text1>,<text2>)

Dax Function:

Exact = EXACT('Two Columns'[Column1],'Two Columns'[Column2])

Ex)

Structure		Formatting	Properties
X	✓	1 Exact = EXACT('Two Columns'[Column1],'Two Columns'[Column2])	
Column1	Column2	Exact	
0	0	True	
1	0	False	
1	1	True	
0	0	True	
-1	0	False	
-2	-2	True	
1	-1	False	
-2	-1	False	
1	1	True	

- **SEARCH:** To find the position of the character in a text string or a column value we have to use search function.

Note: It is not case sensitive

For example, I am taking below table for applying search function. Table name for this table is “DATE”

DATE	NUM	Day	Merged	Merged.1
15-09-2022	1	15	September:Thursday	2022:September
08-05-2020		8	May:Friday	2020:May
11-03-2019	0	11	March:Monday	2019:March
05-02-2018	21	5	February:Monday	2018:February
02-05-2016	2	2	May:Monday	2016:May
15-02-2017		15	February:Wednesday	2017:February

Syntax: `SEARCH(<find_text>, <within_text>[, <start_num>][, <NotFoundValue>])`

Dax Function:

Search = `Search(":",'DATE'[Merged],3,0)`

Note: In the above example need to find the “:” delimiter in each cell of the column “Merged” in “DATE” table.

Search should start from number mentioned in 3rd argument. Here start_num = 3. If not found search should return 0

Terms and their definitions

find_text -> The text that you want to find

Within_text -> The text in which you want to search for **find_text**, or a column containing text.

Start_num ->(optional) The character position in **within_text** at which you want to start searching. If omitted, 1.

NotFoundValue -> (optional, but strongly recommended) The value that should be returned when the operation does not find a matching substring, typically 0, -1, or BLANK(). If not specified, an error is returned.

- **FIND:** Returns the starting position of one text string within another text string or column containing that text.

Note: Find is case sensitive.

Syntax:

`FIND(<find_text>,<within_text>[,<start_num>][,<NotFoundValue>])`

Dax Function:

`Find = Find("e",'DATE'[Merged],3,0)`

Ex)

Structure		Formatting						
X	✓	1	Find = Find("e",'DATE'[Merged],3,0)					
DATE	NUM	Day	Merged	Merged.1	Search	Find		
15-09-2022	1	15	September:Thursday	2022:September	10	5		
08-05-2020		8	May:Friday	2020:May	4	0		
11-03-2019	0	11	March:Monday	2019:March	6	0		
05-02-2018	21	5	February:Monday	2018:February	9	0		
02-05-2016	2	2	May:Monday	2016:May	4	0		
15-02-2017		15	February:Wednesday	2017:February	9	1	0	

Case Sensitive: Suppose I want to find the position of “M” in the column “**Merged**” in search function, then it will skip the “m” values position in that column.

It will display only “M” position of their corresponding rows. But in “Search” function it will consider “m” characters also. See the below screenshots

Find:

Structure				Formatting			
							Find
	X	✓	1 Find = Find("M", 'DATE'[Merged],3,0)				
DATE	NUM	Day	Merged	Merged.1	Search	Find	
15-09-2022	1	15	September:Thursday	2022:September	10	0	
08-05-2020		8	May:Friday	2020:May	4	0	
11-03-2019	0	11	March:Monday	2019:March	6	7	
05-02-2018	21	5	February:Monday	2018:February	9	10	
02-05-2016	2	2	May:Monday	2016:May	4	5	
15-02-2017		15	February:Wednesday	2017:February	9	0	

Search:

Structure				Formatting			
							Search
	X	✓	1 Search = Search("M", 'DATE'[Merged],1,0)				
DATE	NUM	Day	Merged	Merged.1	Search		
15-09-2022	1	15	September:Thursday	2022:September	6		
08-05-2020		8	May:Friday	2020:May	1		
11-03-2019	0	11	March:Monday	2019:March	1		
05-02-2018	21	5	February:Monday	2018:February	10		
02-05-2016	2	2	May:Monday	2016:May	1		
15-02-2017		15	February:Wednesday	2017:February	0		

Note:

- ✓ Always we have to use “search” function only.
- ✓ Both functions will return the position of a first character in a search string. Suppose if we want to find the position of “Mb”, then both functions will return the position of the first character “M”.

➤ **Fixed:** It will convert numeric value in decimal value. That means it rounds a number to the specified number of decimals and returns the result as text.

Syntax:

`FIXED(<number>, <decimals>, <no_commas>)`

Dax Function:

`Fixed = Fixed(COMPARE[id1],2)`

Ex)

Structure					Formatting
X	✓	1	Fixed = Fixed(COMPARE[id1],2)		
id1	id2	name	Exact	Fixed	
1	0	a	False	1.00	
3	3	b	True	3.00	
15	1	c	False	15.00	
1	1	d	True	1.00	
4	2	e	False	4.00	
5	5	f	True	5.00	

- **LEFT:** Returns the specified number of characters from the start of a text string.

Syntax: LEFT(<text>, <num_chars>)

Dax Function:

Left = LEFT('Format'[Sentence],6)

Note: Based on 2nd argument; **LEFT** function returns the specified no of characters from left side.

Ex)

Structure					Formatting
X	✓	1	Left = LEFT('Format'[Sentence],6)		
Lower	Space	Sentence	Upper	Left	
Abril	Space	United States	Mara	United	
Hashimoto	12222	Great Britain Uk	Philip	Great	
Gent	ABC 12A	France	Kathleen	France	
Hanner	22F	United States	Nereida	United	
Magwood	23223323	United States	Gaston	United	
Brumm	232232	London	Etta	London	
Hurn	AFDFD	India	Earlean	India	
Melgar	ADF	United Arab Emirates	Vincenza	United	
Weiland	~	India	Fallon	India	
Winward	DF	Aruna Chal Pradesh	Arcelia	Aruna	
Bouska	FDDDFDF	China	Franklyn	China	
Unknow	DFD	France	Sherron	France	

- **RIGHT:** Returns the specified number of characters from right most characters of a text string.

Syntax: RIGHT(<text>, <num_chars>)

Dax Function:

Left = RIGHT('Format'[Sentence],6)

Ex)

Structure		Formatting						
		1 Right = RIGHT('Format'[Sentence],6)	Lower	Space	Sentence	Upper	Left	Right
Abril	Space	United States	Mara	United	States	States		
Hashimoto	12222	Freat Britain Uk	Philip	Freat	ain Uk			
Gent	ABC 12A	France	Kathleen	France	France			
Hanner	22F	United States	Nereida	United	States			
Magwood	23223323	United States	Gaston	United	States			
Brumm	223232	London	Etta	London	London			
Hurn	AFDFD	India	Earlean	India	India			
Melgar	ADF	United Arab Emirates	Vincenza	United	irates			
Weiland	~	India	Fallon	India	India			
Winward	DF	Aruna Chal Pradesh	Arcelia	Aruna	raresh			
Bouska	FDDDFDF	China	Franklyn	China	China			
Unknow	DFD	France	Sherron	France	France			

Note1: Based on 2nd argument; **RIGHT** function returns the specified no of characters from right side.

Note2: Here “**num_chars**” is optional, if you omitted it, it will return 1 character.

- **LOWER:** Converts all letters in a text string to lowercase.

Syntax: LOWER(<text>)

Dax Function:

Lower = Lower('Format'[Column6])

Ex)

Structure		Formatting						Properties				
		1 Lower = Lower('Format'[Column6])	Small	Space	Sentence	Big	Left	Right	Column5	Column6	Lower	Upper
Abril	Space	United States	Mara	United	States	Small			Big	Big	big	SMALL
Hashimoto	12222	Freat Britain Uk	Philip	Freat	ain Uk	philip			HASHIMOTO	HASHIMOTO	hashimoto	PHILIP
Gent	ABC 12A	France	Kathleen	France	France	kathleen			GENT	GENT	gent	KATHLEEN
Hanner	22F	United States	Nereida	United	States	nareida			HANNER	HANNER	hanner	NAREIDA
Magwood	23223323	United States	Gaston	United	States	gaston			MAGWOOD	MAGWOOD	magwood	GASTON
Brumm	223232	London	Etta	London	London	etta			BRUMM	BRUMM	brumm	ETTA
Hurn	AFDFD	India	Earlean	India	India	earlean			HURN	HURN	hurn	EARLEAN
Melgar	ADF	United Arab Emirates	Vincenza	United	irates	vincenza			MELGAR	MELGAR	melgar	VINCENZA
Weiland	~	India	Fallon	India	India	fallon			WEILAND	WEILAND	weiland	FALLON
Winward	DF	Aruna Chal Pradesh	Arcelia	Aruna	raresh	arcelia			WINWARD	WINWARD	winward	ARCELIA
Bouska	FDDDFDF	China	Franklyn	China	China	frnaklyn			BOUSKA	BOUSKA	bouska	FRNAKLYN
Unknow	DFD	France	Sherron	France	France	sherron			UNKNOWN	UNKNOWN	unknown	SHERRON

- **UPPER:** Converts all letters in a text string to all uppercase letters.

Syntax: UPPER(<text>)

Dax Function:

Upper = UPPER('Format'[Column5])

Ex)

Structure Formatting Properties

1 `Upper = Upper('Format'[Column5])`

Small	Space	Sentence	Big	Left	Right	Column5	Column6	Lower	Upper
Abril	Space	United States	Mara	United	States	Small	Big	big	SMALL
Hashimoto	12222	Freat Britain Uk	Philip	Freat	ain Uk	philip	HASHIMOTO	hashimoto	PHILIP
Gent	ABC 12A	France	Kathleen	France	France	kathleen	GENT	gent	KATHLEEN
Hanner	22F	United States	Nereida	United	States	nareida	Hanner	hanner	NAREIDA
Magwood	23223323	United States	Gaston	United	States	gaston	MAGWOOD	magwood	GASTON
Brumm	223232	London	Etta	London	London	etta	BRUMM	brumm	ETTA
Hurn	AFDFD	India	Earlean	India	India	earlean	HURN	hurn	EARLEAN
Melgar	ADF	United Arab Emirates	Vincenza	United	irates	vincenza	MELGAR	melgar	VINCENZA
Weiland	~	India	Fallon	India	India	fallon	WEILAND	weiland	FALLON
Winward	DF	Aruna Chal Pradesh	Arcelia	Aruna	radesh	arcelia	WINWARD	winward	ARCELIA
Bouska	FDDDFDF	China	Franklyn	China	China	frnaklyn	BOUSKA	bouska	FRNAKLYN
Unknow	DFD	France	Sherron	France	France	sherron	UnKnown	unknown	SHERRON

- **LEN:** Returns the number of characters in a text string or column containing text string.

Syntax: `LEN(<text>)`

Dax Function:

`Len = LEN('Format'[Sentence])`

Ex)

Structure Formatting Properties Sort

1 `len = LEN('Format'[Sentence])`

Small	Space	Sentence	Big	Left	Right	Columns	Column5	Lower	Upper	len
Abril	Space	United States	Mara	United	States	Small	Big	big	SMALL	13
Hashimoto	12222	Freat Britain Uk	Philip	Freat	ain Uk	philip	HASHIMOTO	hashimoto	PHILIP	16
Gent	ABC 12A	France	Kathleen	France	France	kathleen	GENT	gent	KATHLEEN	6
Hanner	22F	United States	Nereida	United	States	nareida	Hanner	hanner	NAREIDA	13
Magwood	23223323	United States	Gaston	United	States	gaston	MAGWOOD	magwood	GASTON	13
Brumm	223232	London	Etta	London	London	etta	BRUMM	brumm	ETTA	6
Hurn	AFDFD	India	Earlean	India	India	earlean	HURN	hurn	EARLEAN	5
Melgar	ADF	United Arab Emirates	Vincenza	United	irates	vincenza	MELGAR	melgar	VINCENZA	20
Weiland	~	India	Fallon	India	India	fallon	WEILAND	weiland	FALLON	5
Winward	DF	Aruna Chal Pradesh	Arcelia	Aruna	radesh	arcelia	WINWARD	winward	ARCELIA	18
Bouska	FDDDFDF	China	Franklyn	China	China	frnaklyn	BOUSKA	bouska	FRNAKLYN	5
Unknow	DFD	France	Sherron	France	France	sherron	UnKnown	unknown	SHERRON	6

- **TRIM:** Removes all spaces from text except for single spaces between words.

Syntax: `TRIM(<text>)`

Dax Function:

`Trim = TRIM('TRIM TABLE'[Column1])`

Ex)

Structure Formatting

1 `Trim = TRIM('TRIM TABLE'[Column1])`

Column1	Trim
BHASKAR RAO	BHASKAR RAO
AMEELA	AMEELA
JEEVAN	JEEVAN
SANTOSHI	SANTOSHI
NIHARIKA	NIHARIKA
SRINIVASA RAO	SRINIVASA RAO

- **REPLACE:** It replaces part of a text string, based on the number of characters you specify, with a different text string.

Syntax: `REPLACE(<old_text>, <start_num>, <num_chars>, <new_text>)`

Dax Function:

Replace = `REPLACE('Text Data'[CombineValues],3,1,":")`

Ex)

Structure			Formatting			Properties			Sort
X	✓	1	<code>Replace = REPLACE('Text Data'[CombineValues],3,1,":")</code>						
String1	String2	String3	CombineValues	Concatenate	ConcatenateX	Concatenate1	Replace		
AA	aaa	111	AA aaa	AAaaa	AA:BB:CC:DD:EE:FF:GG:HH:II	AA aaa	AA:aaa		
BB	bbb	222	BB bbb	BBbbb	AA:BB:CC:DD:EE:FF:GG:HH:II	BB bbb	BB:bbb		
CC	ccc	333	CC ccc	CCccc	AA:BB:CC:DD:EE:FF:GG:HH:II	CC ccc	CC:ccc		
DD	ddd	444	DD ddd	DDddd	AA:BB:CC:DD:EE:FF:GG:HH:II	DD ddd	DD:ddd		
EE	eee	555	EE eee	EEeee	AA:BB:CC:DD:EE:FF:GG:HH:II	EE eee	EE:eee		
FF	fff	666	FF fff	FFfff	AA:BB:CC:DD:EE:FF:GG:HH:II	FF fff	FF:fff		
GG	ggg	777	GG ggg	GGggg	AA:BB:CC:DD:EE:FF:GG:HH:II	GG ggg	GG:ggg		
HH	hhh	888	HH hhh	HHhhh	AA:BB:CC:DD:EE:FF:GG:HH:II	HH hhh	HH:hhh		
II	iiii	999	II iiii	IIiiii	AA:BB:CC:DD:EE:FF:GG:HH:II	II iiii	II:iiii		

Here, I want to replace “|” with “:” in “**CombineValues**” column then I have given column name, 3 (Position to replace i.e “|”), 1 (Number of characters to replace i.e 1), “:” means new character.

Note: In REPLACE function, we can replace all the characters with specified characters.

Ex)

Structure			Formatting			Properties			Sort	Groups
X	✓	1	<code>Replace1 = Replace("Venkat Rao",2,9,"ENKAT")</code>							
String1	String2	String3	CombineValues	Concatenate	ConcatenateX	Concatenate1	Replace	Replace1		
AA	aaa	111	AA aaa	AAaaa	AA:BB:CC:DD:EE:FF:GG:HH:II	AA aaa	AA:aaa	VENKAT		
BB	bbb	222	BB bbb	BBbbb	AA:BB:CC:DD:EE:FF:GG:HH:II	BB bbb	BB:bbb	VENKAT		
CC	ccc	333	CC ccc	CCccc	AA:BB:CC:DD:EE:FF:GG:HH:II	CC ccc	CC:ccc	VENKAT		
DD	ddd	444	DD ddd	DDddd	AA:BB:CC:DD:EE:FF:GG:HH:II	DD ddd	DD:ddd	VENKAT		
EE	eee	555	EE eee	EEeee	AA:BB:CC:DD:EE:FF:GG:HH:II	EE eee	EE:eee	VENKAT		
FF	fff	666	FF fff	FFfff	AA:BB:CC:DD:EE:FF:GG:HH:II	FF fff	FF:fff	VENKAT		
GG	ggg	777	GG ggg	GGggg	AA:BB:CC:DD:EE:FF:GG:HH:II	GG ggg	GG:ggg	VENKAT		
HH	hhh	888	HH hhh	HHhhh	AA:BB:CC:DD:EE:FF:GG:HH:II	HH hhh	HH:hhh	VENKAT		
II	iiii	999	II iiii	IIiiii	AA:BB:CC:DD:EE:FF:GG:HH:II	II iiii	II:iiii	VENKAT		

Here, from 2nd character onwards, total 9 characters has been replaced with specified string “**ENKAT**”.

- **SUBSTITUTE:** Replaces existing text with new text in a text string.

Syntax: `SUBSTITUTE(<text>, <old_text>, <new_text>, <instance_num>)`

Dax Function:

Substitute = `SUBSTITUTE('Text Data'[ConcatenateX],":","&",3)`

Ex)

Structure		Formatting		Properties		Sort		Groups		Relationships		Calculations	
String1	String2	String3	CombineValues	Concatenate	ConcatenateX	Concatenate1	Replace	Replace1	Substitute				
AA	aaa	111	AA aaa	AAaaaa	AA:BB:CC:DD:EE:FF:GG:HH	AA aaa	AA aaa	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				
BB	bbb	222	BB bbb	B8bbb	AA:BB:CC:DD:EE:FF:GG:HH	BB bbb	BB bbb	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				
CC	ccc	333	CC ccc	CCccc	AA:BB:CC:DD:EE:FF:GG:HH	CC ccc	CC ccc	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				
DD	ddd	444	DD ddd	DDddd	AA:BB:CC:DD:EE:FF:GG:HH	DD ddd	DD ddd	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				
EE	eee	555	EE eee	EEeee	AA:BB:CC:DD:EE:FF:GG:HH	EE eee	EE eee	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				
FF	fff	666	FF fff	FFfff	AA:BB:CC:DD:EE:FF:GG:HH	FF fff	FF fff	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				
GG	ggg	777	GG ggg	GGggg	AA:BB:CC:DD:EE:FF:GG:HH	GG ggg	GG ggg	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				
HH	hhh	888	HH hhh	HHhhh	AA:BB:CC:DD:EE:FF:GG:HH	HH hhh	HH hhh	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				
II	iii	999	II iii	IIiii	AA:BB:CC:DD:EE:FF:GG:HH	II iii	II iii	VENKAT	AA:BB:CC&DD:EE:FF:GG:HH:				

Note: substitute function replace “|” value with “&” in 3 rd instance only. If you don’t give instance i.e 3rd argument; then substitute will replace all the old values with new value in all instances.

- **FORMAT:** Converts a value to text according to the specified format.

Syntax:

FORMAT(<value>, <format_string>[, <locale_name>])

Dax Function:

Format_Date = FORMAT('Date Table'[Date], "dd/MM/yyyy")

Ex)

Structure		Formatting	
X	✓	1 Format_Date = FORMAT('Date Table'[Date], "dd/MM/yyyy")	
Date	Date - Copy	Format_Date	
10-11-2022	10 November 2022	10/11/2022	
05-10-2021	05 October 2021	05/10/2021	
14-08-2019	14 August 2019	14/08/2019	
21-07-2020	21 July 2020	21/07/2020	
22-04-2021	22 April 2021	22/04/2021	

If you want to display month name, month number, week name etc, then you have to use format function.

Returning different formats on date column:

- | | |
|----------------|--|
| Day | : Format_Day = FORMAT('Date Table'[Date], "d") |
| | : Format_Day = FORMAT('Date Table'[Date], "dd") |
| Day in Short | : Format_Day = FORMAT('Date Table'[Date], "ddd") |
| Day in Full | : Format_Day = FORMAT('Date Table'[Date], "dddd") |
| Month | : Format_Month = FORMAT('Date Table'[Date], "M") |
| | : Format_Month = FORMAT('Date Table'[Date], "MM") |
| Month in Short | : Format_Month = FORMAT('Date Table'[Date], "MMM") |

Month in Full : Format_Month = `FORMAT('Date Table'[Date],"MMMM")`

Year : Format_Year = `FORMAT('Date Table'[Date],"y")`
: Format_Year = `FORMAT('Date Table'[Date],"yy")`

Year in Short : Format_Year = `FORMAT('Date Table'[Date],"yyy")`
Year in Full : Format_Year = `FORMAT('Date Table'[Date],"yyyy")`

Current Date in IST format

Format_in_IST = `FORMAT(now(),"hh:mm am/pm dddd, dd MMMM yyyy")`

Structure		Formatting		Properties	
X	✓	1	Format_in_IST = <code>FORMAT(now(),"hh:mm am/pm dddd, dd MMMM yyyy")</code>		
Date	Date - Copy	Format_Date	Format_in_IST		
10-11-2022	10 November 2022	10/11/2022	03:33 pm Wednesday, 19 October 2022		
05-10-2021	05 October 2021	05/10/2021	03:33 pm Wednesday, 19 October 2022		
14-08-2019	14 August 2019	14/08/2019	03:33 pm Wednesday, 19 October 2022		
21-07-2020	21 July 2020	21/07/2020	03:33 pm Wednesday, 19 October 2022		
22-04-2021	22 April 2021	22/04/2021	03:33 pm Wednesday, 19 October 2022		

Note: For month related, we have to put capital letters like "M".

Format as Currency:

Format_in_Currency = `FORMAT('Value'[Num],"$#.00")`

Structure		Formatting		Properties	
X	✓	1	Format_in_Currency = <code>FORMAT('Value'[Num],"\$#.00")</code>		
Num	Format_in_Currency				
15	\$15.00				
8	\$8.00				
11	\$11.00				
5	\$5.00				
2	\$2.00				
15	\$15.00				

Note: Here "#" is digit place holder. \$ is currency type. (.) is Decimal placeholder

- **MID:** Returns a string of characters from the middle of a text string, given a starting position and length.

Syntax: `MID(<text>, <start_num>, <num_chars>)`

Dax Function:

Mid = `MID('Format'[Sentence],2,5)`

Ex)

Structure Formatting Properties Sort Groups

1 `Mid = MID('Format'[Sentence],1,5)`

Sentence	Big	Left	Right	Column5	Column6	Lower	Upper	Len	Mid
Abrial Space United States	Mara	United	States	Small	Big	big	SMALL	13	nited
Hashimoto 12222 Freat Britain Uk	Philip	Freat	ain Uk	philip	HASHIMOTO	hashimoto	PHILIP	16	reat
Gent ABC 12A France	Kathleen	France	France	kathleen	GENT	gent	KATHLEEN	6	rance
Hanner 22F United States	Nereida	United	States	nareida	Hanner	hanner	NAREIDA	13	nited
Magwood 23223323 United States	Gaston	United	States	gaston	MAGWOOD	magwood	GASTON	13	nited
Brumm 223232 London	Etta	London	London	etta	BRUMM	brumm	ETTA	6	ondon
Hurn AFD FD India	Earlean	India	India	earlean	HURN	hurn	EARLEAN	5	ndia
Melgar ADF United Arab Emirates	Vincenza	United	Irates	vincenza	MELGAR	melgar	VINCENZA	20	nited
Weiland ~ India	Fallon	India	India	fallon	WEILAND	weiland	FALLON	5	ndia
Winward DF Aruna Chal Pradesh	Arcelia	Aruna	radesh	arcelia	WINWARD	winward	ARCELIA	18	runa
Bouska FDDDFD China	Franklyn	China	China	frnaklyn	BOUSKA	bouska	FRNAKLYN	5	hina
Unknow DFD France	Sherron	France	France	sherron	UnKnown	unknown	SHERRON	6	rance

Note: Here <start_num> means “The position of the first character you want to extract. Position starts from 1.

<num_chars> means “Number of characters to return”.

- **VALUE:** Converts a text string that represents a number to a decimal number. That means if you want to convert number or numeric value in text data type to decimal, then we can use Value function.

Syntax: `VALUE(<text>)`

Dax Function:

`Value = VALUE('Value Table'[Num in Text])`

Ex)

Name	Value	Format	Text
123 Data type	Decimal number	\$ %	General
Structure	Formatting		

Name	Value	Format	General
123 Data type	Text	\$ %	General
Structure	Formatting		

- **REPT:** Repeats text a given number of times. Use REPT to fill a cell with a number of instances of a text string.

Syntax: `REPT(<text>, <num_times>)`

Dax Function:

`Rept = REPT('Text Data'[String1],4)`

Ex)

Structure		Formatting	
X	✓	1	Rept = Rept("Power Bi",5)
Num in Text	Value	Rept	
10	10	Power BiPower BiPower BiPower Bi	
15	15	Power BiPower BiPower BiPower Bi	
52	52	Power BiPower BiPower BiPower Bi	
1	1	Power BiPower BiPower BiPower Bi	
22	22	Power BiPower BiPower BiPower Bi	
22	22	Power BiPower BiPower BiPower Bi	

Logical Functions

Logical functions act upon an expression to return information about the values or sets in the expression.

- **If:** Checks a condition, and returns one value when it's TRUE, otherwise it returns a second value.

Syntax:

`IF(<logical_test>, <value_if_true>[, <value_if_false>])`

Note: Either **value_if_true**, **value_if_false**, or **BLANK**

Dax Function:

If = `IF('Format'[Len]>=10,1,0)`

Ex)

Structure		Formatting		Properties		Sort		Groups	
X	✓	1	[# - IF('Format'[Len]>=10,1,0)]	Len	Mid	if			
Abrial	Space	United States	Mara	United	States	Small	Big	big	SMALL
Hashimoto	12222	Great Britain UK	Philip	Great	ain Uk	philip	HASHIMOTO	hashimoto	PHILIP
Gent	ABC 12A	France	Kathleen	France	France	kathleen	GENT	gent	KATHLEEN
Hanner	22F	United States	Nereida	United	States	nareida	HANNER	hanner	NAREIDA
Magwood	23223323	United States	Gaston	United	States	gaston	MAGWOOD	magwood	GASTON
Brumm	223232	London	Etta	London	London	etta	BRUMM	brumm	ETTA
Hurn	AFDFD	India	Earlean	India	India	earlean	HURN	hurn	EARLEAN
Melgar	ADF	United Arab Emirates	Vincenza	United	Irates	vincenza	MELGAR	melgar	VINCENZA
Weiland	~	India	Fallon	India	India	fallon	WEILAND	weiland	FALLON
Winward	DF	Aruna Chal Pradesh	Arcilia	Aruna	radesh	arcilia	WINWARD	winward	ARCELIA
Bouska	FOODFDF	China	Franklyn	China	China	frnaklyn	BOUSKA	bouska	FRNAKLYN
Unknow	DFD	France	Sherron	France	France	sherron	UnKnown	unknown	SHERRON

Here I have written a condition as if "len" column value is greater than or equal to 10 then it should return 1, otherwise it should return 0

Here you can write "True", or "False" values in 2nd and 3rd arguments.

Ex)

If = `IF('Format'[Len]=6,"XYZ","ABC")`

If = `IF('Format'[Len]>=10,TRUE, FALSE)`

If = **IF('Format'[Len]<>6,1,0)**
If = **IF('Format'[Len]<=10,"Required","Not Required")**



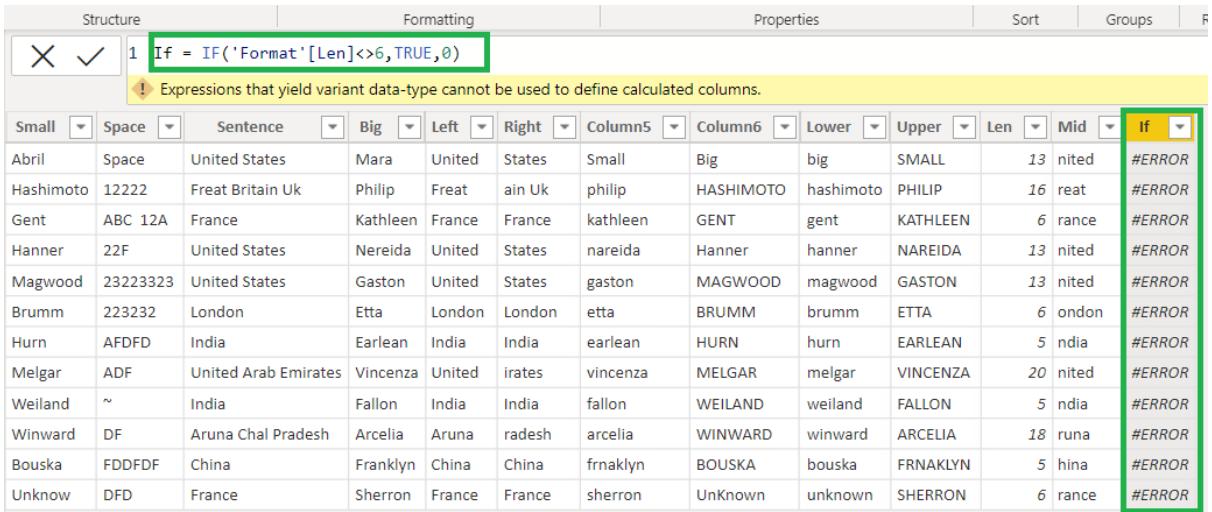
The screenshot shows a Microsoft Excel table with 15 columns. The last column, 'If', contains the formula **IF('Format'[Len]<=10,"Required","Not Required")**. A dropdown menu is open over this column, showing the options 'Not Required', 'Not Required', 'Required', 'Not Required', 'Not Required', 'Required', 'Required', 'Required', 'Required', 'Required', 'Required', 'Required', 'Required', 'Required', and 'Required'. The first two rows show the formula in the cell and its result ('Not Required').

Note: Here return values for this function always will be in same data type. If they are different, immediately it will through an error. That means 2nd and 3rd argument should be in same data type.

Ex)

If = **IF('Format'[Len]<>6,1,0)**

Error: Expressions that yield variant data-type cannot be used to define calculated columns.



The screenshot shows the same table as above, but the formula in the 'If' column has changed to **IF('Format'[Len]<>6,TRUE,0)**. A yellow warning bar at the top states 'Expressions that yield variant data-type cannot be used to define calculated columns.' The 'If' column now displays '#ERROR' for every row.

Suppose if you want to write multiple conditions in "IF" function then you need to use below operators between all the conditions.

&& : If both conditions satisfied then it returns 1, otherwise returns 0
|| : If either of the conditions satisfied it returns 1, otherwise returns 0

Ex)

Dax Function:

If_multiple_condition = **If('DATE'[index changed]<=4 && 'DATE'[Index.1]<=2,1,0)**

		Structure		Formatting		Properties			
X	✓	1	If_multiple_condition = If('DATE'[index changed]<=4 && 'DATE'[Index.1]<=2,1,0)						
index changed	Index.1	If_single_condition	If_multiple_condition	Day Name	Switch	AND			
0	0	TRUE		1 Thursday	THU	0			
1	2	TRUE		1 Friday	FRI	0			
4	4	TRUE		0 Monday	Mon	0			
6	6	FALSE		0 Monday	Mon	1			
4	8	TRUE		0 Monday	Mon	0			
5	10	FALSE		0 Wednesday	SUN	0			

Dax Function:

If_multiple_condition = If('DATE'[index changed]<=4 || 'DATE'[Index.1]<=2,1,0)

		Structure		Formatting		Properties			
X	✓	1	If_multiple_condition = If('DATE'[index changed]<=4 'DATE'[Index.1]<=2,1,0)						
index changed	Index.1	If_single_condition	If_multiple_condition	Day Name	Switch	AND			
0	0	TRUE		1 Thursday	THU	0			
1	2	TRUE		1 Friday	FRI	0			
4	4	TRUE		1 Monday	Mon	0			
6	6	FALSE		0 Monday	Mon	1			
4	8	TRUE		1 Monday	Mon	0			
5	10	FALSE		0 Wednesday	SUN	0			

- **SWITCH:** Evaluates an expression against a list of values and returns one of multiple possible result expressions. By this function, we can return multiple outputs.

Syntax:

SWITCH(<expression>, <value>, <result>[, <value>, <result>]...[, <else>])

Note:

- ✓ All result expressions and else expression must be of same data type.
- ✓ Always we should write expression as true() in switch condition. Then only it will enter into the switch loop.

Dax Function:

```
Switch = SWITCH(True(),
'DATE'[Day Name]="Monday","Mon",
'DATE'[Day Name]="Tuesday","TUE",
'DATE'[Day Name]="Wednesday","WED",
'DATE'[Day Name]="Thursday","THU",
'DATE'[Day Name]="Friday","FRI",
'DATE'[Day Name]="Saturday","SAT","SUN")
```

Ex)

Structure		Formatting		
Index.1	If_single_condition	If_multiple_condition	Day Name	Switch
0	TRUE		Thursday	THU
2	TRUE		Friday	FRI
4	TRUE		Monday	Mon
6	FALSE		Monday	Mon
8	TRUE		Monday	Mon
10	FALSE		Wednesday	SUN

Note:

- ✓ If you want to go to the next line in the switch function, you have to press "shift"+"Enter".
- ✓ Suppose if we have to write conditions for "N" values then you have to write "N-1" conditions, and "N" condition we can write directly as last value.
- ✓ Even though if we have only "**Sunday**", "**Tuesday**" and "**Saturday**" in a column, you have to write conditions for all **Day Names**
- ✓ We can write multiple conditions in **SWITCH** function by using either "**&&**" or "**||**".

➤ **AND**: Checks whether both arguments are TRUE, and returns TRUE if both arguments are TRUE. Otherwise returns false.



Syntax:

AND(<logical1>,<logical2>)

Dax Function:

AND = IF(and('DATE'[Day Name]="Monday",'DATE'[index changed]=5),1,0)

Ex)

Structure		Formatting		Properties		
index changed	Index.1	If_single_condition	If_multiple_condition	Day Name	Switch	AND
0	0	TRUE		Thursday	THU	0
1	2	TRUE		Friday	FRI	0
4	4	TRUE		Monday	Mon	0
6	6	FALSE		Monday	Mon	1
8	8	TRUE		Monday	Mon	0
5	10	FALSE		Wednesday	SUN	0

Note: We can write only two conditions in **AND** function, we can't write multiple conditions. If you want to write multiple conditions you have to use “&&” symbol in between all conditions.

- **OR:** Checks whether one of the arguments is TRUE to return TRUE. The function returns FALSE if both arguments are FALSE.

Syntax:

OR(<logical1>,<logical2>)

Dax Function:

OR = If(OR('DATE'[Day Name]="Monday",'DATE'[index changed]=4),"T","F")

Ex)

index changed	Index.1	If_single_condition	If_multiple_condition	Day Name	Switch	AND	OR
0	0	TRUE		1 Thursday	THU	0	F
1	2	TRUE		1 Friday	FRI	0	F
4	4	TRUE		1 Monday	Mon	0	T
6	6	FALSE		0 Monday	Mon	1	T
4	8	TRUE		1 Monday	Mon	0	T
5	10	FALSE		0 Wednesday	SUN	0	F

Note: We can write only two conditions in **OR** function, we can't write multiple conditions. If you want to write multiple conditions you have to use “||” symbol in between all conditions.

- **NOT:** Changes FALSE to TRUE, or TRUE to FALSE.

Syntax:

NOT(<logical>)

Dax Function:

NOT = If(NOT('DATE'[Day Name])="Monday",1,0)

Ex)

Structure		Formatting		Properties	
X	✓	1	NOT = If(NOT('DATE'[Day Name])="Monday",1,0)		
Index.1	If_single_condition	If_multiple_condition	Day Name	Switch	AND
0	TRUE		1 Thursday	THU	0 F
2	TRUE		1 Friday	FRI	0 F
4	TRUE		1 Monday	Mon	0 T
6	FALSE		0 Monday	Mon	1 T
8	TRUE		1 Monday	Mon	0 T
10	FALSE		0 Wednesday	SUN	0 F
					NOT
					1
					1
					0
					0
					0
					1

- **TRUE**: Returns the logical value TRUE.

Syntax: `TRUE()`

Dax Function:

`If_single_condition = If('DATE'[index changed]<=4,TRUE(),FALSE())`

Ex)

Structure		Formatting		Properties	
X	✓	1	If_single_condition = If('DATE'[index changed]<=4,TRUE(),FALSE())		
index changed	Index.1	If_single_condition	If_multiple_condition	Day Name	
0	0	True		1 Thursday	
1	2	True		1 Friday	
4	4	True		1 Monday	
6	6	False		0 Monday	
4	8	True		1 Monday	
5	10	False		0 Wednesday	

- **FALSE**: Returns the logical value FALSE.

Syntax: `FALSE()`

Dax Function:

`If_single_condition = If('DATE'[index changed]<=4,TRUE(),FALSE())`

Ex)

Structure		Formatting		Properties	
X	✓	1	If_single_condition = If('DATE'[index changed]<=4,TRUE(),FALSE())		
index changed	Index.1	If_single_condition		If_multiple_condition	Day Name
0	0	True		1	Thursday
1	2	True		1	Friday
4	4	True		1	Monday
6	6	False		0	Monday
4	8	True		1	Monday
5	10	False		0	Wednesday

- **COALESCE**: Returns the first expression that does not evaluate to BLANK. If all expressions evaluate to BLANK, BLANK is returned.

Syntax: COALESCE(<expression>, <expression>[, <expression>]...)

Dax Function:

Coalesce = COALESCE('COALESCE'[ID],'COALESCE'[ID1])

Ex)

Structure		Formatting		Properties	
X	✓	1	Coalesce = COALESCE('COALESCE'[ID],'COALESCE'[ID1])		
ID	ID1	Coalesce	Coalesce1	Convert	Currency Divide
1		1	1	₹ 1	₹ 1
0	2	0	0	₹ 0	₹ 0
3	0	3	3	₹ 3	₹ 3
			1		₹ 1
2	4	2	2	₹ 2	₹ 2

Note: Here, **coalesce** function first checks values in the first column. If it is not null, then it will return that value, if it is null then **coalesce** will check the corresponding first value in the second selected column.

If it is null, then it will return null value, if it is not null then it will return that column value as result. Like this **coalesce** will work.

If you want to substitute a default value in place of blank in coalesce result column then you can give like below. Here 1 is the default value, will be replaced in place of null or blank.

Dax Function:

Coalesce1 = Coalesce('COALESCE'[ID],'COALESCE'[ID1],1)

		Structure		Formatting	
X	✓	1	Coalesce1 = Coalesce('COALESCE'[ID], 'COALESCE'[ID1], 1)		
ID	ID1	Coalesce	Coalesce1	Convert	Currency
1		1	1	₹ 1	₹ 1
0	2	0	0	₹ 0	₹ 0
3	0	3	3	₹ 3	₹ 3
			1		₹ 1
2	4	2	2	₹ 2	₹ 2

Math and Trig Functions

The mathematical functions in Data Analysis Expressions (DAX) are very similar to the Excel mathematical and trigonometric functions. This section lists the mathematical functions provided by DAX.

- **CONVERT:** Converts an expression of one data type to another.

Syntax:

CONVERT(<Expression>, <Datatype>)

Dax Function:

Convert = Convert('COALESCE'[Coalesce],CURRENCY)

Ex)

<table border="1"> <tr> <td>Name</td><td>Convert</td><td>\$% Format</td><td>Currency</td><td>▼</td></tr> <tr> <td>123 Data type</td><td>Fixed decimal num...</td><td colspan="3" rowspan="3">CONVERT(Expression, DataType)</td></tr> <tr> <td colspan="5">Structure</td></tr> <tr> <td colspan="5">Convert an expression to the specified data type.</td></tr> </table>	Name	Convert	\$% Format	Currency	▼	123 Data type	Fixed decimal num...	CONVERT(Expression, DataType)			Structure					Convert an expression to the specified data type.					<table border="1"> <tr> <td>Name</td><td>Convert</td><td>\$% Format</td><td>Currency</td><td>▼</td></tr> <tr> <td>123 Data type</td><td>Fixed decimal num...</td><td>\$ ▶ %</td><td>₹ 0.00</td><td>Auto</td></tr> <tr> <td colspan="5">Structure</td></tr> <tr> <td colspan="5">Formatting</td></tr> </table>	Name	Convert	\$% Format	Currency	▼	123 Data type	Fixed decimal num...	\$ ▶ %	₹ 0.00	Auto	Structure					Formatting																																												
Name	Convert	\$% Format	Currency	▼																																																																													
123 Data type	Fixed decimal num...	CONVERT(Expression, DataType)																																																																															
Structure																																																																																	
Convert an expression to the specified data type.																																																																																	
Name	Convert	\$% Format	Currency	▼																																																																													
123 Data type	Fixed decimal num...	\$ ▶ %	₹ 0.00	Auto																																																																													
Structure																																																																																	
Formatting																																																																																	
<table border="1"> <tr> <td>X</td> <td>✓</td> <td>1</td> <td>Convert = Convert('COALESCE'[Coalesce],</td> <td>▼</td> </tr> <tr> <td>ID</td> <td>ID1</td> <td>Coalesce</td> <td>Coalesce1</td> <td>Convert</td> </tr> <tr> <td>1</td> <td></td> <td>1</td> <td>1</td> <td>Boolean</td> </tr> <tr> <td>0</td> <td>2</td> <td>0</td> <td>0</td> <td>CURRENCY</td> </tr> <tr> <td>3</td> <td>0</td> <td>3</td> <td>3</td> <td>DATETIME</td> </tr> <tr> <td></td> <td></td> <td></td> <td>1</td> <td>DOUBLE</td> </tr> <tr> <td>2</td> <td>4</td> <td>2</td> <td>2</td> <td>INTEGER</td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> <td>STRING</td> </tr> </table>	X	✓	1	Convert = Convert('COALESCE'[Coalesce],	▼	ID	ID1	Coalesce	Coalesce1	Convert	1		1	1	Boolean	0	2	0	0	CURRENCY	3	0	3	3	DATETIME				1	DOUBLE	2	4	2	2	INTEGER					STRING	<table border="1"> <tr> <td>X</td> <td>✓</td> <td>1</td> <td>Convert = Convert('COALESCE'[Coalesce],CURRENCY)</td> <td>▼</td> </tr> <tr> <td>ID</td> <td>ID1</td> <td>Coalesce</td> <td>Coalesce1</td> <td>Convert</td> </tr> <tr> <td>1</td> <td></td> <td>1</td> <td>1</td> <td>₹1</td> </tr> <tr> <td>0</td> <td>2</td> <td>0</td> <td>0</td> <td>₹0</td> </tr> <tr> <td>3</td> <td>0</td> <td>3</td> <td>3</td> <td>₹3</td> </tr> <tr> <td></td> <td></td> <td></td> <td>1</td> <td>₹1</td> </tr> <tr> <td>2</td> <td>4</td> <td>2</td> <td>2</td> <td>₹2</td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> <td>₹2</td> </tr> </table>	X	✓	1	Convert = Convert('COALESCE'[Coalesce],CURRENCY)	▼	ID	ID1	Coalesce	Coalesce1	Convert	1		1	1	₹1	0	2	0	0	₹0	3	0	3	3	₹3				1	₹1	2	4	2	2	₹2					₹2
X	✓	1	Convert = Convert('COALESCE'[Coalesce],	▼																																																																													
ID	ID1	Coalesce	Coalesce1	Convert																																																																													
1		1	1	Boolean																																																																													
0	2	0	0	CURRENCY																																																																													
3	0	3	3	DATETIME																																																																													
			1	DOUBLE																																																																													
2	4	2	2	INTEGER																																																																													
				STRING																																																																													
X	✓	1	Convert = Convert('COALESCE'[Coalesce],CURRENCY)	▼																																																																													
ID	ID1	Coalesce	Coalesce1	Convert																																																																													
1		1	1	₹1																																																																													
0	2	0	0	₹0																																																																													
3	0	3	3	₹3																																																																													
			1	₹1																																																																													
2	4	2	2	₹2																																																																													
				₹2																																																																													

Note: You can change the data type of anything to either Boolean, currency, Datetime, Double, Integer, String.

- **CURRENCY:** Evaluates the argument and returns the result as currency data type.

Syntax:

CURRENCY(<value>)

Dax Function:

Currency = Currency('COALESCE'[Coalesce1])

Ex)

ID	ID1	Coalesce	Coalesce1	Convert	Currency
1		1	1	₹ 1	₹ 1
0	2	0	0	₹ 0	₹ 0
3	0	3	3	₹ 3	₹ 3
			1		₹ 1
2	4	2	2	₹ 2	₹ 2

Note: By default power bi will assign currency symbol based on the location in regional settings. Suppose if your system location is in “USA”, then it will show you “\$” symbol.

To see the location in Regional settings

Navigation: Go to “File” ribbon, Select “Options and settings” option and click on “Options”. Go to “Current File” section and select “Regional Settings”. Then you can see “Locale for import” as “English(India)”



Options

- GLOBAL
 - Data Load
 - Power Query Editor
 - DirectQuery
 - R scripting
 - Python scripting
 - Security
 - Privacy
 - Regional Settings
 - Updates
 - Usage Data
 - Diagnostics
 - Preview features
 - Auto recovery
 - Report settings
- CURRENT FILE
 - Data Load
 - Regional Settings
 - Privacy

Locale for import
Locale determines the regional settings used to interpret numbers, dates, and time in imported text for this file.
English (India)

➤ **DIVIDE:** Performs division and returns alternate result or BLANK() on division by 0. That is

Suppose if you want to do division on two values, simply you can do like below.

Division = 'COALESCE'[ID1] / 'COALESCE'[ID]

Result:

ID	ID1	Coalesce	Coalesce1	Convert	Currency	Divide	Division
1		1	1	₹ 1	₹ 1		
0	2	0	0	₹ 0	₹ 0	0	∞
3	0	3	3	₹ 3	₹ 3	0	0
			1		₹ 1		
2	4	2	2	₹ 2	₹ 2	2	2

Note: Here, division returns a “infinity” value (∞), but this value is not understood by business users, instead of (∞) symbol, we need to give some default value. So to do this, we have to use “DIVIDE” function.

Syntax:

DIVIDE(<numerator>, <denominator> [,<alternateresult>])

Dax Function:

Divide = Divide('COALESCE'[ID1], 'COALESCE'[ID])

Ex)

ID	ID1	Coalesce	Coalesce1	Convert	Currency	Divide	Division
1		1	1	₹ 1	₹ 1		
0	2	0	0	₹ 0	₹ 0		
3	0	3	3	₹ 3	₹ 3	0	
			1		₹ 1		
2	4	2	2	₹ 2	₹ 2	2	2

Note: Divide function returns an empty value if something is divided by “0”. If you want, you can put the alternate value in that place. In the below example, 100 is the alternate value.

ID	ID1	Coalesce	Coalesce1	Convert	Currency	Divide	Division
1		1	1	₹ 1	₹ 1		
0	2	0	0	₹ 0	₹ 0	100	
3	0	3	3	₹ 3	₹ 3	0	
			1		₹ 1		
2	4	2	2	₹ 2	₹ 2	2	2

- **EVEN:** Suppose if **even** numbers are there in a column, then same value will be returned. In case if **odd** numbers are there, then next nearest **even** number will be returned.

Note: Returns number rounded up to the nearest **even** integer

Syntax:

EVEN(number)

Dax Function:

Even = **EVEN('COALESCE'[Coalesce1])**

Ex)

		Structure		Formatting		Properties	
X	✓	1	Even = Even('COALESCE'[Coalesce1])				
ID	ID1	Coalesce	Coalesce1	Convert	Currency	Divide	Division
1		1	1	₹ 1	₹ 1		2
0	2	0	0	₹ 0	₹ 0	100	∞
3	0	3	3	₹ 3	₹ 3	0	0
			1		₹ 1		2
2	4	2	2	₹ 2	₹ 2	2	2

- **ODD:** Suppose if **odd** numbers are there in a column, then same value will be returned. In case if **even** numbers are there, then next nearest **odd** number will be returned.

Note: Returns number rounded up to the nearest **odd** integer

Syntax:

ODD(number)

Dax Function:

Odd = **ODD('COALESCE'[Coalesce1])**

Ex)

		Structure		Formatting		Properties	
X	✓	1	Odd = Odd('COALESCE'[Coalesce1])				
ID	ID1	Coalesce	Coalesce1	Convert	Currency	Divide	Division
1		1	1	₹ 1	₹ 1		2
0	2	0	0	₹ 0	₹ 0	100	∞
3	0	3	3	₹ 3	₹ 3	0	0
			1		₹ 1		2
2	4	2	2	₹ 2	₹ 2	2	2

Extracting value before decimal point:

Note: Suppose, if you have decimal value, but need to extract without decimal place, then have to use “0” in place of “Auto” in column tools. But here rounding will be done based on the values after the decimal place.

But we have two functions like INT and TRUNC to display values before the decimal point.

The screenshot shows the Power BI Data Editor interface. At the top, there's a ribbon with tabs like 'Name', 'Format', 'Summarization', 'Properties', 'Sort by column', and 'Data groups'. Below the ribbon is a table with several columns: 'Round', 'Round - Copy', 'Int', 'Trunc', 'Mod', 'Rand', 'Round1', 'Blank', 'Divide', 'mod1', and 'Division'. The 'Division' column is highlighted with a yellow border. The 'Int' and 'Trunc' columns are also highlighted with yellow borders. The 'Format' tab is selected, showing settings for 'Decimal number' (set to 0) and 'Number format' (\$, #, #). The table contains data rows such as 1.52, 4.5678, 3.21456, etc., with their corresponding integer or truncated values in the adjacent columns.

- **INT:** Rounds a number down to the nearest integer.

Syntax:

`INT(<number>)`

Dax Function:

Int = `Int(ROUNDING[Round])`

Ex)

2.535---->2

-2.535--> -3 (nearest lower integer will come)

This screenshot shows the Power BI Data Editor with a DAX formula being entered into the formula bar: `Int = Int(ROUNDING[Round])`. The formula is highlighted with a green border. Below the formula bar is a table with columns: 'Round', 'Round - Copy', 'Int', 'Trunc', 'Mod', and 'Rand'. The 'Int' column is highlighted with a yellow border. The 'Round' column is also highlighted with a green border. The table contains data rows such as 1.52, 4.5678, 3.21456, etc., with their corresponding integer values in the 'Int' column.

Note: We can't display values after decimal point in INT function. But we can display decimal values also in TRUNC function.

- **TRUNC:** Truncates a number to an integer by removing the decimal, or fractional, part of the number.

Syntax:

TRUNC(<number>,<num_digits>)

Dax Function:

Trunc = Trunc(ROUNDING[Round])

Ex)

Structure		Formatting			
X	✓	1	Trunc = Trunc(ROUNDING[Round])		
Round	Round - Copy	Int	Trunc	Mod	
1.52	2	1	1	1	
4.5678	5	4	4	1	
3.21456	3	3	3	1	
5.556	6	5	5	1	
8.4555	8	8	8	1	

2.535----→2

-2.535--→ -2 (Nearest upper integer will come)

Note: INT and TRUNC both are same in functionality but in TRUNC you can display values after decimal point also.

Structure		Formatting			
X	✓	1	Trunc = Trunc(ROUNDING[Round],2)		
Round	Round - Copy	Int	Trunc	Mod	Rand
1.52	2	1	1.52	1	0.563901774585247
4.5678	5	4	4.56	1	0.971193734556437
3.21456	3	3	3.21	1	0.148052084259689
5.556	6	5	5.55	1	0.14235014282167
8.4555	8	8	8.45	1	0.212524273432791

- **MOD:** Returns the remainder after a number is divided by a divisor. The result always has the same sign as the divisor.

Syntax:

MOD(<number>,<divisor>)

Dax Function:

Mod = Mod(ROUNDING[Int])

Mod = Mod(ROUNDING[Int],2)

Mod = Mod(7,2) --→ 1

Mod = Mod(-3,-2) --→ -1 (Takes sign of the divisor)

Ex)

		Structure			Formatting	
X	✓	1	Mod = Mod(ROUNDING[Int],2)			
Round	Round - Copy	Int	Trunc	Mod		
1.52	2	1	1.52	1	0	
4.5678	5	4	4.56	0	0	
3.21456	3	3	3.21	1	0	
5.556	6	5	5.55	1	0	
8.4555	8	8	8.45	0	0	

Note: MOD function calculated the result value based on the below formula

$$\text{MOD}(n, d) = n - d * \text{INT}(n/d)$$

- **RAND:** Returns a random number greater than or equal to 0 and less than 1. Each and every time unique values will be generated.

Syntax:

RAND()

Dax Function:

Rand = Rand()

Ex)

		Structure			Formatting	
X	✓	1	Rand = Rand()			
Round	Round - Copy	Int	Trunc	Mod	Rand	
1.52	2	1	1.52	1	0.243043427588418	
4.5678	5	4	4.56	0	0.450349072227255	
3.21456	3	3	3.21	1	0.109005946898833	
5.556	6	5	5.55	1	0.751525596482679	
8.4555	8	8	8.45	0	0.430293577024713	

Note: Each and every time these random values will be changed while you refresh the data or the report. If you want to refresh the data set, right click on the data set and click on refresh.

- **ROUND:** Rounds a number to the specified number of digits.

Syntax:

ROUND(<number>, <num_digits>)

Dax Function:

Round1 = Round(ROUNDING[Round],2)

Ex)

Structure		Formatting	
X	✓	1	Round1 = Round(ROUNDING[Round],2)
Round	Round - Copy	Int	Trunc
1.52	2	1	1.52
4.5678	5	4	4.56
3.21456	3	3	3.21
5.556	6	5	5.55
8.4555	8	8	8.45
			Rand
			Round1
			1.52
			4.57
			3.21
			5.56
			8.46

Note: Based on the number of digits you have given in the second argument, round function will give the results.

Other Functions

These functions perform unique actions that cannot be defined by any of the categories.

- **BLANK**: Returns a blank.

Syntax:

BLANK()

Dax Function:

Blank = If(ROUNDING[Int]=3,Blank(),ROUNDING[Int])

Blank1 = If(ROUNDING[Blank]=BLANK(),1,ROUNDING[Int])

Ex)

Structure		Formatting		Properties	
X	✓	1	Blank = If(ROUNDING[Int]=3,Blank(),ROUNDING[Int])		
Round	Round - Copy	Int	Trunc	Mod	Rand
1.52	2	1	1.52	1	0.243043427588418
4.5678	5	4	4.56	0	0.450349072227255
3.21456	3	3	3.21	1	0.109005946898833
5.556	6	5	5.55	1	0.751525596482679
8.4555	8	8	8.45	0	0.430293577024713
			Round1		Blank
					1
					4
					3.21
					5.56
					8

- **ERROR**: Raises an error with an error message.

Syntax:

ERROR(<text>)

Dax Function:

```
Error =  
Var v1=SELECTEDVALUE(ROUNDING[Int])  
Var v2=If(v1=3,Error("This is Three"),v1)  
Return V2
```

Note: Most of times **Error** function will use in “Measure”.

Variable: Variables are useful at the time of runtime. At the time of writing function, sometimes function will return outputs, that output will be stored in the variable.

If you want to see the error message, then you have to take a slicer visual and card visual.

First take the **slicer** visual, put the “INT” column in the “fields” box. Arrange values in the proper order in slicer visual.

Then take the **card** visual and put the measure in the “fields” box.

Select any value in the **slicer** visual that value will be displayed in **card** visual as well as per the above measure formula.

But if you select value “3” then it will through an error in the card. See the below screenshot.



Relationship Functions

These functions are useful for managing and utilizing relationships between tables.

Note: If you want to get the columns from another data set then we have to use “RELATED” and “LOOKUPVALUE”. If you want to apply these functions based on the below rules.

Related:

- ✓ Relationship is required between the data sets(Relationships will be given in the “**Model**” tab in power bi desktop).
- ✓ Cardinality should be any one of the below.
1-1, 1-*, *-1

Lookupvalue:

- ✓ Relationship is not required between the data sets

➤ **RELATED:** Returns a related value from another table.

Syntax: RELATED(<column>)

Dax Function:

Related = RELATED(Dept1[Deptname])

Ex)

The screenshot shows the Power BI Data View interface. In the top bar, there are tabs for 'Structure', 'Formatting', and 'Properties'. The 'Properties' tab is active, showing a formula bar with the text '1 Related = RELATED(Dept1[Deptname])'. Below the formula bar is a table with three rows of data: Empid (1001, 1002, 1003), Empname (SURESH, VENKATESH, RAJESH), Deptno (101, 102, 103), and a column labeled 'Related' which contains the values Sales, Service, and Support respectively. To the right of the table is a 'Fields' pane with a search bar containing 'relat' and a list of available fields: 'Emp1' (selected) and 'Related'. A green box highlights the formula bar and the 'Related' dropdown in the table columns.

Note: Here I am getting “Deptname” column from “Dept1” table to the “Emp1” table.

Process and Rules need to check:

Table Name: Emp1

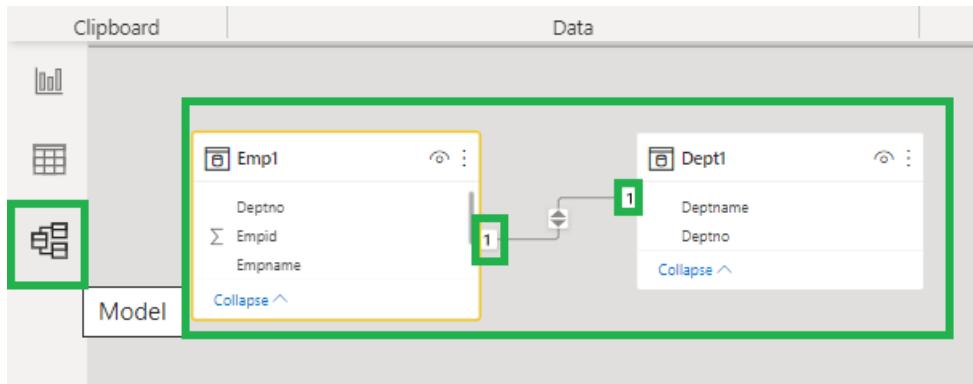
Empid	Empname	Deptno
1001	SURESH	101
1002	VENKATESH	102
1003	RAJESH	103

Table Name: Dept1

Deptno	Deptname
101	Sales
102	Service
103	Support

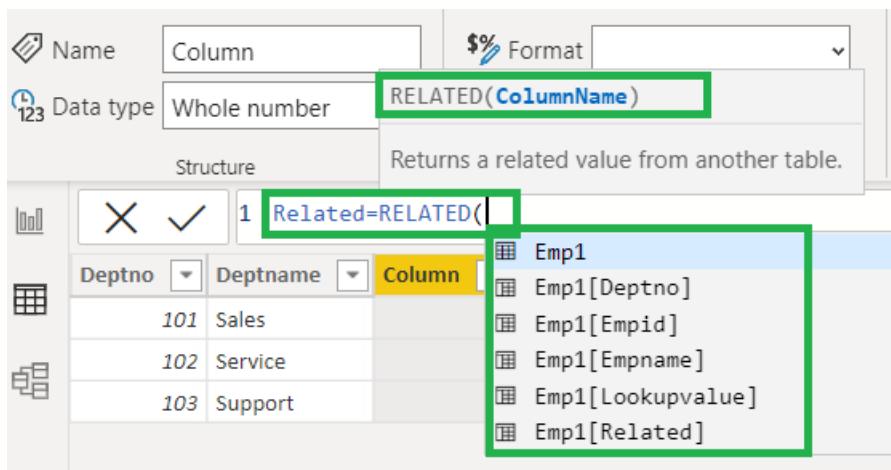
I want to get the column “Deptname” from “Dept1” table to the “Emp1” table.

So I have to check whether the relationship is there between the data sets or not. For that, go to “**Model**” tab and check.



Relationship is established between the data sets “**Emp1**” and “**Dept1**” table and also cardinality (1-1). So the rules of the **RELATED** function are satisfied. Then we can apply that function.

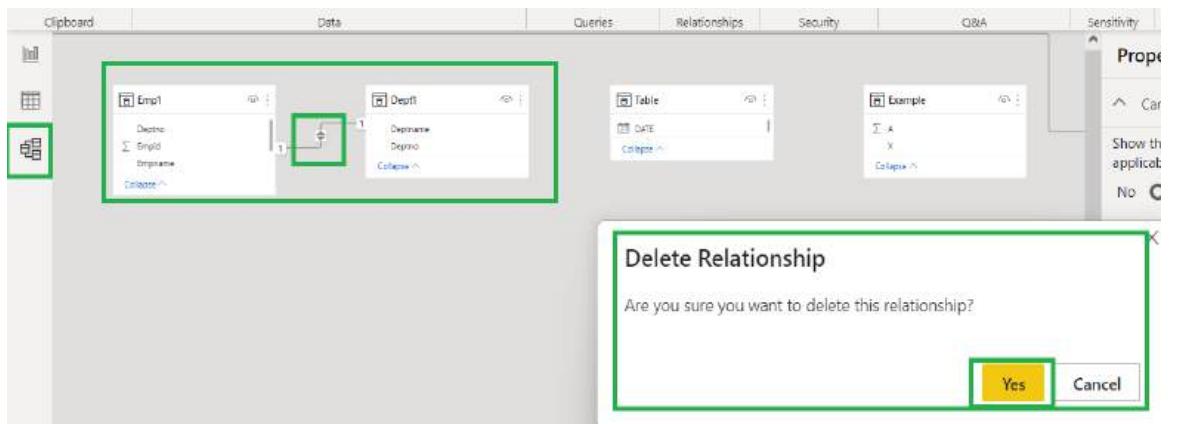
Note: If you see the below screenshot you can understand how the relationship is working.



Here, currently I am in “**Dept1**” table, I want the column “**Empname**” column from “**Emp1**” table. If you write related like this, then automatically that functions will only display the columns of the table having relationship with the current table.

Note2: If you delete relationship between the tables in the model then “Related” function throws an error message.

To remove relationship **right click** on the **relationship line** and click on “**Delete**”. Then it will ask whether you want to delete the relationship or not. Click on “**Yes**” button. Then the line will be removed between the data sets.



Structure		Formatting		Properties	
X	✓	1 Related = Related(Dept1[Deptname])	The column 'Dept1[Deptname]' either doesn't exist or doesn't have a relationship to any table available in the current context.		
Empid	Emplname	Deptno	Lookupvalue	Related	
1001	SURESH	101	Sales	#ERROR	
1002	VENKATESH	102	Service	#ERROR	
1003	RAJESH	103	Support	#ERROR	

Note:

- ✓ If you want to get the column from another table without having relationship between them, then you have to use “**LOOKUPVALUE**” function.
 - ✓ Lookupvalue will work even if the relationship is there between the two data sets.
- **LOOKUPVALUE:** To get the column from another table without having relationship with that table we can use this function.

Returns the value for the row that meets all criteria specified by one or more search conditions.

Syntax:

```
LookupValue= LOOKUPVALUE(
    <result_columnName>,
    <search_columnName>,
    <search_value>
        [, <search2_columnName>, <search2_value>]...
        [, <alternateResult>]
    )
)
```

Dax Function:

```
Lookupvalue =
LOOKUPVALUE(Dept1[Deptname],Dept1[Deptno],Emp1[Deptno])
```

Note: Here we have passed three arguments

- 1) Required column
- 2) Common column from the target table

3) Common column from the source table.

Ex)

The screenshot shows the Power BI Data Editor interface. A new column 'Lookupvalue' is being created, as indicated by the formula bar which contains the DAX code: `1 Lookupvalue = LOOKUPVALUE(Dept1[Deptname],Dept1[Deptno],Emp1[Deptno])`. The table has columns Empid, Empname, Deptno, Lookupvalue, and Related. The 'Lookupvalue' column is highlighted with a yellow background and a green border. The data in this column is derived from the 'Dept1' table based on the 'Deptno' value from the current row and the 'Deptno' value from the 'Emp1' table.

Empid	Empname	Deptno	Lookupvalue	Related
1001	SURESH	101	Sales	Sales
1002	VENKATESH	102	Service	Service
1003	RAJESH	103	Support	Support

- **USERELATIONSHIP:** Specifies the relationship to be used in a specific calculation as the one that exists between columnName1 and columnName2.

Syntax:

`USERELATIONSHIP(<columnName1>,<columnName2>)`

Note: This function will be used in only “CALCULATE” function only. If the relationship is inactive between two tables, then this function makes it active. We will discuss more in “Model”.

Table Manipulation Functions

These functions return a table or manipulate existing tables. All these functions will be applied only in “New Table” option only. That means, by using these functions new data sets will be created.

- **ADDCOLUMNS:** Adds calculated columns to the given table or table expression.

Syntax: `ADDCOLUMNS(<table>, <name>, <expression>[, <name>, <expression>]...)`

Dax Function:

AddColumns = `ADDCOLUMNS(Dept1,"Sales",IF(Dept1[Deptno]>100,200,100))`

Dept1→ Existing Table Name

Sales→ Desired column name in new table

Condition→ Based on this, values will be inserted in the new column.

Ex)

Existing table: Dept1

Deptno	Deptname
101	Sales
102	Service
103	Support

- **CROSSJOIN:** Returns a table that contains the Cartesian product of all rows from all tables in the arguments. The columns in the new table are all the columns in all the argument tables.

Note: All the columns must be different in all the tables for using this function. Otherwise it will throw an error. **Generate** function will do the same work as well.

Syntax: CROSSJOIN(<table>, <table>[, <table>]...)

Dax Function:

CrossJoin = CROSSJOIN(Aggregation,Append1)

Ex)

NUMBER	VALUE	DATE	RATE
1	1	11-05-2022	5
2	2	10-09-2021	2
1	4	25-02-2020	
2	4	12-04-1998	
5	2	08-07-2021	5
2	1	11-05-2022	5
3	2	10-09-2021	2
4	4	25-02-2020	
2	4	12-04-1998	
5	2	08-07-2021	5
2	1	11-05-2022	5
3	2	10-09-2021	2
4	4	25-02-2020	
2	4	12-04-1998	
5	2	08-07-2021	5
2	1	11-05-2022	5
3	2	10-09-2021	2
4	4	25-02-2020	
2	4	12-04-1998	
5	2	08-07-2021	5

Id	Name
1	AA
2	BB
3	CC
4	DD

- **DATABLE:** To create a new table, we can use this function. We can pass the values at the time of creation of the table.

Syntax:

```
DATATABLE(Column Name1, Data Type1, Column Name2, Data Type2..., {{Value1, Value2...}, {ValueN, ValueN+1...}...})
```

Dax Function:

```
DataTable = DATATABLE("Id", INTEGER, "Name", STRING, "Salary", CURRENCY,  
{  
    {1001, "Vasu", 10000},  
    {1002, "Venu", 12000},  
    {1003, "Veda", 15000}  
})
```

Note: Here, column names and their values should be given in double quotes ("") only. Whatever the order we have given the column names, in the same order we have to put the values as well.

In transform data, we can use “Enter Data” option to create a new data set, but here, by using “DATATABLE” function we can create a new table.

Ex)

The screenshot shows the Power BI Data Editor interface. In the top ribbon, the 'New table' button is highlighted with a green box. The 'Structure' pane on the left contains the DAX code:

```
1 DataTable = DATATABLE("Id", INTEGER, "Name", STRING, "Salary", CURRENCY,  
2 {  
3     {1001, "Vasu", 10000},  
4     {1002, "Venu", 12000},  
5     {1003, "Veda", 15000}  
6 })
```

The 'Fields' pane on the right lists the table 'DataTable' under the search results. Below the code, a preview table shows three rows of data:

Id	Name	Salary
1001	Vasu	₹ 10,000
1002	Venu	₹ 12,000
1003	Veda	₹ 15,000

- **DISTINCT(Column):** Returns a one-column table that contains the distinct values from the specified column.

In other words, duplicate values are removed and only unique values are returned.

We will not use DISTINCT function basically on table, we will apply it on a column only.

Syntax:

```
DISTINCT(<column>)
```

Dax Function:

Distinct_Table = DISTINCT('DATE'[MonthName])

Ex)

The screenshot shows the Power BI Data Editor interface. In the top ribbon, the 'Table tools' tab is selected. A new table named 'Distinct_Table' is being created, as indicated by the text 'Distinct_Table = DISTINCT('DATE'[MonthName])' in the formula bar. The 'Fields' pane on the right shows the newly created table 'Distinct_Table'. The table contains one column named 'MonthName' with the following data: September, May, March, February.

Note: In this function, we can't pass multiple columns as arguments; only one column will be accepted. In that column duplicate values will be removed and created a column in new table with the same name of that column.

- **EXCEPT:** Returns the rows of one table which do not appear in another table.

Note: Two tables must have same number of columns and data types.

Syntax:

EXCEPT(<table_expression1>, <table_expression2>)

Except1

Country
INDIA
CHINA
JAPAN
IRAN
AUSTRALIA
USA
UK

Except2

Country1
USA
UK
AUSTRALIA

Dax Function:

Except = EXCEPT(Except1, Except2)

Ex)

The screenshot shows the Power BI Data Editor interface. A new table named 'Except' is being created, as indicated by the text 'Except = EXCEPT(Except1, Except2)' in the formula bar. The 'Fields' pane on the right shows the newly created table 'Except'. The table contains one column named 'Country' with the following data: INDIA, CHINA, JAPAN, IRAN.

- **GENERATESERIES:** To generate sequential values, we can use this function. It will create a new data set and name of column will be "Value"

Syntax:

GENERATESERIES(<startValue>, <endValue>[, <incrementValue>])

Dax Function:

GenerateSeries = GENERATESERIES(101,110,1)

Ex)

Value
101
102
103
104
105
106
107
108
109
110

- **INTERSECT**: Only matching records will be returned and creates a new data set with those records.

Syntax:

INTERSECT(<table_expression1>, <table_expression2>)

Note: That means a table that contains all the rows in table_expression1 that are also in table_expression2. It is same as inner join in the transform data.

Except1

Country
INDIA
CHINA
JAPAN
IRAN
AUSTRALIA
USA
UK

Except2

Country1
USA
UK
AUSTRALIA

Dax Function:

Intersect = INTERSECT(Except1,Except2)

Ex)

Country
AUSTRALIA
USA
UK

- **SUMMARIZE**: Returns a summary table for the requested totals over a set of groups.

Syntax:

```
SUMMARIZE(<table>,<groupBy_columnName>[, <groupBy_columnName>]...[, <name>, <expression>]...)
```

Dax Function:

Summarize = **SUMMARIZE**(**financials**, **financials[Country]**, **financials[Segment]**, "Total_Sales",**sum(financials[Sales])**, "Total_Profit",**sum(financials[Profit])**)

Ex)

The screenshot shows the Power BI Data View interface. A calculated column named "Summarize" is displayed in the Fields pane. The formula for this column is highlighted with a green box: `= SUMMARIZE(financials, financials[Country], financials[Segment], "Total_Sales", sum(financials[Sales]), "Total_Profit", sum(financials[Profit]))`. The main table view shows data grouped by Country and Segment, with columns for Total_Sales and Total_Profit.

- **GROUPBY**: The GROUPBY function is similar to the SUMMARIZE function

Syntax:

```
GROUPBY (<table> [, <groupBy_columnName> [, <groupBy_columnName> [, ...]]] [, <name>, <expression> [, <name>, <expression> [, ...]]])
```

Dax Function:

Groupby =
GROUPBY(**financials**,**financials[Country]**,**financials[Segment]**, "Total_sales",**sumx(CURRENTGROUP(),financials[Sales])**, "Total_Profit",**sumx(CURRENTGROUP(),financials[Profit])**)

Note: In this function you have to use **CURRENTGROUP** function and we have to use two parameter aggregation function.

Ex)

The screenshot shows the Power BI Data View interface. On the left is a table named 'financials_Country' with columns: financials_Country, financials_Segment, Total_sales, and Total_Profit. The data includes rows for various countries and segments like Government, Midmarket, Enterprise, and Channel Partners, with their respective sales and profit values. On the right, there is a 'Fields' pane containing a 'Groupby' entry. The 'Groupby' entry has a tooltip showing the DAX code: `1 Groupby = GROUPBY(financials,financials[Country],financials[Segment], "Total_sales",sumx(CURRENTGROUP(),financials[Sales]),"Total_Profit",sumx(CURRENTGROUP(),financials[Profit]))`. The 'Groupby' entry also lists the columns: financials_Country, financials_Segment, Σ Total_Profit, and Σ Total_sales.

- **UNION:** To combine rows of multiple datasets then we can use this function. In transform, for this same thing, we will use “**Append Queries**” option. Whatever the rules we have for “Append Queries” will be applied here as well.

Rules: Column names and data types must be same in all the tables and number of columns should be same in **UNION**.

Syntax:

`UNION(<table_expression1>,<table_expression2> [,<table_expression>]...)`

Dax Function:

`Union = UNION(Append1,Append2,Append3)`

Ex)

<u>Append1</u>		<u>Append2</u>		<u>Append3</u>	
Id	Name	Id	Name	Id	Name
1	AA	5	FF	7	ZZ
2	BB	6	GG		
3	CC				
4	DD				

Result:

The screenshot shows the Power BI Data Editor interface. At the top, there's a ribbon with tabs like File, Home, Help, and Table tools. Under Table tools, there are several icons: 'Mark as date table', 'Calendars', 'Manage relationships', 'New measure', 'Quick measure', 'New column', and 'Calculations'. The 'New table' icon is highlighted with a green box. Below the ribbon, the title bar says 'Union'. The main area shows a table with columns 'Id' and 'Name'. The data is as follows:

Id	Name
1	AA
2	BB
3	CC
4	DD
5	FF
6	GG
7	ZZ

- **ROW:** To insert a row for the existing table we have to use **ROW** function. Or else you can create a new table with single row.

Syntax:

ROW(<name>, <expression>[,<name>, <expression>]...])

Dax Function:

Row1 = **ROW("Name", "Sudheer", "DOB", "1990-04-15", "Location", "Vizag")**

The screenshot shows the Power BI Data Editor interface. The title bar says 'Row1'. The formula bar at the top shows the DAX formula: 'Row1 = ROW("Name", "Sudheer", "DOB", "1990-04-15", "Location", "Vizag")'. Below the formula bar, there's a table with three columns: 'Name', 'DOB', and 'Location'. The data is as follows:

Name	DOB	Location
Sudheer	1990-04-15	Vizag

If you want to insert a new row in an existing table we have to use UNION along with ROW function.

Dax Function:

Row = **UNION('Union',ROW("Id",8,"Name","KK"))**

Ex)

The screenshot shows the Power BI Data Editor interface. The title bar says 'Row'. The formula bar at the top shows the DAX formula: 'Row = UNION('Union',ROW("Id",8,"Name","KK"))'. A tooltip appears below the formula bar showing the table structure: 'Name Row', 'Storage mode Import', and 'Data refreshed 10/20/2022, 6:14:09 PM'. On the right side, there's a 'Fields' pane with a tree view. The 'Row' node is selected and highlighted with a green box. The 'Id' column in the table preview is also highlighted with a green box. The table preview shows the following data:

Id	Name
1	AA
2	BB
3	CC
4	DD
5	FF
6	GG
7	ZZ
8	KK

Note: ROW function will create a new data set with the existing table columns by inserting new row values.

- **SELECTCOLUMNS**: Returns a table with selected columns from the table and new columns specified by the DAX expressions.

Syntax:

SELECTCOLUMNS(<Table>, [<Name>], <Expression>, <Name>, ...)

Note: Here, column names we can give as new names or you can give same names of the existing table.

We should give column names in double quotes only and select the respective column to display.

Dax Function:

SelectColumns = **SELECTCOLUMNS(financials,"Countries",financials[Country], "SegmentSection",financials[Segment])**

Ex)

Countries	SegmentSection
Germany	Government
Germany	Government
Canada	Government
Germany	Government
Germany	Government
France	Government
France	Government
Mexico	Government
Mexico	Government
France	Government
France	Government
Canada	Government
Mexico	Government
Canada	Government

- **VALUES**: It will work similar like distinct function. When the input parameter is a column name, returns a one-column table that contains the distinct values from the specified column. A BLANK value can be added.

Duplicate values are removed and only unique values are returned.

Syntax:

VALUES(<TableNameOrColumnName>)

Note: When the input parameter is a table name, returns the rows from the specified table. Duplicate rows are preserved. A BLANK row can be added.

Dax Function:

VALUES(<TableNameOrColumnName>)

Ex)

Table Name: Aggregation

The screenshot shows the Power BI Data Editor interface. A table named "Values" is open, containing four columns: NUMBER, VALUE, DATE, and RATE. The "Table tools" ribbon is at the top, with the "New table" button highlighted. The formula bar displays the formula "Values = VALUES(Aggregation[NUMBER])". The Fields pane on the right lists the table's structure, with "Values" selected.

Statistical Functions

Statistical functions calculate values related to statistical distributions and probability.

- **RANKX:** It will generate ranks for all the rows in the select column.

Syntax:

RANKX(<table>, <expression>[, <value>[, <order>[, <ties>]]])

Dax Function:

RankX = **RANKX(EMP,EMP[empid],,DESC,Dense)**

Ex)

The screenshot shows the Power BI Data Editor. The formula bar contains "RankX=RANKX(EMP,EMP[empid],DESC,Dense)". A context menu is open over the "Order" parameter, with "DESC" selected. Below the formula bar, the EMP table is shown with columns empid, ename, job, sal, and hiredate.

Suppose I am taking “**DESC**” values with “**DENSE**” ties. Then you can see below output.

Result:

Structure							Formatting
X ✓ 1 RankX = RANKX(EMP,EMP[empid],,DESC,Dense)							RankX
empid	ename	job	sal	hiredate		RankX	
100	SACHIN	CLERK	₹ 4,000	03 November 2021		5	
100	KUMAR	ANALYST	₹ 9,000	06 August 2022		5	
102	RAHUL	MANAGER	₹ 6,000	10 May 2020		4	
103	DAVID	CLERK	₹ 5,000	05 October 2020		3	
104	PHANI			06 August 2022		2	
105	VENKAT			15 June 2019		1	

Note: Dense and Skip are similar to “Dense Rank” and “Rank” functions in “SQL Server” respectively.

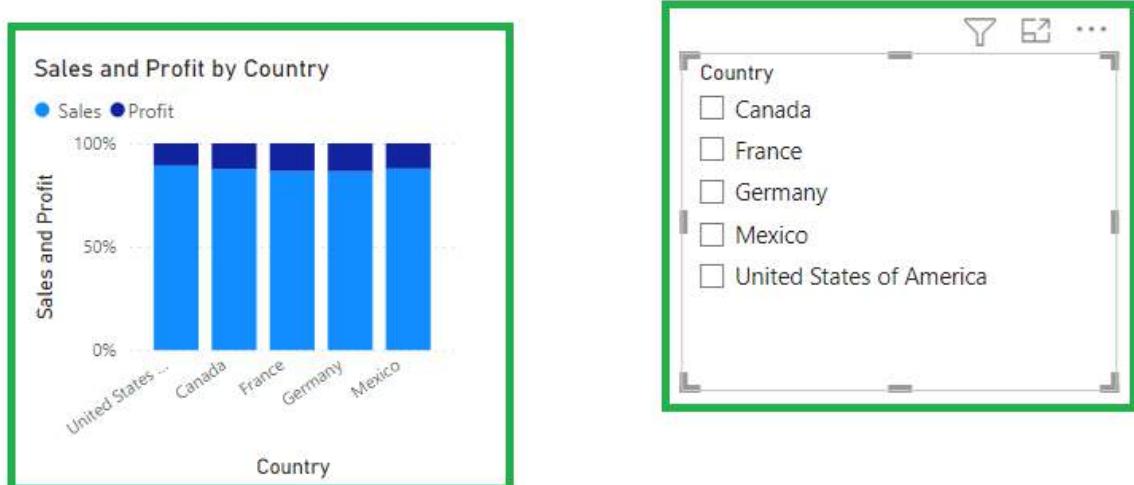
Filter Functions

Power BI has a feature of having default filter for each value. You can see in visuals. That means if you select any value in the slicer visual, then automatically that value will be filtered in other visuals as well.

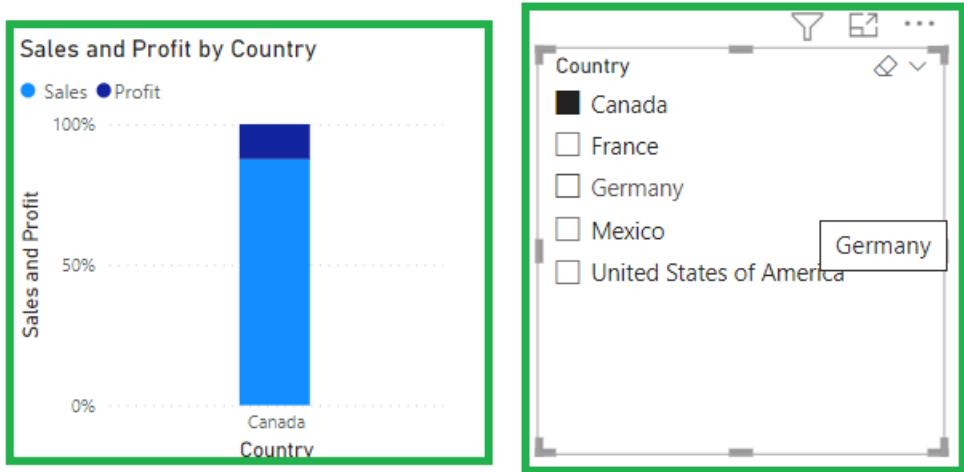
These are designed by using “Artificial Intelligence”. These are called as “Interactions”. We can manage them by using “Edit Interactions” option in “Format” tab.

Look at the below screen shots to understand better.

Before selecting a value: Here I have used “Stacked Column Chart” and “Slicer” visuals.



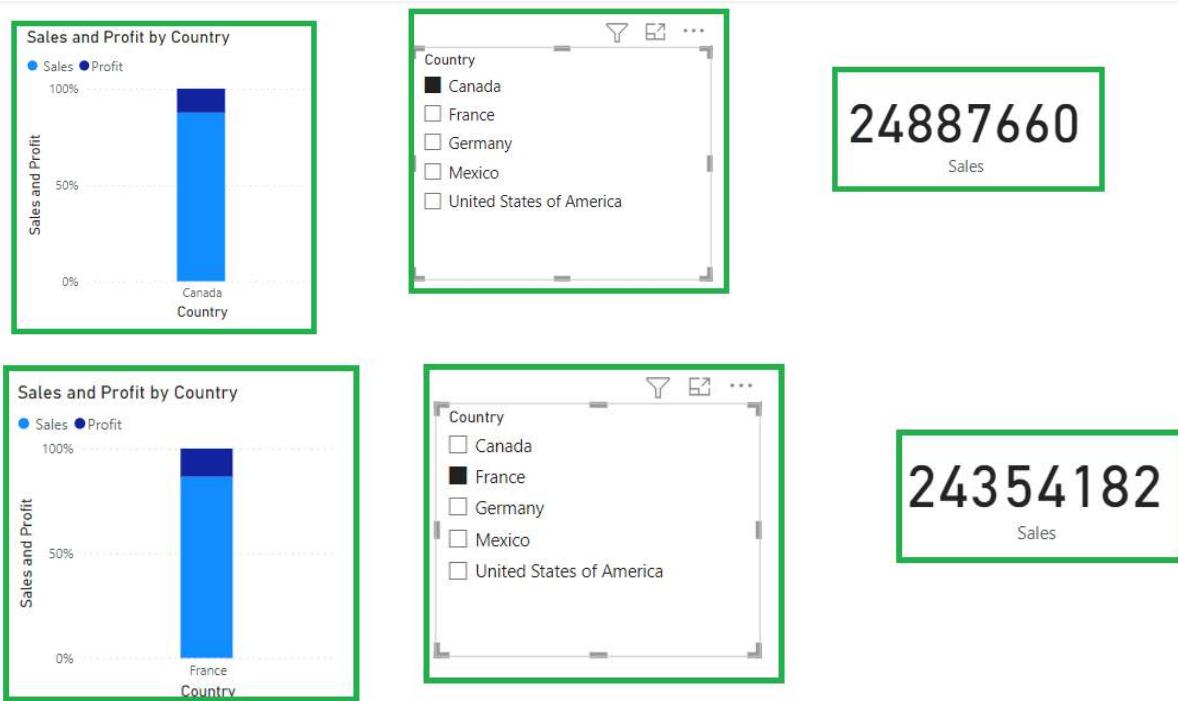
After selecting a value: Here I have selected “Canada” value. So the filter automatically applied on corresponding visual.



Note: If you select any value in any visual, then automatically corresponding value will be filtered in all the visuals in that page only. But other pages will not effect.

Generally, if you select any value in slicer visual that corresponding value should be filtered in all visuals of that page.

But if you take sum of sales value in “Card” visual, that value also changing unfortunately, but it should not be done, because we are not changing anything, we only just selecting the value right.



In the above screenshots, I have filtered “**Canada**” and “**France**” value; automatically corresponding values have been filtered in remaining visuals.

But in card visual, I have taken sum of sales from “**Financials**” data set. It is also changed according to the filter you applied. But it should not effect.

So to overcome these kinds of problems we can use “**Filter Functions**”. That means, if you don’t want to do filter on some visuals, then we can use “**Filter Functions**”.

Note: All the filter functions most probably will be used in “**New Measure**”.

- **ALL:** This function is useful for clearing filters and creating calculations on all the rows in a table.

Note: Always use this function in calculate function only. You can use this function to restrict the filter on selected columns or entire the table.

Syntax:

`ALL([<table> | <column>[, <column>[, <column>[,...]]]])`

Dax Function:

All_Filter = `CALCULATE(sum(financials[Sales]),
ALL(financials[Country],financials[Product]))`

Note: In the above example, I have applied filter on country column and product column. That means if you select any value in county and product columns in visuals, filters will be cleared these columns and the value will not change in the measure by selecting any value in these columns. See below screenshots.

Ex)

The screenshot shows the Power BI interface with three visualizations and the Fields pane:

- Visual 1:** A bar chart titled "Sales vs Profit" showing sales percentages for five countries: United States, Canada, France, Germany, and Mexico. The chart has "Country" on the x-axis and "Sales vs Profit" on the y-axis.
- Visual 2:** A single value card displaying "118726385" under the heading "Sales".
- Visual 3:** A single value card displaying "118726385" under the heading "Calculate".
- Visual 4:** A single value card displaying "118726385" under the heading "All_Filter".
- Fields Pane:** Shows the "Financials" table with various columns. The "Country" and "Product" columns are highlighted with green boxes. The "All_Filter" column is also highlighted with a green box. The "Sales" column is shown as a calculated column.

Note: Whatever the columns you have mentioned in the ALL functions, for those columns only filter will be cleared, remaining columns will be filtered.

If you don't want to filter on all columns in the table, then you have to pass the entire table name in ALL function. Then filters will be cleared for all the columns in that table.

Dax Function:

All_Filter = [CALCULATE\(sum\(financials\[Sales\]\),ALL\(financials\)\)](#)

Ex)

Note: In the above example I have passed table as argument for ALL function. That means filters will be cleared for all the columns existing in that table.

If you select any value in slicer visual for any column in that table, then measure value will not effect.

- **ALLCROSSFILTERED:** This function clears all filters which are applied to a table.

Note: Always use this function in calculate function only. You have to pass the table name as argument for this function. Otherwise it will throw an error.

Syntax:

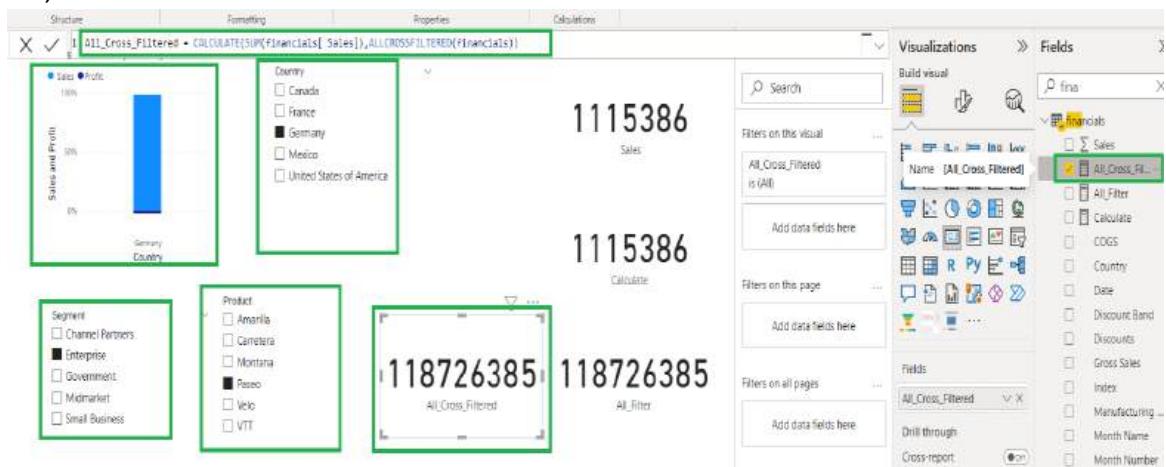
ALLCROSSFILTERED(<table>)

Dax Function:

All_Cross_Filtered = CALCULATE(SUM(financials[Sales]),
ALLCROSSFILTERED(financials))

Note: We can't clear filters on selected columns, filters will be cleared for all the columns in that table.

Ex)



- **ALLEXCEPT:** This function will apply the filters on the columns you have mentioned in the formula and filters will be cleared for all the rest of the columns in that table. This is quite opposite to **ALL** function.

Note: Always use this function in calculate function only. You can pass all the columns or selected columns in this function to apply the filters.

Syntax:

ALLEXCEPT(<table>,<column>[,<column>[,...]])

Dax Function:

All_Except = CALCULATE(sum(financials[Sales]),
ALLEXCEPT(financials,financials[Country],financials[Product]))

Note: First argument must be table name and then pass the list of columns to apply filters.

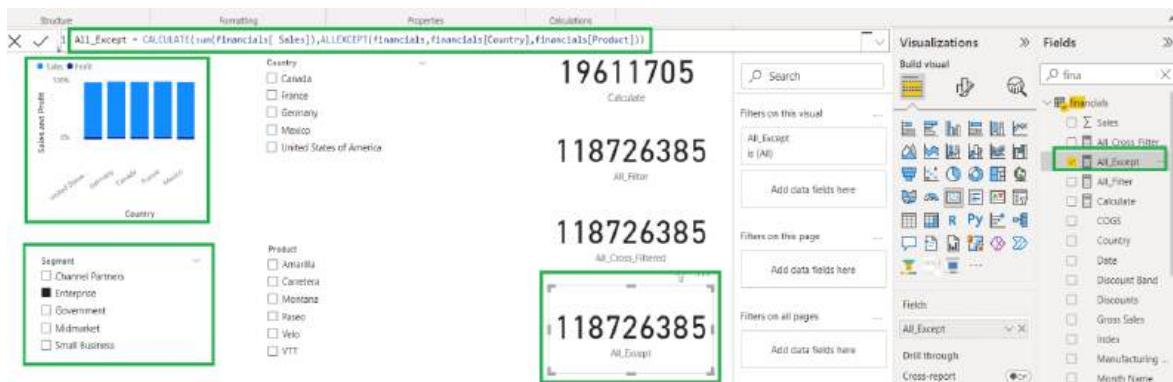
Ex)



In the above example I have applied filters on “**Country**” and “**Product**” columns. So, I have selected value in the “**Country**” and “**Product**” columns automatically measure value has been changed.

But if I select value from “**Segment**” column then measure value will not be effected. See below screen shot.

Ex)



- **ALLNOBLANKROW:** This function is useful for clearing filters and creating calculations on all the rows in a table. But it will not clear filters on blank values. That means filters will be applied on blank values.

Note: Always use this function in calculate function only. You can use this function to restrict the filter on selected columns or entire the table.

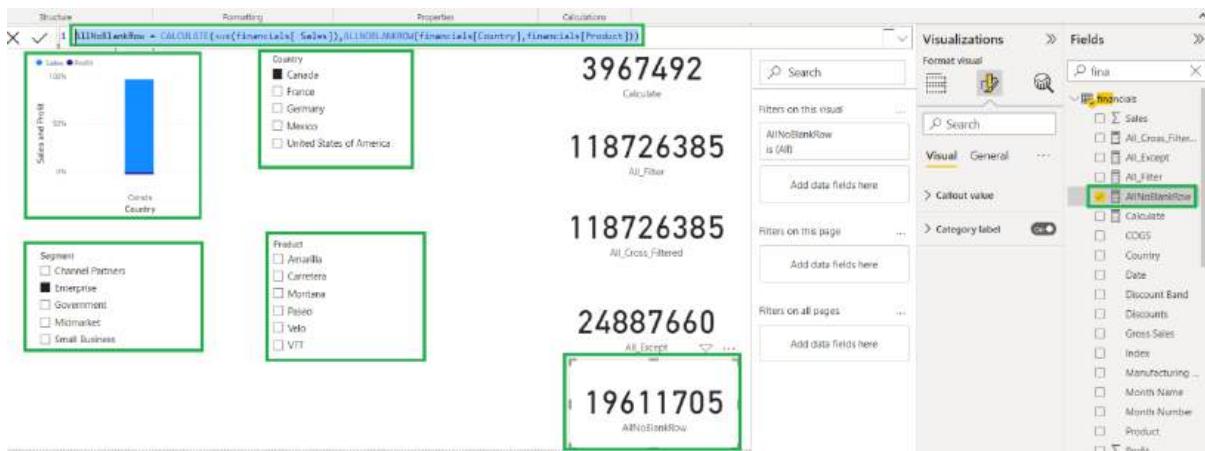
Syntax:

ALLNOBLANKROW([<table> | <column>[, <column>[, <column>[,...]]]])

Dax Function:

AllNoBlankRow = **CALCULATE(sum(financials[Sales]), ALLNOBLANKROW(financials[Country],financials[Product]))**

Ex)



In the above example, I have cleared filters on “Country” and “Product” columns but not applied on “Segment” column.

If we select value from either “Country” or “Product” columns then measure value will not effect, but if we select “Segment” column then measure value will be changed, because we haven’t cleared filter on this column.

- **ALLSELECTED**: This function will apply the filters on selected columns. That means it will act as a default behavior of power bi.

Main Point: **ALLSELECTED** function will ignore filters from inside query (inside visual) and keeps the filters from outside the query (outside visual).

Def: Returns all rows in a table or all values in a column, ignoring any filters that might have been applied inside the query, but keeping the filters come from outside.

Note: Always use this function in calculate function only. You can use this function to apply the filter on selected columns or entire the table.

Syntax:

`ALLSELECTED([<table> | <column>[, <column>[, <column>[,...]]]])`

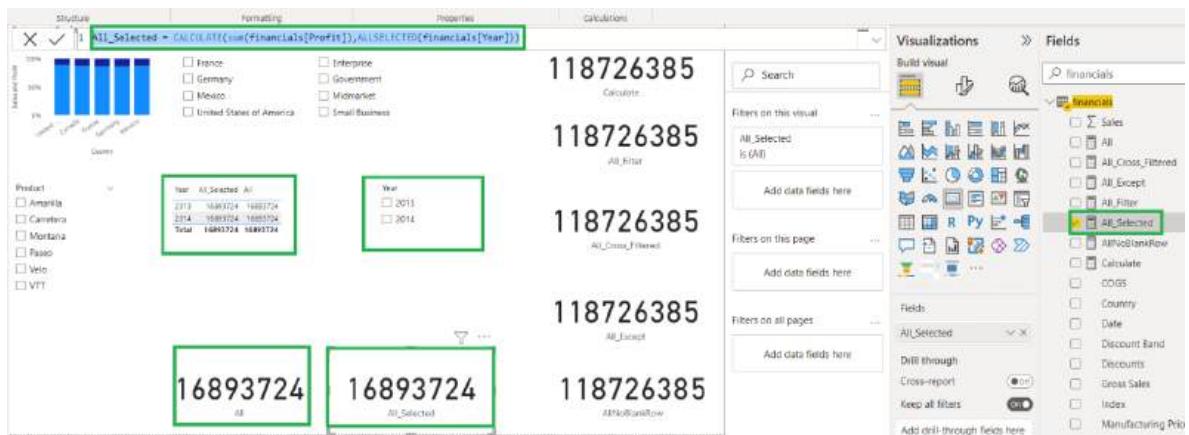
Dax Function:

All_Selected = `CALCULATE(sum(financials[Profit]),
ALLSELECTED(financials[Year]))`

I am taking **ALL** function also for comparing purpose along with **ALLSELECTED**

All = `CALCULATE(SUM(financials[Profit]),all(financials[Year]))`

Ex)



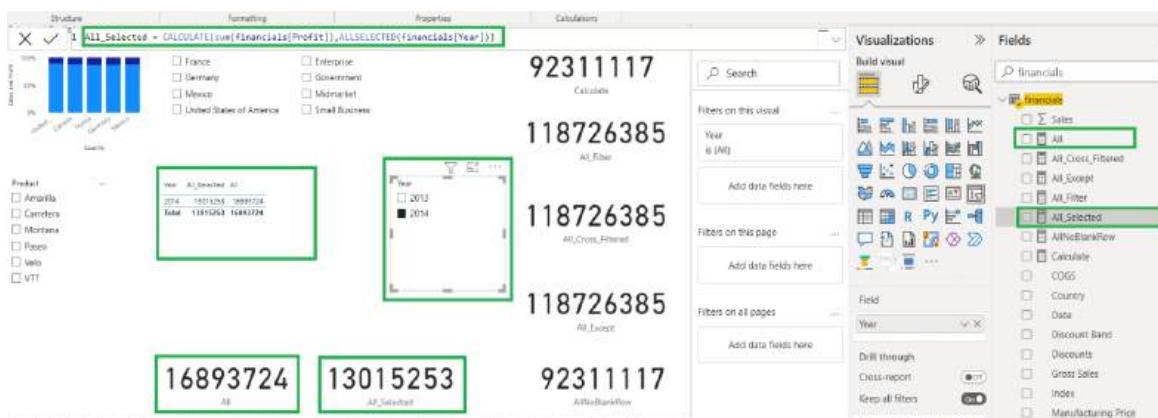
Note: In the above example, I am taking measures by calculating aggregation function on sum of profit in financials data set for **ALL** and **ALLSELECTED** functions.

If we see the measure values in the above screen shots, both are showing the same result (inside query). But if you take slicer visual and select any value then you can see the difference.

For comparing purpose, I have taken table visual and I put Year, All, Allselected columns. Here All and Allselected column values are same with the total profit.

If you want to see the difference in **ALL** and **ALLSELECTED** columns, take slicer visual, put the Year column in that slicer visual.

Then select any value in slicer visual, then “**ALLSELECTED**” column value is changed but “**ALL**” column value has not been changed (Outside Query). See the below screen shot.



- **CALCULATE:** If you want to calculate the values of aggregate functions based on expression, then we have to use calculate function.

Note: In Calculate functions always first argument should be an aggregation function and you can pass multiple parameters.

Syntax:

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

Dax Function:

Calculate = **CALCULATE(sum(financials[Sales]))**

The screenshot shows a Power BI interface with two visualizations. On the left is a stacked bar chart titled "Sales and Profit by Country". The legend indicates "Sales" (blue) and "Profit" (dark blue). The chart shows data for Germany, with Sales at approximately 90% and Profit at approximately 10%. On the right is a large card visual displaying the value "23505352" with the label "Sales" below it. A "Calculate" button is also present. In the top right corner, a context menu is open over the card visual, showing options for "Fields". The "Sum" option is highlighted with a green box.

Note: If you drag the sales column from “Financials” data set to the card and calculating sum of sales in measure, then drag that measure to the “Card” visual are also same. See the above screenshot.

This screenshot shows a different arrangement of Power BI visualizations. It includes a stacked bar chart, a filter pane, and two card visualizations. The first card visual displays the value "23505352" with the label "Sales" below it. The second card visual displays the value "23505352" with the label "Calculate" below it. All three elements are highlighted with green boxes.

Even though if you select other value then automatically filters are applied and values are changed in both measures. So restrict this, we have to use filters.

Note: If we drag a sales column and calculate measure for sales both are same but if we want to apply conditions that we can do in measures only.

If you want to write conditions then we have to use “**Filter Functions**” in **Calculate** function.

Using Filter in Calculate function:

Dax Function:

```
Calculate1 = CALCULATE(sum(financials[ Sales]),
FILTER(financials,financials[Country]="Canada"))
```

Ex)

The screenshot shows a Power BI report interface. In the top right corner, there is a 'Calculations' pane with a formula bar containing the code: `Calculate1 = CALCULATE(sum(financials[Sales]), FILTER(financials,financials[Country]="Canada"))`. Below the formula bar, there is a chart visual showing sales by country. To the right of the chart, a large numerical value **118726385** is displayed, which is highlighted with a green border. Further down, another numerical value **24887660** is also highlighted with a green border. On the far right, the 'Fields' pane is open, showing a list of fields including 'Sales', 'All', 'All_Cross_Filtered', etc., with 'Calculate1' also highlighted with a green border.

In this calculate function, we can write condition using “**Filter**” function, based on that conditions values will be displayed.

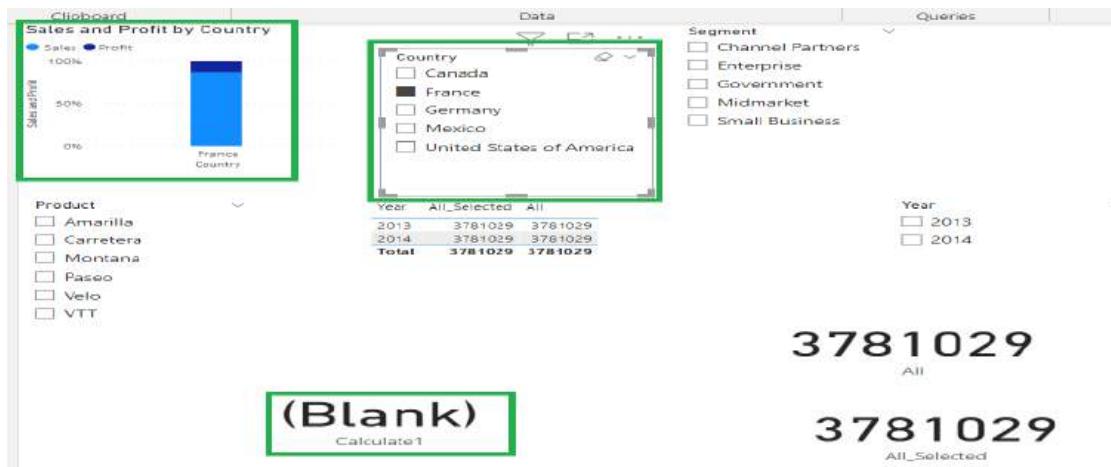
That means I am filtering “**Canada**” values from **country** column of **financials** table and calculating **sum of sales** corresponding to the value “**Canada**”.

If you filter “**Canada**” in slicer visual then corresponding measure value will be displayed. Otherwise it will return “**Blank**”. See the below screenshot.

Filtering “Canada” Values

The screenshot shows a Power BI report interface. On the left, there is a 'Sales and Profit by Country' chart with a single bar for 'Canada'. A 'Country' slicer is visible above the chart, with 'Canada' selected. Below the chart, a table shows sales data for 2013 and 2014, with a total row. The total value is **3529232**, which is highlighted with a green border. At the bottom left, a large numerical value **24887660** is also highlighted with a green border. On the right side, there is a 'Segment' slicer with options like 'Channel Partners', 'Enterprise', etc., and a 'Year' slicer with '2013' and '2014' selected. The overall layout shows how filtering the 'Country' slicer changes the calculated value.

Filtering other values:



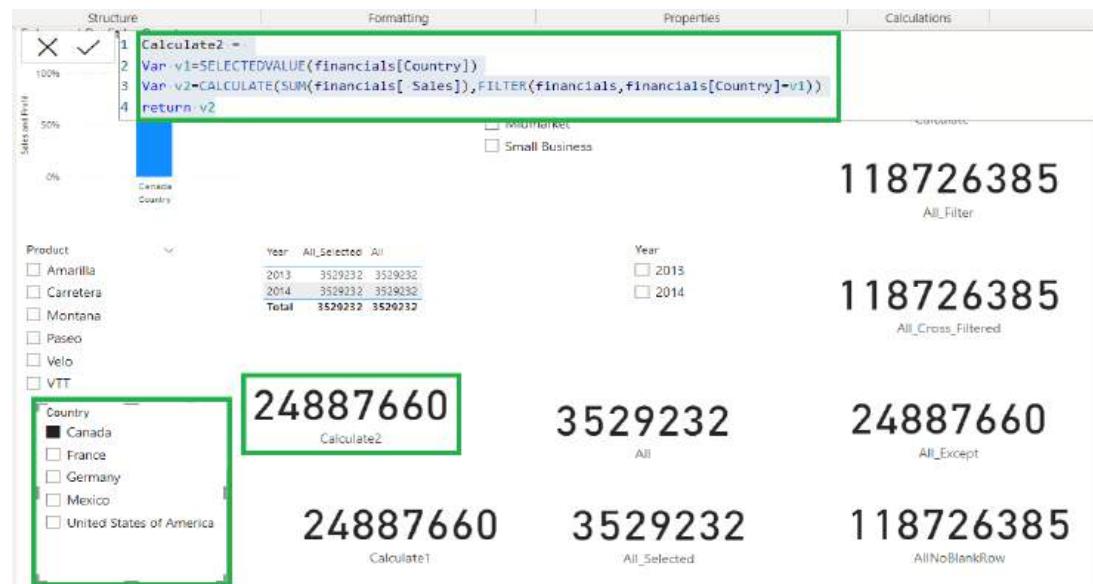
But i want values should be displayed based on the user selection. That means if the user selects any country value, corresponding to that value measure should be displayed. In that case we will write the formula like below.

Dax Function:

Calculate2 =

```
Var v1=SELECTEDVALUE(financials[Country])
Var v2=CALCULATE(SUM(financials[ Sales]),
                  FILTER(financials,financials[Country]=v1))
return v2
```

Ex)



Note: In this scenario, if you select any **country** value in slicer visual, based on that value; total sales will be displayed.

- **SELECTEDVALUE**: User selected value will be picked up from the selection and apply aggregation function according to that value and display the measure value.

For other values it will return alternate value that is mentioned in this function. If you don't mention any value as alternate result, then “**Blank**” will return.

Note: In this function, always use first argument should be column name.

Syntax:

SELECTEDVALUE(<columnName>[, <alternateResult>])

Dax Function:

Calculate2 =

```
Var v1=SELECTEDVALUE(financials[Country])
Var v2=CALCULATE(SUM(financials[ Sales]),
FILTER(financials,financials[Country]=v1))
return v2
```

Note: We can write multiple conditions in filter function as well using “**&&**” and “**||**” symbols.

- **EARLIER**: This function is used for comparing with previous values. Running total will be calculated by checking current value with previous value based on the condition.

Note: Most of the times this function will be applied in calculated column.

Syntax:

EARLIER(<column>, <number>)

Dax Function:

Earlier=

```
CALCULATE(sum(Company[Expenditure]),FILTER(Company,
EARLIER(Company[Date]) >= Company[Date]))
```

Ex)

The screenshot shows a Power BI Data view with a table containing four columns: Company, Date, Expenditure, and Earlier. The data consists of five rows for TCS. The 'Earlier' column contains the running total of 'Expenditure' for all rows up to the current one. The formula bar at the top displays the DAX formula: `EARLIER = CALCULATE(sum(Company[Expenditure]),FILTER(Company,EARLIER(Company[Date])>=Company[Date]))`. The table is styled with a light gray background and white text.

Company	Date	Expenditure	Earlier
TCS	31-08-2022	3000	3000
TCS	30-09-2022	2000	7000
TCS	30-09-2022	2000	7000
TCS	31-10-2022	1000	9000
TCS	31-10-2022	1000	9000

Time Intelligence Functions

These functions are used to manipulate data using time periods, including days, months, quarters, and years, and then build and compare calculations over those periods.

Some of the time intelligence functions will return the values as follows

MONTH	JAN	FEB	MAR	APR	MAY	JUN
	1000	2500	3000	5000	6000	9000
FUNCTIONS	1000	2500	3000	5000	6000	9000
	2000	-500	1000	1500	3000	-1000
	-500	1000		-500		1000
TOTAL	2500	3000	5000	6000	9000	10000
OPENINGBALANCEMONTH	1000	2500	3000	5000	6000	9000
OPENINGBALANCEQUARTER	1000			5000		
OPENINGBALANCEYEAR	1000					
CLOSINGBALANCEMONTH	2500	3000	5000	6000	9000	10000
CLOSINGBALANCEQUARTER			5000			10000
CLOSINGBALANCEYEAR						10000

- **DATEADD**: This function is used for adding values for the date column based on the interval.

Def: Returns a table that contains a column of dates, shifted either forward or backward in time by the specified number of intervals from the dates in the current context.

Note: Most of the times this function will be applied in calculated column. You can create a new data set by applying in “**New Table**” option also.

Syntax:

`DATEADD(<dates>, <number_of_intervals>, <interval>)`

Dax Function:

`DateAdd = DATEADD(FINANCIALS2[Date], 1, YEAR)`

Name Column \$% Format General \sum

Data type Whole number DATEADD(Dates, NumberofIntervals, Interval)

Structure Moves the given set of dates by a specified interval.

X	✓	1 DateAdd=DATEADD(financials[Date],1,	
Product	Discount Band	Units Sold	Manu
Carretera	None	1513	3
Paseo	None	1006	10
..

DAY
MONTH
QUARTER
YEAR

Formatting		Properties		Sort		Groups		Relationships		Calculations	
1 DateAdd = DATEADD(FINANCIALS2[Date],1,YEAR)											
Discount Band	Units Sold	Manufacturing Price	Sale Price	Gross Sales	Discounts	Sales	COGS	Profit	Date	Month Number	Month Name
None	1513	3	350	529550	0	529510	393380	136170	01 December 2014	12	December
None	1006	10	350	352100	0	352100	261560	90540	01 June 2014	6	June
None	1723	10	350	603750	0	603750	448500	155250	01 November 2013	11	November
None	1513	10	350	529550	0	529510	393380	136170	01 December 2013	12	December
None	1006	10	350	352100	0	352100	261560	90540	01 June 2014	6	June
None	1527	100	350	534450	0	534410	397020	137430	01 September 2013	9	September
None	2750	160	350	962500	0	962500	715000	247500	01 February 2014	2	February
Low	1210	3	350	429500	4235	419265	314600	104665	01 March 2014	3	March
Low	1397	3	350	488950	4869.5	484060.5	363220	120840.5	01 October 2014	10	October
Low	2155	3	350	754230	7542.5	746707.5	580300	186407.5	01 December 2014	12	December
Low	2155	10	350	754230	7542.5	746707.5	560300	186407.5	02 December 2014	12	December
Low	943.5	150	350	330225	3302.25	326922.75	245310	81612.75	01 April 2014	4	April
Low	1397	150	350	488950	4889.5	484060.5	363220	120840.5	01 October 2014	10	October
Low	2852	3	350	998200	19964	978236	741520	236716	01 December 2014	12	December
Low	2852	10	350	988200	19964	978236	741520	236716	01 December 2014	12	December

Notes: After applying this function, results will be displayed based on the column we have taken for the “DAX” function.

For example, I have applied **DateAdd** function on “**Date**” column. In this I want to add “1” value to the year value and display result. If you want to display the result value; that value also should be present in the “**Date**” Column as well. Then only result will be displayed otherwise null value will be displayed.

Suppose, I have value in Date column like “01-12-2014”. I have applied function on this.

DATEADD(FINANCIALS2[Date],1,YEAR)

Result should be come as “**01-12-2015**” right. Because I have added 1 value to the year. But **DATEADD** function checks result value in “**Date**” column.

If this value is present in that column; then only it will display that value. Otherwise blank value will be returned.

In the above example, “**01-12-2015**” value is not present in the “Date” Column. That’s why result displayed as blank

Creating single column table by applying this function in “New Table” option.



Note: If you create new data set, only result values will be displayed without nulls. That means all these values are present in “**Date**” column.

- **DATESBETWEEN**: This function is used for adding values for the date column based on the interval.

Def: Returns a table that contains a column of dates that begins with a specified start date and continues until a specified end date.

This function is suitable as filter to use in “**CALCULATE**” function.

Note: Most of the times this function will be applied in calculated column. You can create a new data set by applying in “**New Table**” option also.

Syntax:

DATESBETWEEN(<Dates>, <StartDate>, <EndDate>)

Dax Function:

DatesBetween = **CALCULATE(SUM(FINANCIALS2[Sales]),
DATESBETWEEN(FINANCIALS2[Date],"2013-01-01","2013-12-31"))**

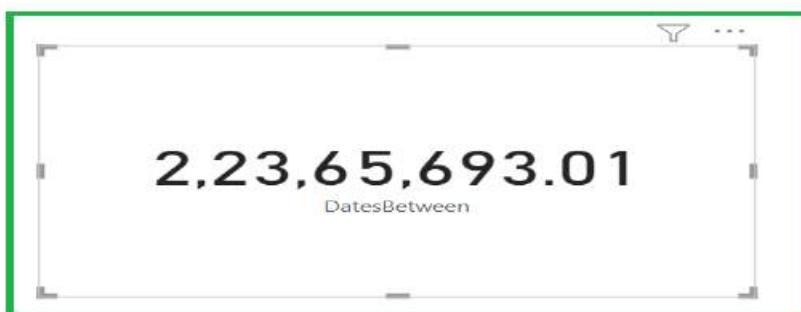
In the above DAX function, I want to display sum of sales for the date column between 01-01-2013 to 31-012-2013. So the result will be coming as follows.

Now we will compare whether the result came correctly or not. Filter the date values in the date column between 01-01-2013 to 31-012-2013 and copy the sales column values to the word.

A screenshot of a Power BI report showing a table with columns: Discounts, Sales, COGS, Profit, Date, Month Number, Month Name, and some summary measures at the bottom. The 'Date' column is selected, and a context menu is open. The menu includes options: Sort ascending, Sort descending, Clear sort, Clear filter, Clear all filters, and Date filters. A sub-menu for Date filters is open, showing a list of dates from 01 September 2013 to 01 December 2013. The months from September to December are checked. A green box highlights the 'OK' button at the bottom right of the dialog.

Result is 22365693.0.

Then go to “Report” tab, take “Card” visual and put the “**Datesbetween**” column in that. Then you can see the same value as you have seen in the above.



- **DATESINPERIOD:** Results will be displayed between the start date and up to some period using intervals value.

Def: Returns a table that contains a column of dates that begins with a specified start date and continues for the specified number and type of date intervals.

This function is suitable as filter to use in “**CALCULATE**” function.

Note: Most of the times this function will be applied in calculated column. You can apply it on “**New Measure**” or you can create a new data set by applying in “**New Table**” option also.

I have applied this function in “New Measure” to find out the sum of sales of 1 year less than max date to the end date using “**Year**” interval.

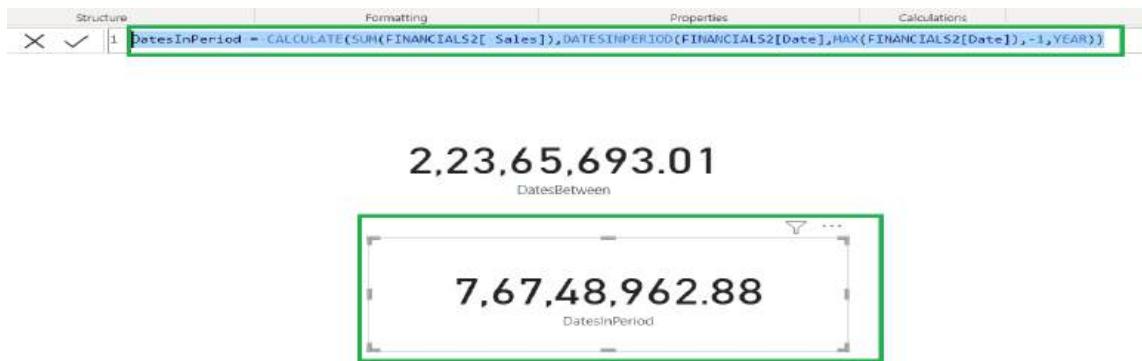
Syntax:

`DATESINPERIOD(<dates>, <start_date>, <number_of_intervals>, <interval>)`

Dax Function:

```
DatesInPeriod = CALCULATE(SUM(FINANCIALS2[ Sales]),  
DATESINPERIOD(FINANCIALS2[Date],MAX(FINANCIALS2[Date]),-1,YEAR))
```

If you calculate manually you can see the result as “76748962.88” by copying sales column values in that period to “Excel” and compare with measure value with this. Both are same. Then result is true.



- **DATESMTD:** Returns a table that contains a column of the dates for the month to date, in the current context.

That means, considering latest month from the selected date column, based on that month, aggregations will be performed.

Syntax:

`DATESMTD(<dates>)`

Dax Function:

```
DatesMTD = CALCULATE(SUM(FINANCIALS2[ Sales]),  
DATESMTD(FINANCIALS2[Date]))
```

Structure Formatting Properties Calculations

X ✓ 1 `DatesMTD = CALCULATE(SUM(FINANCIALS2[Sales]),DATESMTD(FINANCIALS2[Date]))`

2,23,65,693.01

DatesBetween

7,67,48,962.88

DatesInPeriod

96,94,942.90

DatesMTD

- **DATESQTD**: Returns a table that contains a column of the dates for the quarter to date, in the current context.

That means, considering latest quarter from the selected date column, based on that quarter, aggregations will be performed.

Syntax:

`DATESQTD(<dates>)`

Dax Function:

`DatesQTD = CALCULATE(SUM(FINANCIALS2[Sales]),
DATESQTD(FINANCIALS2[Date]))`

Structure Formatting Properties Calculations

X ✓ 1 `DatesQTD = CALCULATE(SUM(FINANCIALS2[Sales]),DATESQTD(FINANCIALS2[Date]))`

2,23,65,693.01

DatesBetween

2,45,93,990.77

DatesQTD

7,67,48,962.88

DatesInPeriod

- **DATESYTD**: Returns a table that contains a column of the dates for the year to date, in the current context.

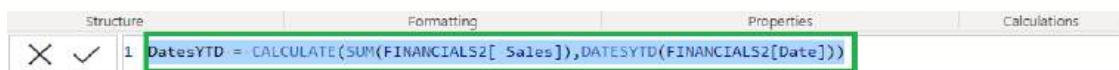
That means, considering latest year from the selected date column, based on that year, aggregations will be performed.

Syntax:

DATESYTD(<dates> [,<year_end_date>])

Dax Function:

DatesYTD = CALCULATE(SUM(FINANCIALS2[Sales]),
DATESYTD(FINANCIALS2[Date]))



2,23,65,693.01

DatesBetween

2,45,93,990.77

DatesQTD

7,67,48,962.88

DatesInPeriod

7,67,48,962.88

DatesYTD

- **ENDOFMONTH:** Returns a table that contains a column of the dates for the year to date, in the current context.

That means, considering latest end of the month from the selected date column, based on that year, aggregations will be performed.

Syntax:

DATESYTD(<dates> [,<year_end_date>])

Dax Function:

EndofMonth = CALCULATE(SUM(FINANCIALS2[Sales]),
ENDOFMONTH(FINANCIALS2[Date]))

Structure Formatting Properties Calculations

EndofMonth = CALCULATE(SUM(FINANCIALS2[Sales]),ENDOFMONTH(FINANCIALS2[Date]))

2,23,65,693.01 DatesBetween	2,45,93,990.77 DatesQTD
7,67,48,962.88 DatesInPeriod	7,67,48,962.88 DatesYTD
96,94,942.90 DatesMTD	96,94,942.90 EndofMonth

Note: DATESMTD and ENDOFMONT are same in this context.

- **FIRSTDATE:** Returns the first date in the current context for the specified column of dates.

That means, considering first date of latest month from the selected date column, based on that date, aggregations will be performed.

Syntax:

FIRSTDATE(<dates>)

Dax Function:

FirstDate = CALCULATE(SUM(FINANCIALS2[Sales]),
FIRSTDATE(FINANCIALS2[Date]))

Structure Formatting Properties

FirstDate = CALCULATE(SUM(FINANCIALS2[Sales]),FIRSTDATE(FINANCIALS2[Date]))

<p>Date</p> <ul style="list-style-type: none"> <input type="checkbox"/> 01 September 2013 <input type="checkbox"/> 01 October 2013 <input type="checkbox"/> 01 November 2013 <input type="checkbox"/> 01 December 2013 <input type="checkbox"/> 01 January 2014 <input type="checkbox"/> 01 February 2014 <input type="checkbox"/> 01 March 2014 <input type="checkbox"/> 01 April 2014 <input type="checkbox"/> 01 May 2014 <input type="checkbox"/> 01 June 2014 <input type="checkbox"/> 01 July 2014 <input type="checkbox"/> 01 August 2014 <input type="checkbox"/> 01 September 2014 <input type="checkbox"/> 01 October 2014 <input type="checkbox"/> 01 November 2014 <input checked="" type="checkbox"/> 01 December 2014 	<p>96,94,942.90 FirstDate</p>
---	-----------------------------------

- **TOTALMTD**: Current year, current month based aggregation will be done and display the result.

Evaluates the value of the **expression** for the month to date, in the current context.

Syntax:

TOTALMTD(<expression>, <dates>[, <filter>])

- **TOTALQTD**: Current year, current Quarter based aggregation will be done and display the result.

Evaluates the value of the **expression** for the quarter to date, in the current context.

Syntax:

TOTALQTD(<expression>, <dates>[, <filter>])

- **TOTALYTD**: Current year based aggregation will be done and display the result.

Evaluates the year-to-date value of the **expression** in the current context.

Syntax:

TOTALYTD(<expression>, <dates>[, <filter>])

- **PARALLELPERIOD**: Returns parallel period of dates by the given set of dates and a specified interval

Syntax:

PARALLELPERIOD(<dates>, <number_of_intervals>, <interval>)

Modeling

What is model?

Providing a relationship between the data sets is called as model.

Why we need provide a relationship?

If we want to take the columns from multiple tables, then we have to give relationship between them.

If we want to design the visual based on only one data set then we don't require relationship between them.

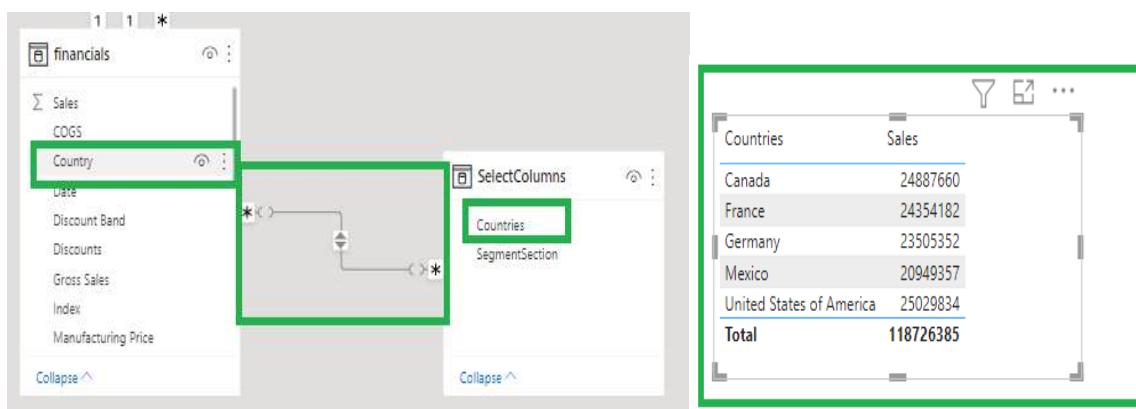
Suppose I am taking columns from two different tables without having relationship between them, See the below screens shot.

Countries	Sales
Canada	118726385
France	118726385
Germany	118726385
Mexico	118726385
United States of America	118726385
Total	118726385

In the above visual, I have taken countries column from “**Financials**” data set and sales column from “**SelectColumns**” data set.

So the values are looking same for all the countries. Why because there is no relationship between them.

If we want to give the relationship between them just put pointer on one data set and drag and drop to another data set, automatically relationship will be established.



Then data is showing in proper way. To create a visual, sometimes we will take columns from multiple tables. In that case, relationship is required between multiple tables.

Note: If we want to provide relationship between the data sets, common columns must be required between them.

We have to decide the common columns before providing relationship. Based on that common column only, data will be displayed.

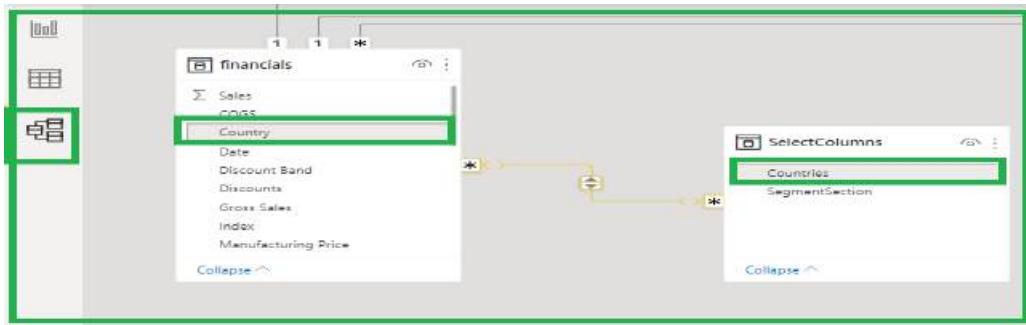
For common column, both columns should be having same data type, names might be different.

We can provide relationship between two data sets in two ways.

- 1) Drag and drop from one data set to another data set
- 2) By using “**Manage Relationship**” option from “**Home**” ribbon

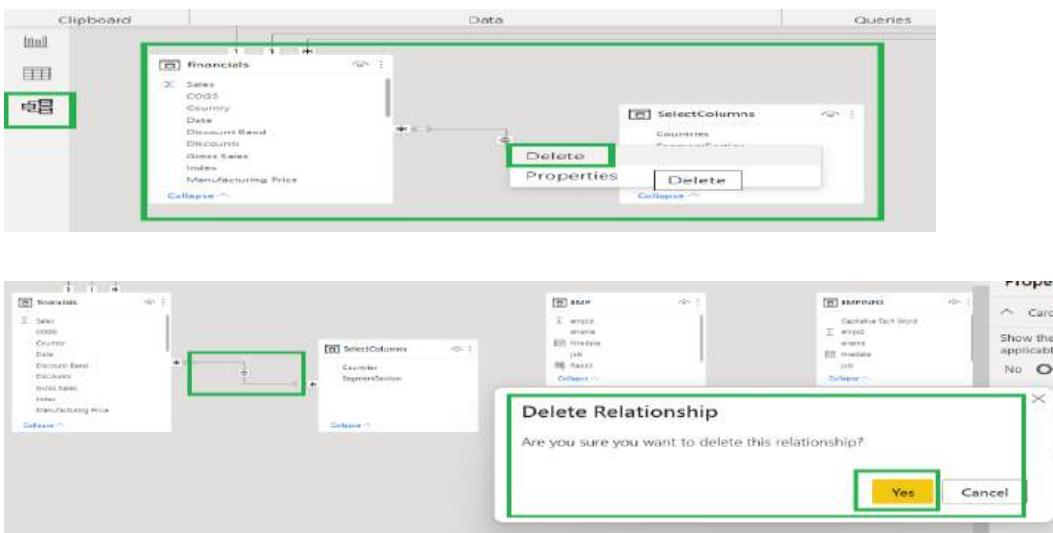
1) Drag and Drop from one data set to another data set.

Go to **Model**; choose the data sets that you want to provide the relationship. Then put mouse pointer on common column of one data set and drag from there to common column of another data set. Automatically relationship will be established.



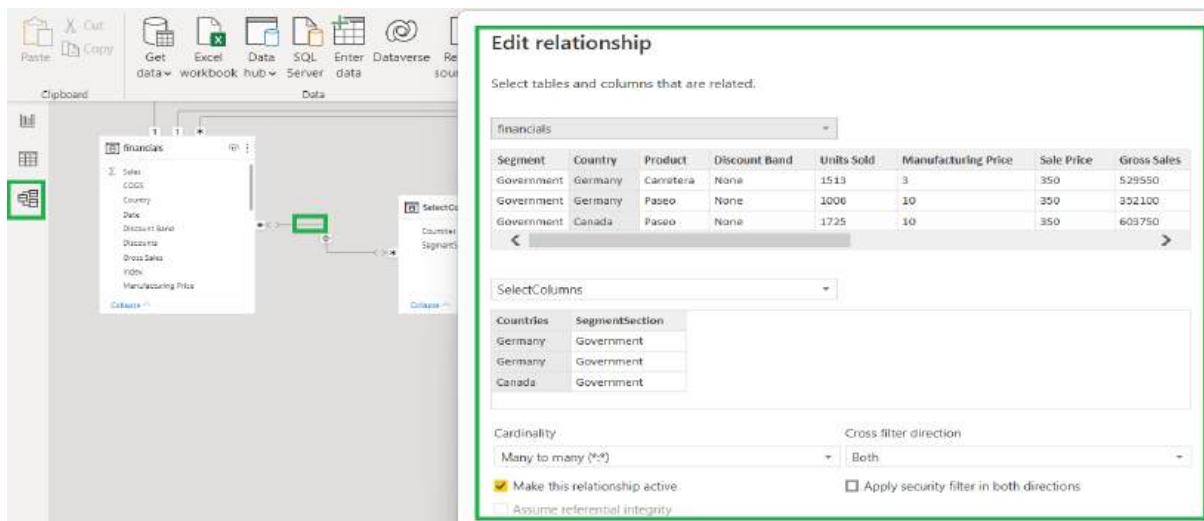
Deleting a relationship between data sets

Go to **Model**, Choose the relationship line between the data sets, and **right click** on that line then you will see **delete** option. Then it will show you the dialogue box and ask whether you want to delete relationship or not. Then you have to click **"Yes"**. Automatically relationship will be deleted between the selected data sets.



Edit a relationship between data sets

Go to **Model**, Choose the relationship line between the data sets, and **double click** on that line then you will see **new window**, which is called "Edit Relationship Window".

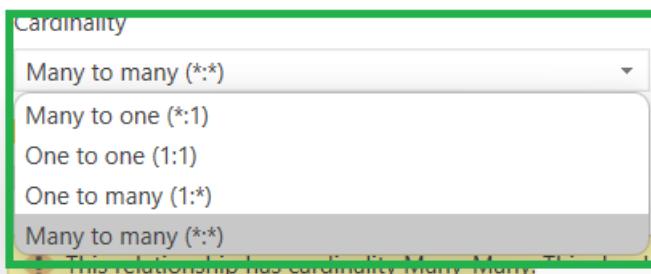


In this window, we have to select two tables and select common column between them. Once common column selected, automatically cardinality will be set by default by the power bi itself.

Cardinality:

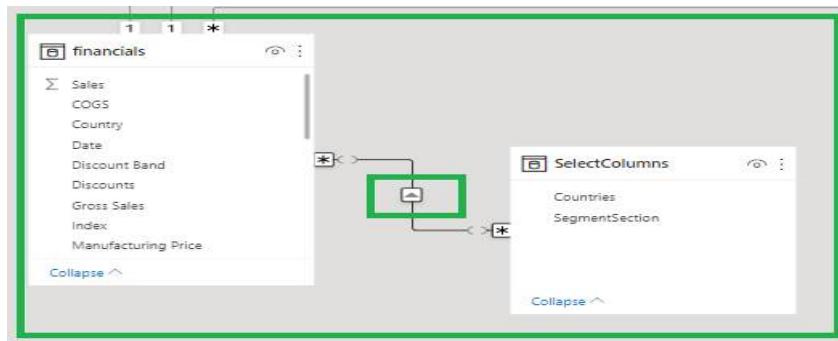
Cardinality determines the relationship between the data sets. There are 4 types of cardinalities.

- Once to One (1 – 1) → No duplicates between common column in both left and right tables.
- Many to One (* – 1) → Duplicates in left table common column and unique values in the right table common column.
- One to Many (1 – *) → Duplicates in right table common column and unique values in the left table common column.
- Many to Many (* – *) → Duplicates in both right and left tables common columns



Cross filter direction: It will decide the direction of filtering of values from one table to another table. These are two types.

- Single: It is single way direction from one table to another



In the above example, direction shows from “**SelectColumns**” table to “**financials**” table.

Let's discuss with example, suppose, I am taking **sales** and **country** columns from financials data set and put into the table visual.

Again I am taking slicer visuals for both columns from both tables. Taking **countries** column from “**SelectColumns**” table, **Country** column from “**financials**” table.

Countries	Sales
Canada	24887660
France	24354182
Germany	23505352
Mexico	20949357
United States of America	25029834
Total	118726385

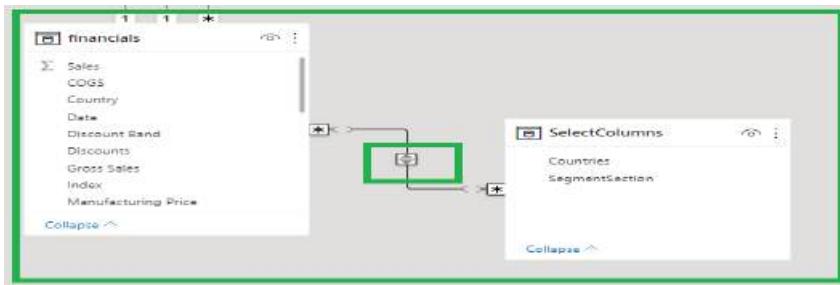
If you select any value from “**Countries**” column which is common column from “**SelectColumns**” table, then automatically remaining two visuals will be effected, because direction is there from “**SelectColumns**” to “**financials**”

Countries	Sales
France	24354182
Total	24354182

But if you select any value in “**country**” column, which is common column from “**financials**” data set, then “**Countries**” column won't be affected, because there is no direction from “**country**” column to “**countries**” column.

Countries	Sales
France	24354182
Total	24354182

→ Both: It is both way directions from both tables.



If we have changed cross filter direction from “Single” to “Both”, then automatically bi-directional symbol will be appeared. See the above screen shot.

If you select any value from either “Countries” or “Country”, then filter will be done in both ways. See the below screenshot.



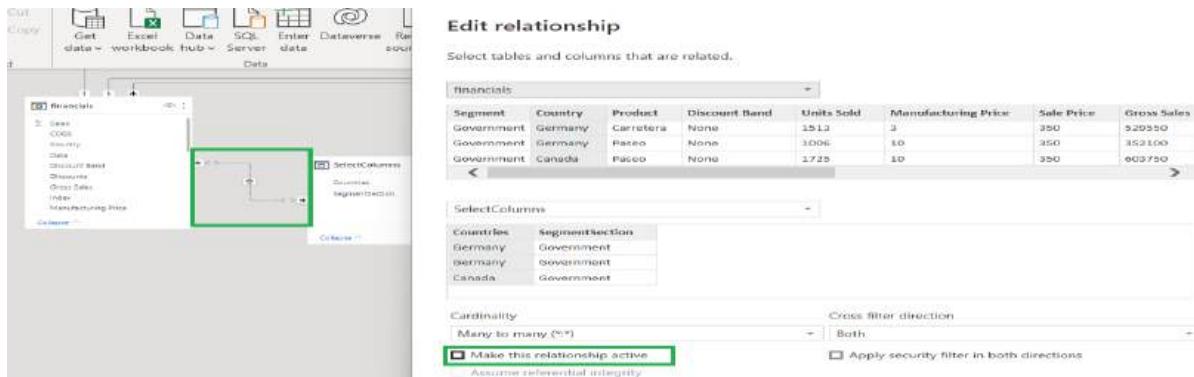
Note: We can change cross filter direction from “single” to “Both”, but we can’t mention direction from which table to which table. By default system will give the direction.

Relationship status: Active/ Inactive

Active: If the relationship is active, then relationship line will be visible as thick line. By default it will be in **active** only



Inactive: If the relationship is inactive, then relationship line will be visible as dotted lines.



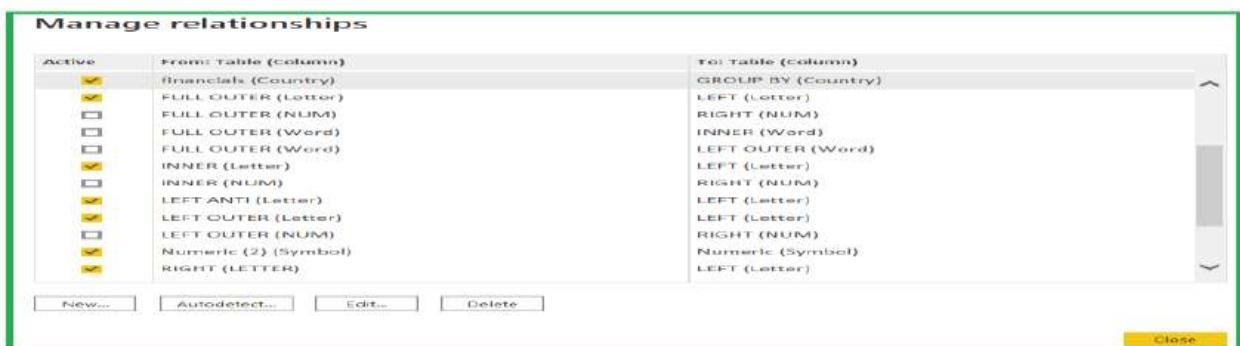
Note: In the Edit relationship window, if you see the “**Make this relationship active**” check box, make it always with tick mark. Then only all the operations will be performed very well.

If you want to make it “**inactive**”, click on the check box, automatically tick mark will be disappeared and relationship will change into inactive status.

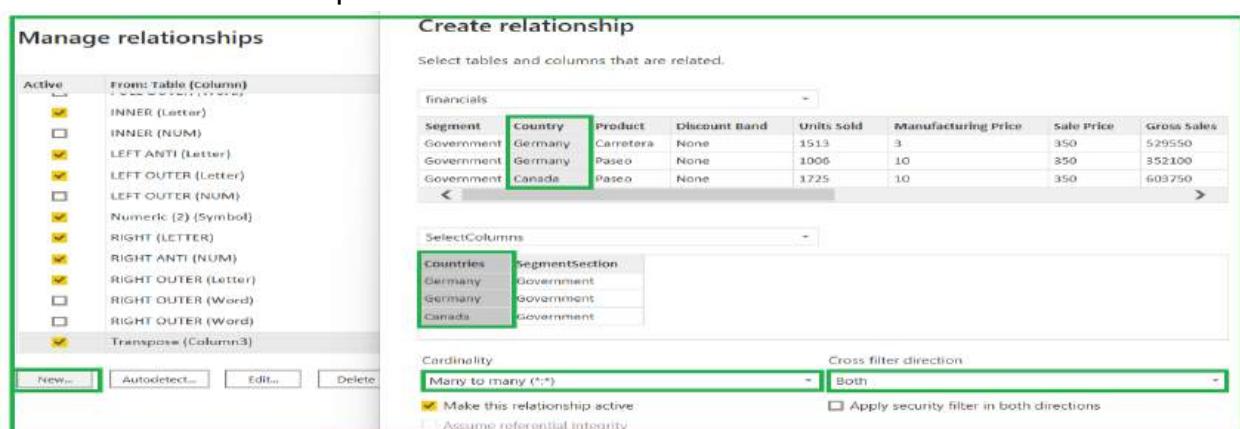
2) By using “Manage Relationship” option from “Home” ribbon:

You can create, delete or edit a relationship by simply go to “**Manage Relationship**” option.

Go to “**Home**” ribbon → “**Manage Relationship**” → New dialogue box will open.



New: If you want to create a new relationship, you have to click on “New” button. Then new window will open. See the below screenshot.

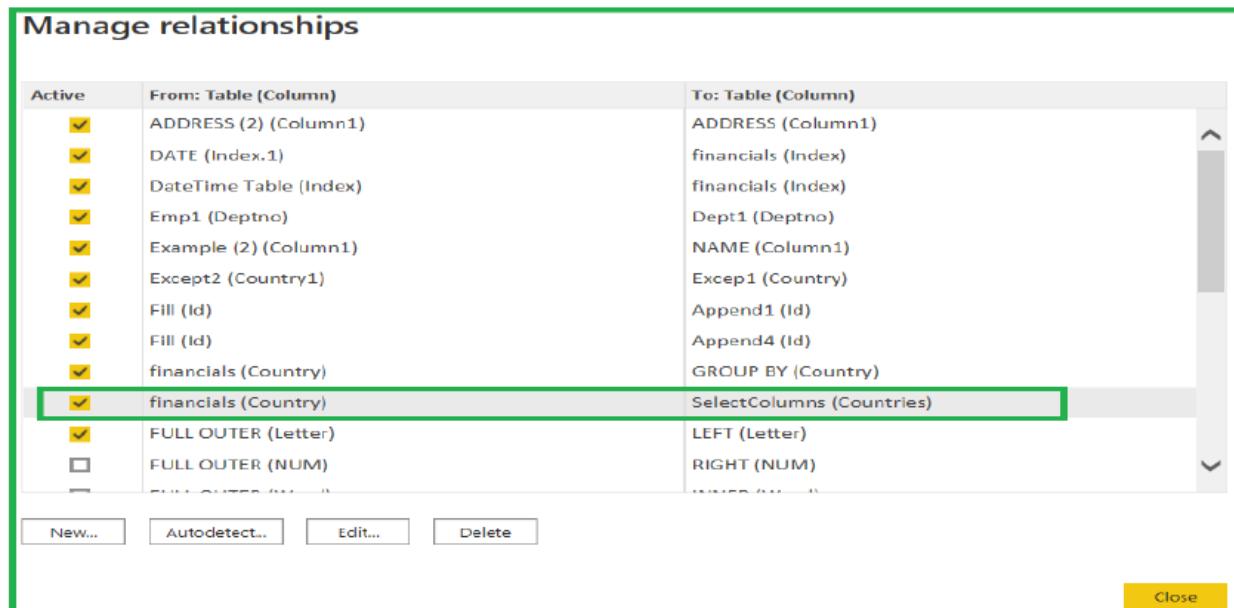


Select the tables from dropdown menu which you want to give the relationship, and select the common column between them. Automatically “**Cardinality**” will be set.

If there are any duplicates in common column then it will show “**Many to Many**” relationship.

In this example, both tables common columns having duplicates. That's why it is showing “**Many to Many**” relationship. Cross filter directions is also set as “**Both**”.

That means filters can be happened in both directions. Then click on “**Ok**”. That relationship will be visible in “**Manage Relationship**” option.



Edit: If you want to edit relationship, first select that relation in “**Manage relationships**” window and click on “**Edit**”. Then new window will be opened.

Edit relationship

Select tables and columns that are related.

financials

Segment	Country	Product	Discount Band	Units Sold	Manufacturing Price	Sale Price	Gross Sales
Government	Germany	Carretera	None	1513	3	350	529550
Government	Germany	Paseo	None	1006	10	350	352100
Government	Canada	Paseo	None	1725	10	350	603750

SelectColumns

Countries	SegmentsSection
Germany	Government
Germany	Government
Canada	Government

Cardinality

Many to many (*:*)

Make this relationship active

Assume referential integrity

Cross filter direction

Single (SelectColumns filters financials)

Single (SelectColumns filters financials)

Single (financials filters SelectColumns)

Both

Single (Sel)

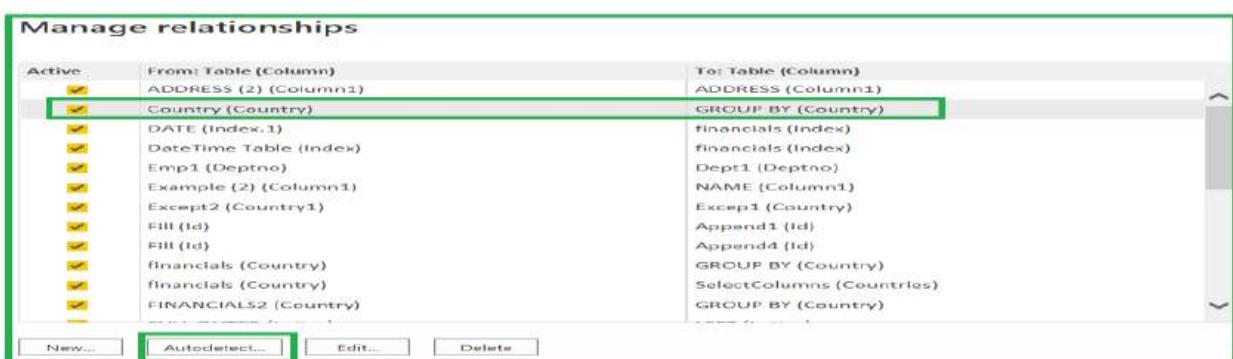
Here I want to change “**Cross filter direction**” as “**Single**”. I have selected that option and click on “**OK**”. Then automatically it changes to “**Single**” way direction.

Delete: If you want to delete a relationship, first select that relation in “Manage relationships” window and click on “Delete”. Then new window will be opened. In that, Click on “Delete”. Relationship will be deleted.



Autodetect: If you add a new table to the model that has field names matching with other tables in the model, Then Power BI automatically creates a relationship between each of the existing tables and the new table. That relationship will be visible in “Manage Relationships” option.

Suppose I have created “Country” data set with single column named as “Country” in “Enter Data” option. Then go to “Manage Relationships” option and click on “Autodetect” option. You can see new table having relation with “groupby” table country column.



So if you want to disable “autodetect” option in “Manage Relationships” option then you have to follow below steps.

To disable the auto-detect relationship feature in Power BI, you have to go to **File>Options and Settings>Options**. You can then go to the **Current File>Data Load**; disable the **autodetect relationships** in Power BI

Options and settings

Options

GLOBAL

- Data Load
- Power Query Editor
- DirectQuery
- R scripting
- Python scripting
- Security
- Privacy
- Regional Settings
- Updates
- Usage Data
- Diagnostics
- Preview features
- Auto recovery
- Report settings

CURRENT FILE

- Data Load**
- Regional Settings
- Privacy
- Auto recovery

Type Detection

- Detect column types and headers for unstructured sources

Relationships

- Import relationships from data sources on first load (i)
- Update or delete relationships when refreshing data (i)
- Autodetect new relationships after data is loaded (i)

[Learn more](#)

Time intelligence

- Auto date/time (i) [Learn more](#)

Background Data

- Allow data previews to download in the background

Parallel loading of tables

- Enable parallel loading of tables (i)

Q&A

- Turn on Q&A to ask natural language questions about your data (i)
- Create a local index so Q&A will work with your remote model
- Share your synonyms with everyone in your org

[Learn more](#)

Make relationship active and inactive:

If you want to make relationship active, then you have to click on check box of that relationship.

Manage relationships

Active	From: Table (Column)	To: Table (Column)
<input checked="" type="checkbox"/>	ADDRESS (2) (Column1)	ADDRESS (Column1)
<input type="checkbox"/>	Country (Country)	GROUP BY (Country)
<input checked="" type="checkbox"/>	DATE (Index,1)	financials (Index)
<input checked="" type="checkbox"/>	DateTime Table (Index)	financials (Index)
<input checked="" type="checkbox"/>	Emp1 (Deptno)	Dept1 (Deptno)
<input checked="" type="checkbox"/>	Example (2) (Column1)	NAME (Column1)
<input checked="" type="checkbox"/>	Except2 (Country1)	Excep1 (Country)
<input checked="" type="checkbox"/>	Fill (id)	Append1 (id)
<input checked="" type="checkbox"/>	Fill (id)	Append4 (id)
<input checked="" type="checkbox"/>	Financials (Country)	GROUP BY (Country)
<input checked="" type="checkbox"/>	Financials (Country)	SelectColumns (Countries)
<input checked="" type="checkbox"/>	FINANCIALS2 (Country)	GROUP BY (Country)

New... Autodetect... Edit... Delete

Manage relationships

The screenshot shows the 'Manage relationships' dialog. On the left, under 'From: Table (Column)', 'Country (Country)' is selected. On the right, under 'To: Table (Column)', 'GROUP BY (Country)' is selected. Both columns have checkboxes checked.

If we want to make a relationship inactive, you have to just uncheck the relationship.

Manage relationships

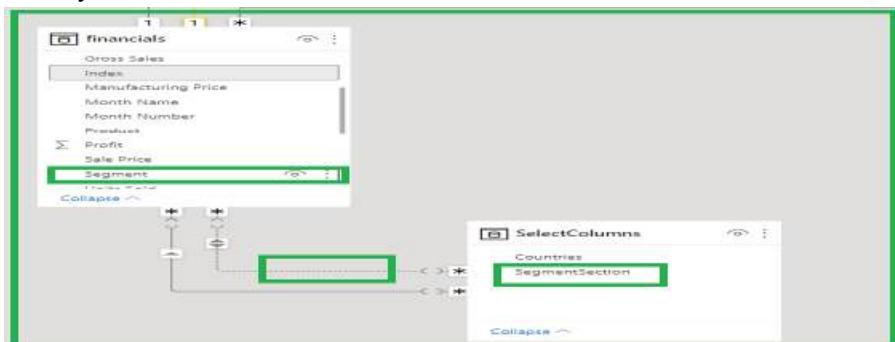
The screenshot shows the 'Manage relationships' dialog. On the left, under 'From: Table (Column)', 'Country (Country)' is selected and its checkbox is unchecked. On the right, under 'To: Table (Column)', 'GROUP BY (Country)' is selected. Both columns have checkboxes checked.

Manage relationships

The screenshot shows the 'Manage relationships' dialog. On the left, under 'From: Table (Column)', 'Country (Country)' is selected and its checkbox is unchecked. On the right, under 'To: Table (Column)', 'GROUP BY (Country)' is selected. Both columns have checkboxes checked.

Note: We can provide relationships for multiple common columns between the two data sets. But first common column relationship will be in **active** mode, rest of the common columns relation will be in **inactive** mode.

Suppose I am giving another common column between **financials** and "**selectcolumns**" tables. Common column is "**Segment**" and "**SegmentSection**". Now you can see the dotted lines for second common column.



Multiple columns relationship: if we want to make all the common columns relation active, that is not possible directly. But we can do that by using “**combine values**” dax function.

First select the columns which we want to combine in table1 and same thing do in second table as well. Then provide a relationship between them. Number of columns to combine in one table should match with another table.

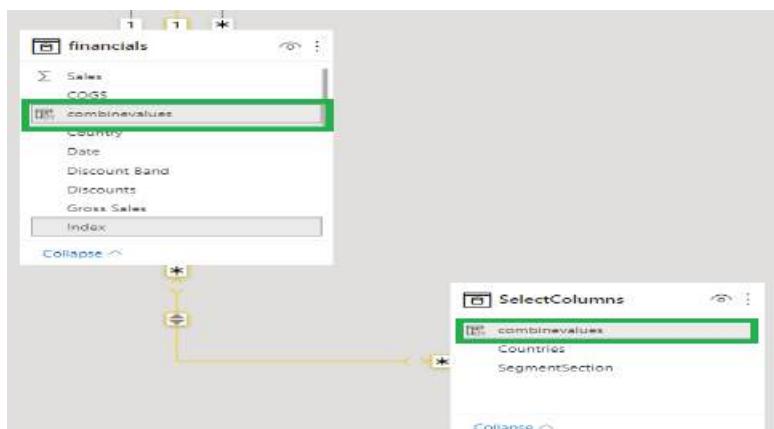
Ex)

```
Combinevalues(":",financials[country],financials[product])
```

Common column in **financials** table → **country:product**.

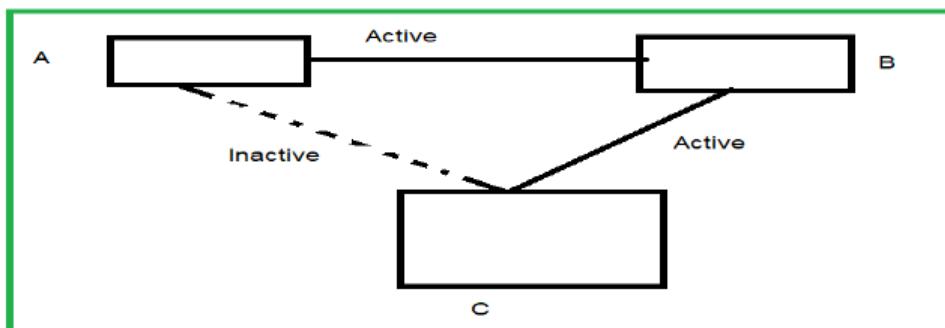
Common column in “**selectcolumns**” → **countries:segmentsection**

You have to give relationship between them. Then all the relationships will be in active state.

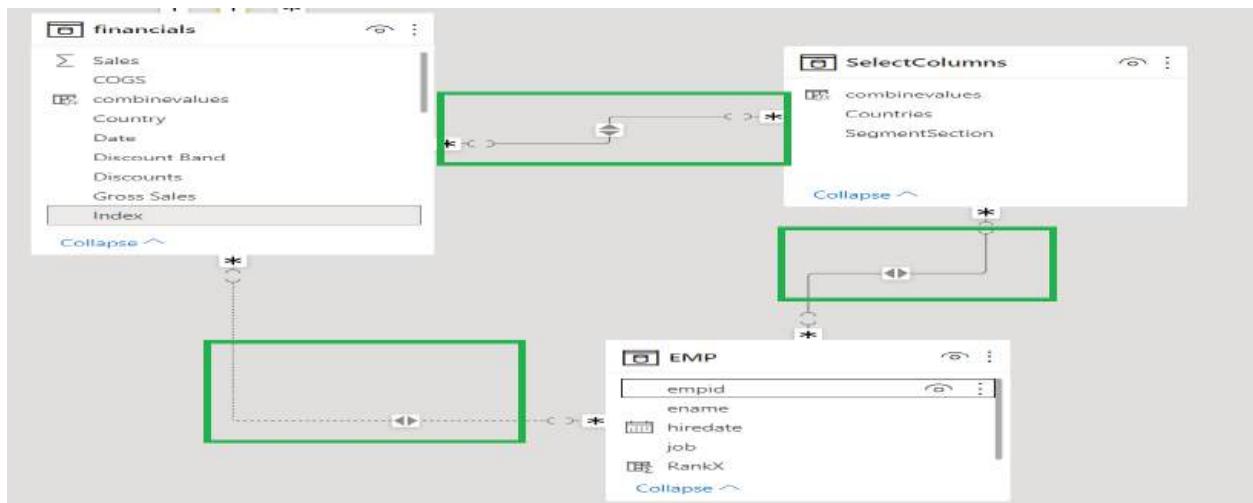


Note: If we want to make the relationship active then we can use “Userrelationship” function.

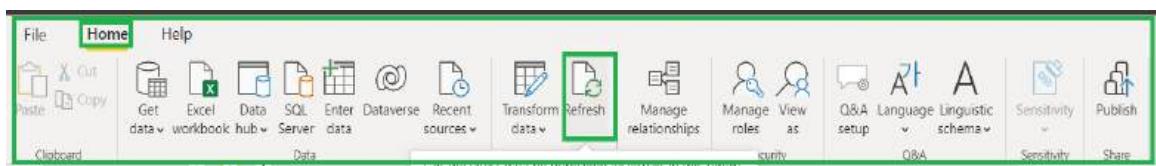
Triangle Relationship: Triangle relationship is not supported in power BI (Circular reference in Excel)



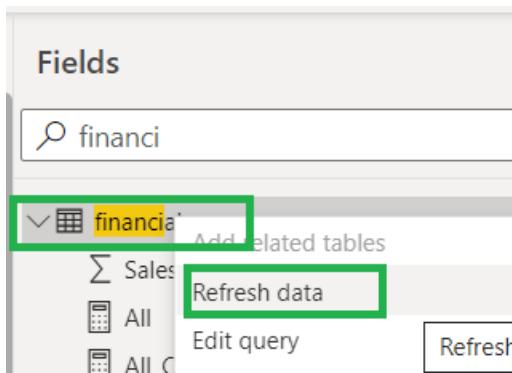
See the below example, I have provide relationship between the three data sets but two data sets are in active state, remaining one is in inactive state.



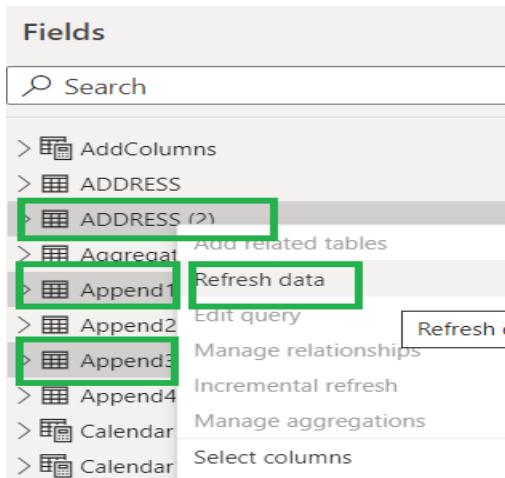
- **Refresh**: If you want to refresh all the data sets in model, then you have to click on this option. Whatever the data sets under the ‘fields’ pane, all will be refreshed or reloaded.



If you want to refresh particular dataset, then right click on that data set and click on “**Refresh data**” option. That data set will be refreshed.



If we want to refresh selection of the data sets, then you have to go to “Model” then select required data sets from fields pane by using “**CTRL+CLICK**” and right click on any one of data set. Then select “**Refresh Data**” option.



Many to Many relationship: Many to many relationship will not work properly. Most of the times; we have to avoid this many to many relationship. To avoid this; we have to use “**distinct**” function.

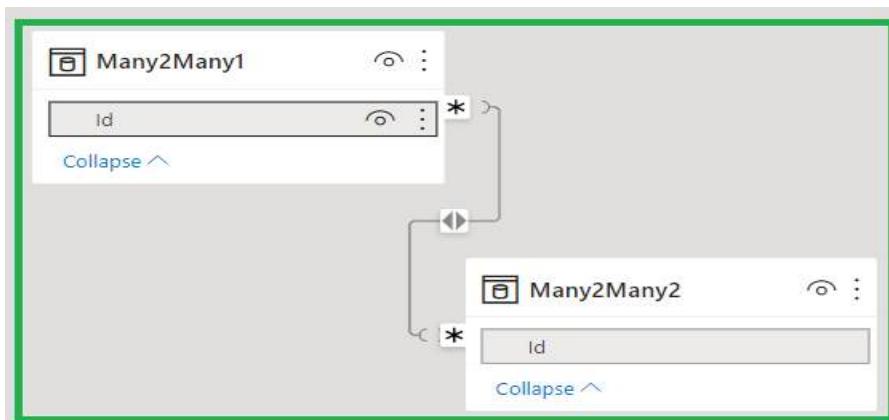
Ex)

Many2Many1 Many2Many2

Id
1
1
2
3
4

Id
2
3
3
4

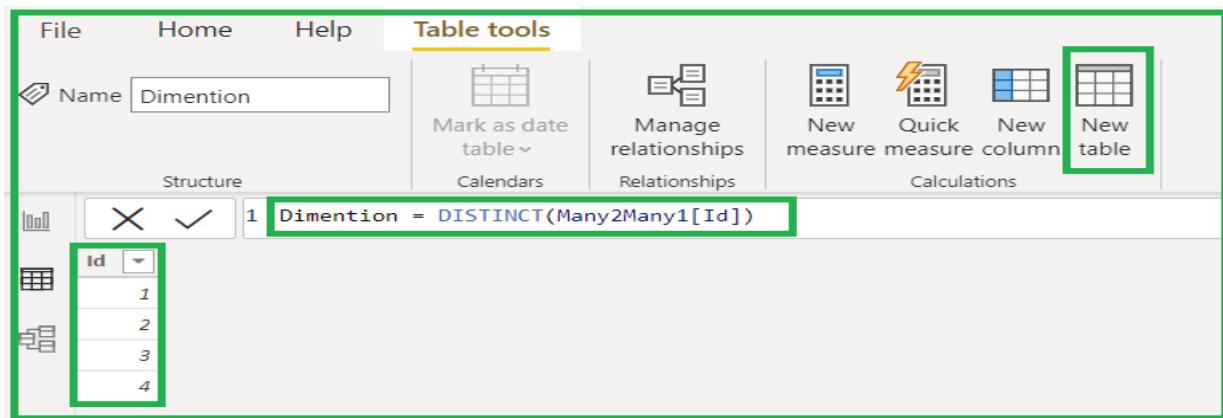
I have taken above two tables for example, and make a relationship between them in “Model”. Then it will show like below.



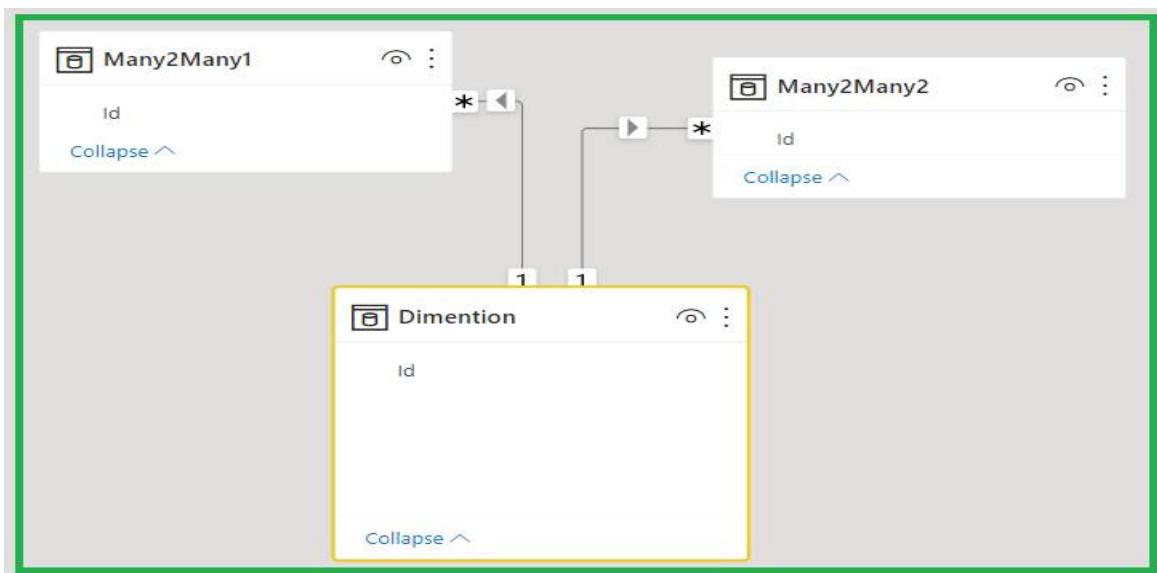
In some cases many to many relationship will not work; in that case we have to take on dimension table.

Create a dimension table on any common column of the above two tables using “**distinct**” function.

I am creating dimension table on “**Many2Many1**” data set by using **distinct** function.



Note: We have to create dimension table on the table having more number of distinct values. In “**Many2Many1**” I have 4 distinct values. That’s why I am applying on this.



So to avoid many to many relationship, we have to create dimension table and provide the relationships between the three data sets. That is “**Many to One**” and “**One to Many**”

Dimension table: Having distinct values is called as dimension table.

Fact table: Having duplicate values is called as fact table.

Doing operations if the relationship is in inactive state

If we want to do operations while the relationship is in “inactive” state, then we have to use “**USERELATIONSHIP**” function to change it to “active”.

Ex)

I want to create a measure using “**USERELATIONSHIP**” function by applying in **CALCULATE** function.

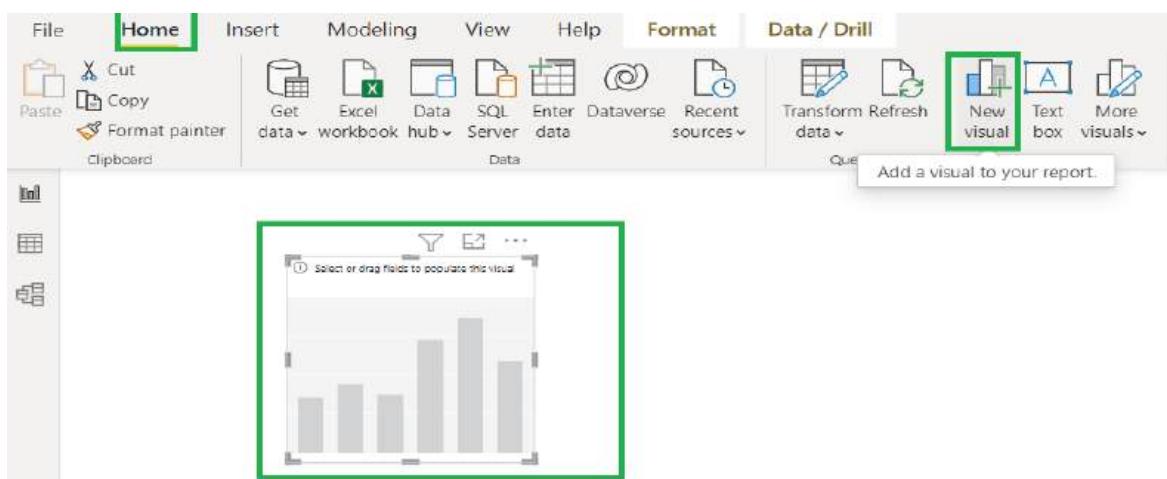
Click on “**New Measure**” and write the function as follows.

Dax Function:

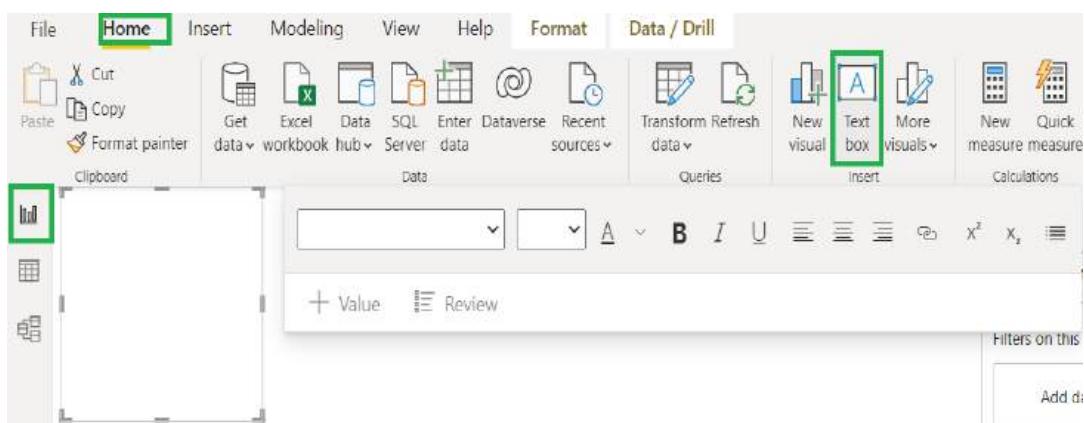
```
Userrelationship = CALCULATE(SUM(financials[ Sales]),  
USERELATIONSHIP(financials[Segment],  
'SelectColumns'[SegmentSection]))
```

Like this we can do the operations by using “**userrelationship**” function in “**calculate**” function.

- **New Visual:** If you click on this option by default it will bring “stacked column chart” into the development window. Or else you can directly drag the same visual from visualization pane. We can't get other visual from this option.



- **Text Box:** If you want to put title for the report we have to use “**Text Box**”.



- **More Visuals:** If you want to get visuals from app source or from a file you can use this option.

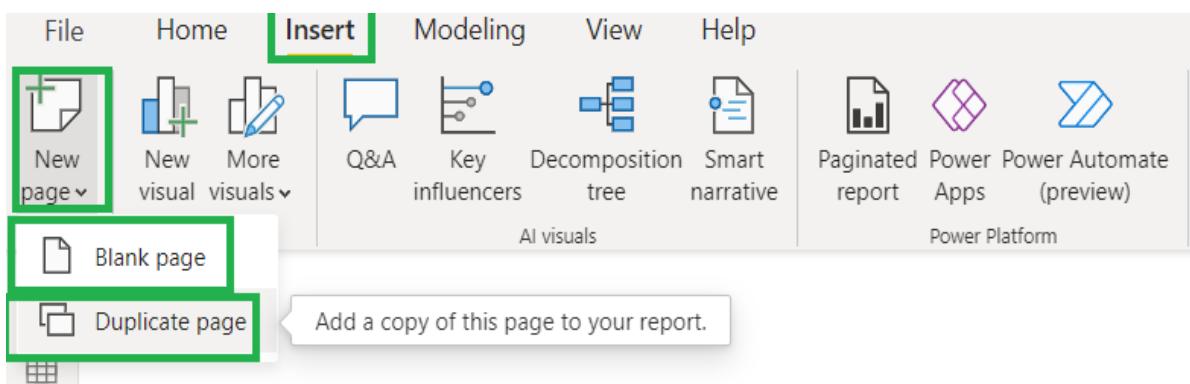
From App source: To get the custom visual from the market store you can use this option. This is same as “**Get more visuals**” option.

From my files: To get the custom visual from the file. This is same as “**import a visual from the file**” option.



❖ **Insert Ribbon:**

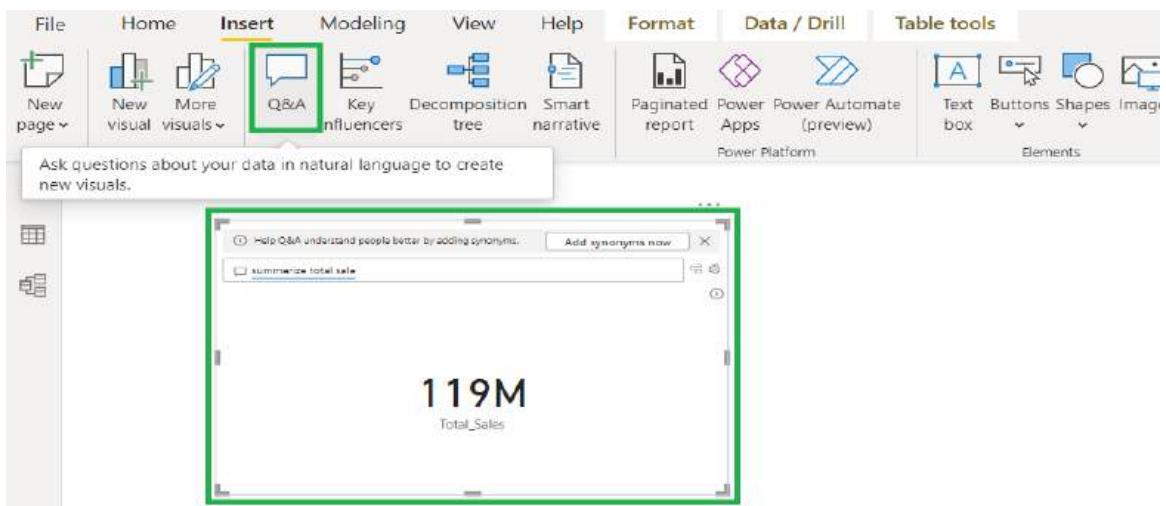
- **New Page:** If we want to create a new page, we have to click on this option. Or else you can create a duplicate page for the selected page.



Note: We can create a page by simply click on (+) button in the pages window. You can create a duplicate page by right click on that page.

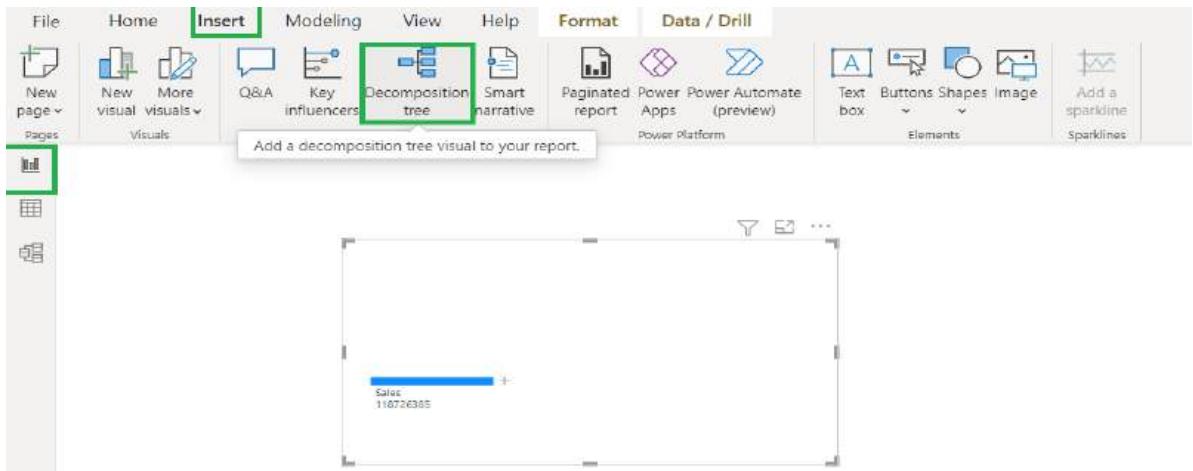
- **Q & A:** If you type a proper question in text box, then visual will be created based on that question. To select the **Q& A**, you can directly double click on the development window. These visuals are called as "**Artificial Intelligence visuals**"

Ex) I have written question as "**Summarize total sales**" then it shows the result as below.



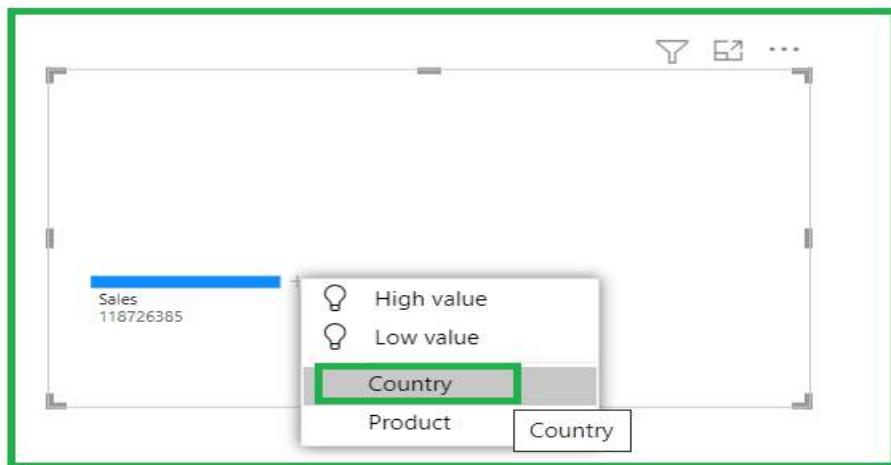
- **Decomposition Tree:** Results will be shown in different hierarchies is called as decomposition tree.

Go to “Insert” Menu, click on “Decomposition Tree” option.

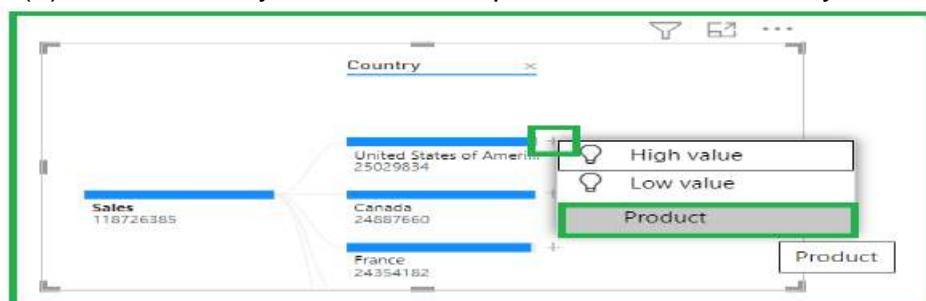


I am taking “financials” table and I put the **sales** column in “Analyze” section and **country, product** in “Explain by” section.

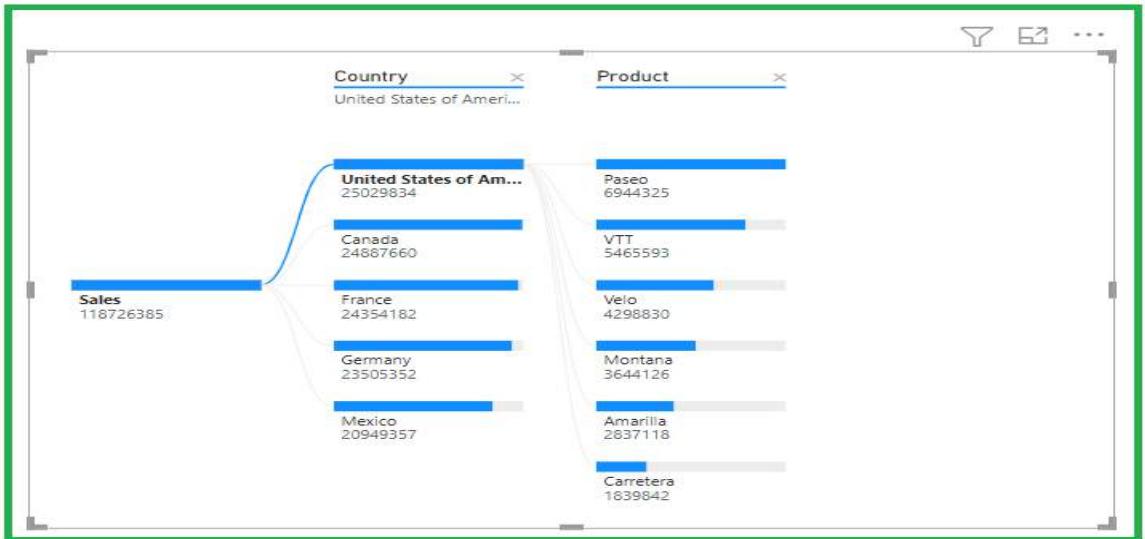
Total sales will be visible by the **analyze** section. If you click on (+) button then you can see another window, in that if you want to see the **sales** based on “**Country**” then you can click on that.



Again if you want to see the sales by country, along with product, then you have to click on (+) button. Then you can see the product for each country.



Decomposition tree is nothing but one type of filter. First sum of sales is displayed, then country wise total sales displayed, again that country wise sales will be divided product wise. That means we can see the different hierarchies in tree structure.



Note: We have to put aggregate values for ex. **Sum of sales**, these type of columns; we have to put in “**Analyze**”, remaining columns you have to put in “**Explain By**” option.

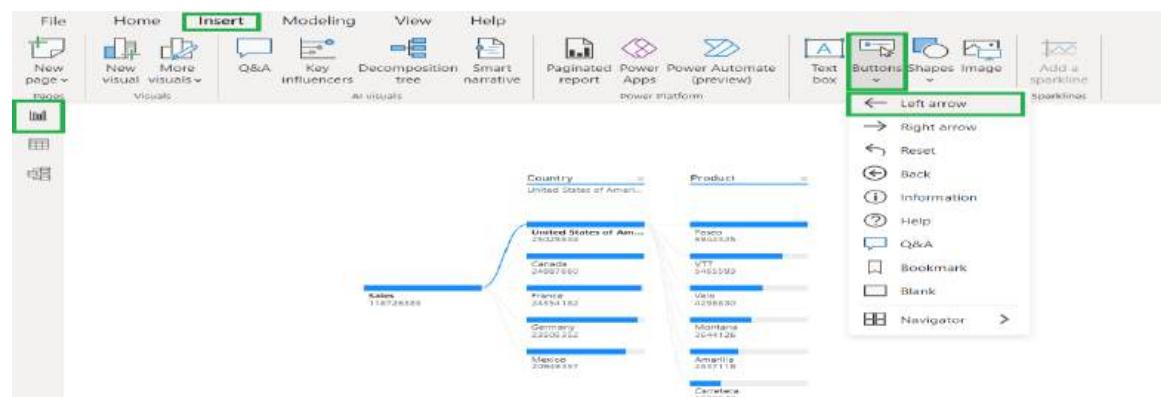
- **Buttons:** It will perform an action by simply clicking on the button.

In **Power BI Desktop**, on the **Insert** ribbon, select **Buttons** to reveal a drop-down menu, where you can select the button you want from a collection of options.

In the **Power BI service**, open the report in “**Editing**” view. Select **Buttons** in the top menu bar to reveal a drop-down menu, where you can select the button you want from a collection of options.

Ex)

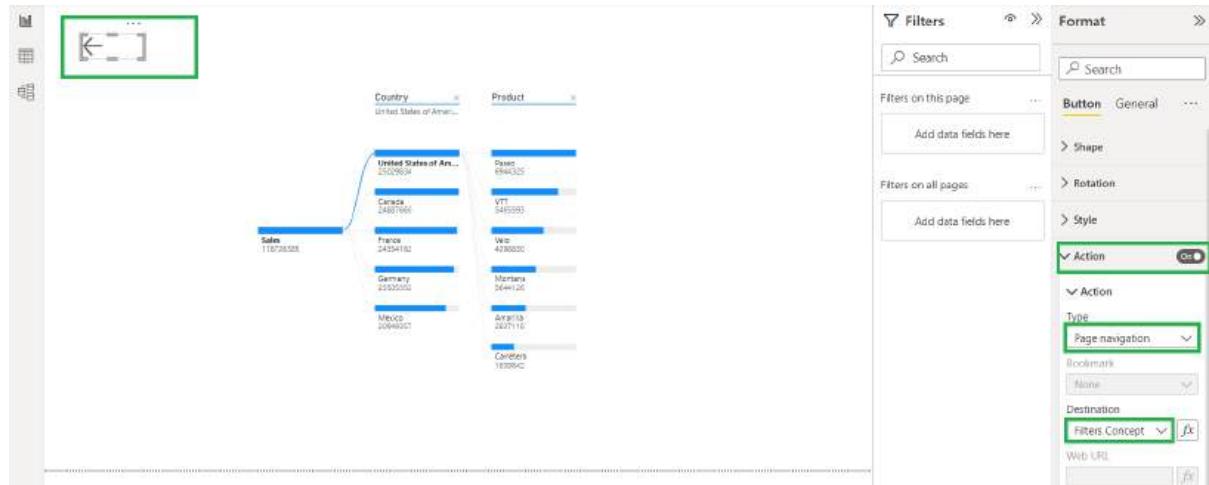
Take Button from “**Buttons**” option from “**Insert**” ribbon. I am taking “**Left arrow**” (**←**) button for example.



If we want to move one page to another page, we can use button and if you want to go to the particular website you can use the button. Etc.

We can perform an action by clicking on that button in power bi service, but if you want to perform an action in power bi desktop, we have to use “**CTRL+CLICK**” option.

Select “**Button**”-> Go to “**Format**” pane->Click on “**Action**” to “**ON**”-> Select “**Type**” as “**Page Navigation**” -> Destination as “**Filters concept**” page. Then press control button and click on “**Left Arrow**” button. It will take us to the “**Filters concept**” page.

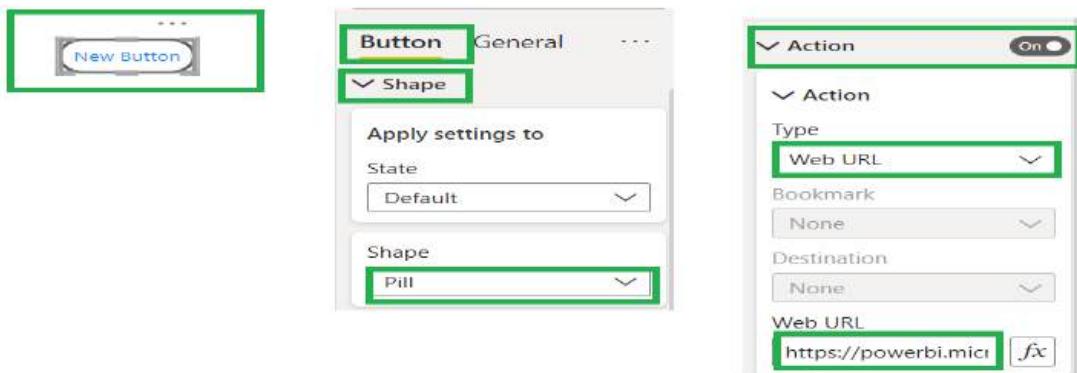


Most of the times; we will use blank button in our report.



If we want to perform an action, go to “**Format**” and I want to change shape as “Pill” and “Action” should be “ON” and I put type as “Web URL”, and I have pasted <https://powerbi.microsoft.com/en-us/> in that. Button has been created.

To see the action of that button press the “**CTRL+CLICK**”. Then automatically it will take us to <https://powerbi.microsoft.com/en-us/> in new tab.



The screenshot shows the Microsoft Power BI website. At the top, there are three tabs: "Create buttons in Power BI report", "Data Visualization | Microsoft Power BI", and "Data Visualization | Microsoft Power BI". The URL in the address bar is "powerbi.microsoft.com/en-us/". The main heading is "Find clarity when you need it most" with the subtext "Empower team members to discover insights hidden in your data with Microsoft Power BI.". Below the heading are two buttons: "Start free >" and "Have an account? Sign in >".

For example, I am taking “Page Navigation” button. By default it will show you all the pages as button. If you want to go to the particular page, click on that corresponding button pressing “CTRL” button.

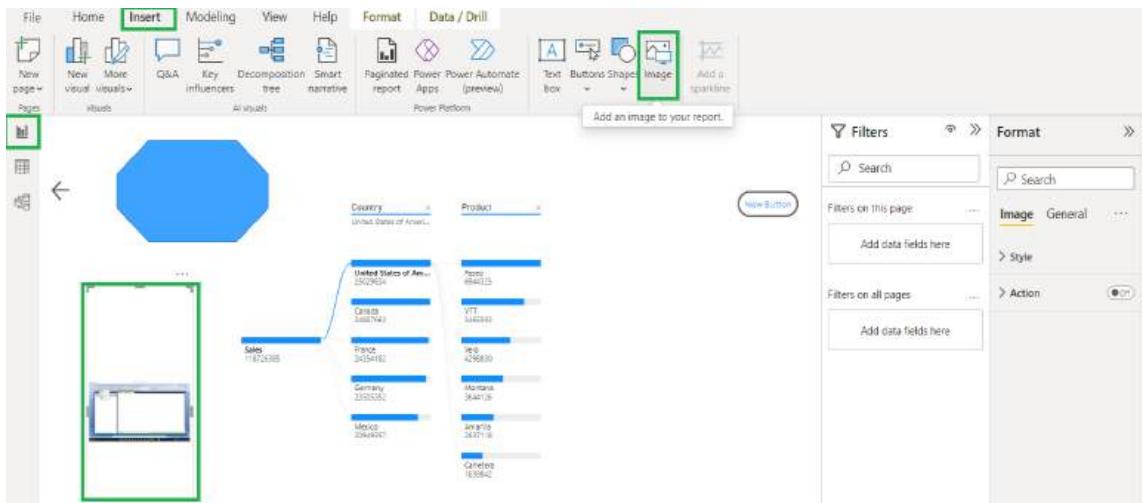
The screenshot shows the Microsoft Power BI desktop application. The ribbon is visible at the top with the "Insert" tab selected. In the center, there is a chart visual showing sales data by country. On the right side, the "Page navigator" pane is open, displaying a list of pages from "Second" to "Page 1" to "Page 8". Each page name is a button. A green box highlights the "Page 1" button. Below the page navigator, there are "Navigator" and "Page navigator" buttons. The "Page navigator" button is also highlighted with a green box.

- **Shapes:** If you want to arrange the visuals in proper order, then you can do that by using shapes. That means you can add a shape to the report if required.

The screenshot shows the Microsoft Power BI desktop application. The ribbon is visible at the top with the "Insert" tab selected. In the center, there is a chart visual showing sales data by country. A blue octagonal shape has been placed on the canvas. The "Shapes" icon in the ribbon is highlighted with a green box. To the right, the "Format" pane is open, showing options for "Basic Shapes" like rectangles, circles, and octagons. An octagon icon is highlighted with a green box. The "Format" pane also includes sections for "Filters", "Search", "Shape", "Rotation", "Style", and "Action".

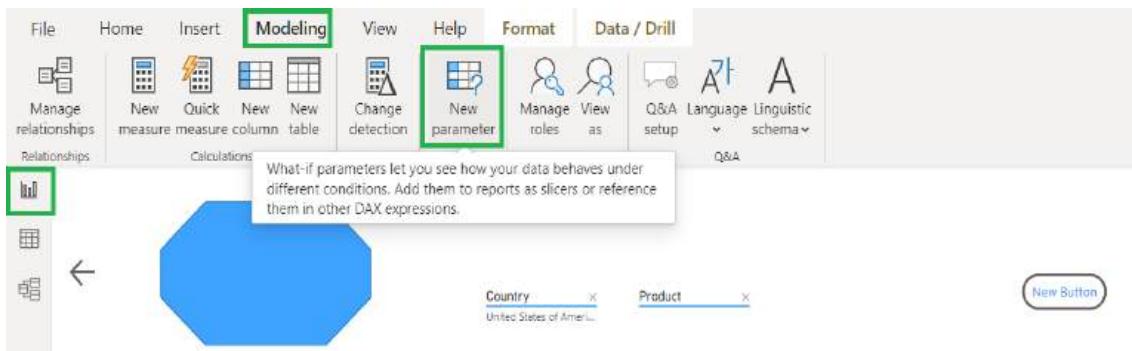
If you want to change the color of the shape, you can do that, you can change the size, rotation, style and action if you want.

- **Image:** If you want to insert an image from local system to the power bi desktop, you can click on “Image” option from “Insert” ribbon. It will ask you the image name from where you want to fetch that.



❖ **Modeling Tab:**

- **New Parameter:** This is also called as “**What if**” parameter. It can easily give the ability to dynamically transform your data. Mainly it is used for scenario base testing and analysis.



Ex)

Suppose I am taking data regarding the loans of a customer and need to calculate **EMI** per month for everyone.

Table Name: Loan Table

Customer Name	Location	Loan
Bhargav	Visakhapatnam	30000
Venkat	Vijayawada	50000
Sudha	Rajahmundry	80000
Ramesh	Narsipatnam	100000
Mark	Benguluru	120000
Narayana	Machilipatnam	90000

If you click on “New Parameter” then it will pop up a new window.

What-if parameter

Name	<input type="text" value="EMI Simulator"/>
Data type	<input type="text" value="Whole number"/>
Minimum	<input type="text" value="0"/>
Maximum	<input type="text" value="20"/>
Increment	<input type="text" value="1"/>
Default	<input type="text"/>
<input checked="" type="checkbox"/> Add slicer to this page	
<input type="button" value="OK"/> <input type="button" value="Cancel"/>	

Here, we need to provide name of the what-if parameter, data type and min value and max value of that parameter and increment value. Then click on **ok**. Once click on ok “**EMI Simulator**” created as a table.

It will add a slicer visual in the current page and it will use generateseries function to generate the values from 1 to 20. And also one new measure will be implicitly created.

EMI Simulator=Generateseries(0,20,1)

The screenshot shows a Power BI slicer visual. The title bar says "EMI Simulator". The list of items in the slicer is: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20.

Measure: EMI Simulator Value = `SELECTEDVALUE('EMI Simulator'[EMI Simulator])` → It is created by default from the system itself.

Now, I am creating “**new measure**” called “**EMI Amount**” on “Loan Table” data.

`EMI Amount = SUM('Loan Table'[Loan])*1*'EMI Simulator'[EMI Simulator Value]/100`

The screenshot shows the Power BI desktop interface. On the left, there is a visual with a green border containing a small chart and some text. In the center, there is a table with columns: Customer Name, Location, Loan, EMI Amount, and EMI Simulator Value. The table data is as follows:

Customer Name	Location	Loan	EMI Amount	EMI Simulator Value
Mark	Bengaluru	120000	6,000.00	5
Narayana	Machilipatnam	90000	4,500.00	5
Ramesh	Narsipatnam	100000	5,000.00	5
Sudha	Rajahmundry	80000	4,000.00	5
Venkat	Vijaywada	50000	2,500.00	5
Bhargav	Visakhapatnam	30000	1,500.00	5
Total		470000	23,500.00	5

On the right, there is a 'Fields' pane with a search bar and a list of fields: EMI Simulator, EMI Simulator Value, Loan Table, and EMI Amount.

Note: If you want, you can use “fields” option in what-if parameter to create a parameter. In that you have to select the column from list of the data sets.

❖ View Tab:

We can see and use themes from the available themes from the view tab. It is the background of the report.

The screenshot shows the Power BI desktop interface with the 'View' tab selected. The 'Themes' section on the left displays various color schemes. The 'Page options' section on the right includes buttons for 'Gridlines', 'Snap to grid', 'Lock objects', 'Filters', 'Bookmarks Selection', and 'Show pane'. A preview of the report with a purple theme is shown on the right.

The preview shows a table with the same data as the previous screenshot:

Customer Name	Location	Loan	EMI Amount	EMI Simulator Value
Mark	Bengaluru	120000	6,000.00	5
Narayana	Machilipatnam	90000	4,500.00	5
Ramesh	Narsipatnam	100000	5,000.00	5
Sudha	Rajahmundry	80000	4,000.00	5
Venkat	Vijaywada	50000	2,500.00	5
Bhargav	Visakhapatnam	30000	1,500.00	5
Total		470000	23,500.00	5

- **Page View:** You can change the page view as “Fit to Page, Fit to width, Actual size”.

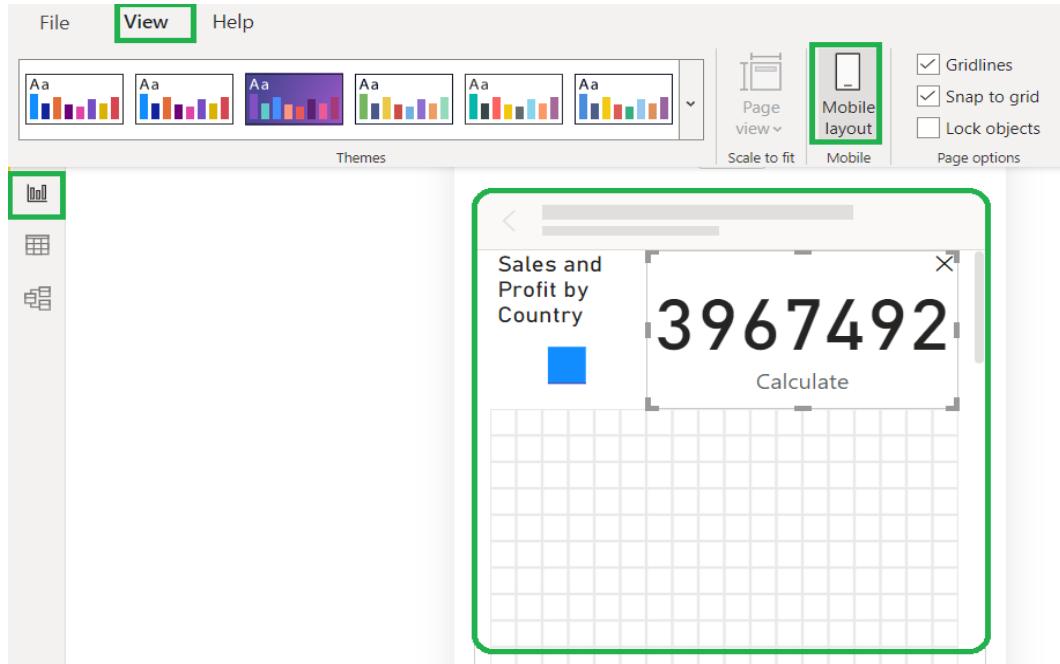
Fit to Page → Default view.

Fit to Width → Width will be increased for some extent and scroll bar will come

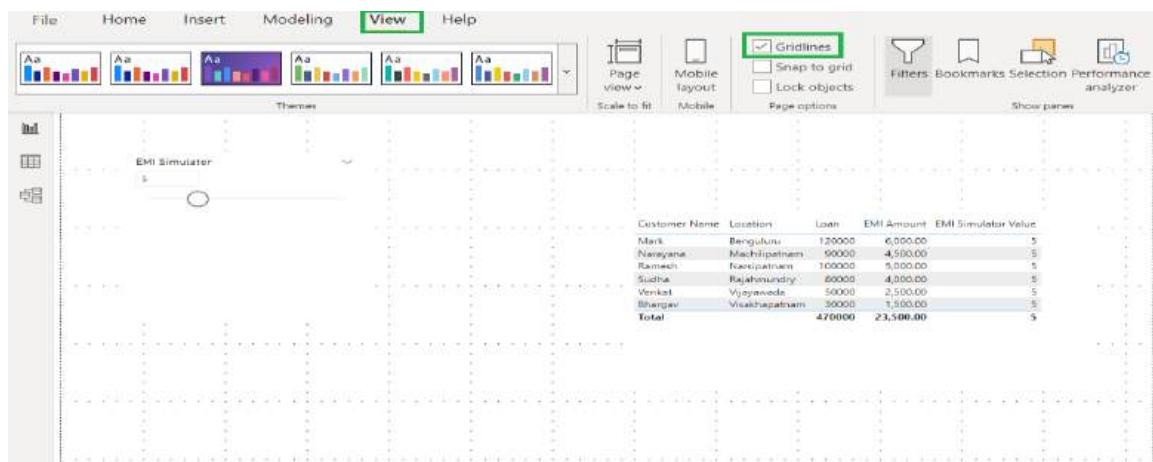
Actual size → It will be viewed as “A4” size. Vertical and Horizontal scroll bars will be displayed.

The screenshot shows the Power BI desktop interface with the 'View' tab selected. The 'Page options' section on the right includes buttons for 'Gridlines', 'Snap to grid', 'Lock objects', 'Filters', 'Bookmarks Selection', 'Performance', 'Sync analyzer', and 'Slicers'. The 'Page view' button is highlighted with a green border. The 'Fit to page' button is also highlighted with a green border.

- **Mobile Layout:** You can design your reports for mobile devices. That means we can change the desktop layout to mobile layout by simply clicking on this and arrange the objects in proper way.

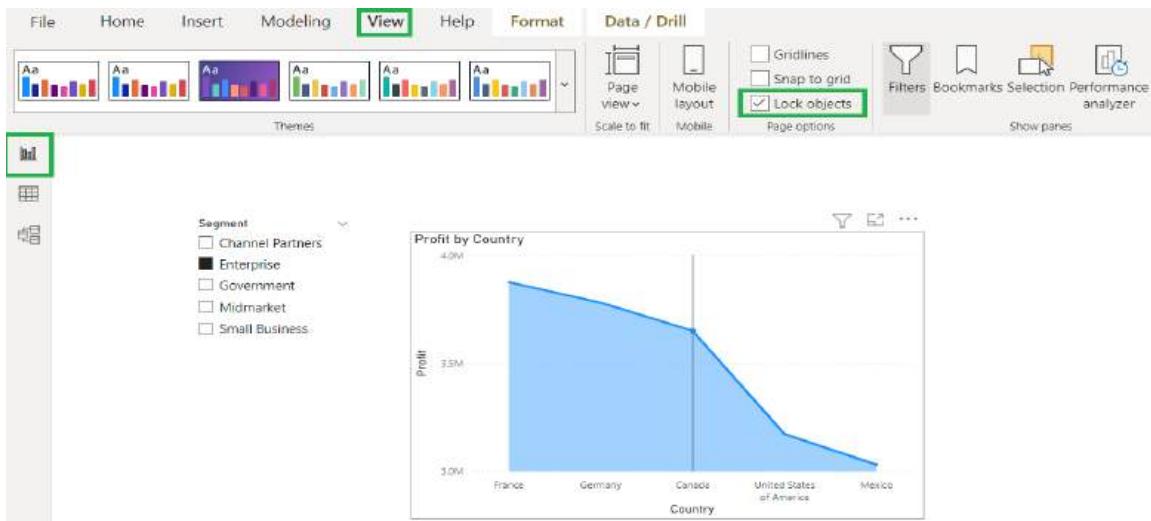


- **Grid Lines:** Background will be visible as grid lines. That means dotted lines. We can arrange objects based on the grid lines in proper place.



- **Lock objects:** If we don't want to move objects from one place to another place we have to use lock objects. Even in all the pages in a report objects will not be moved.

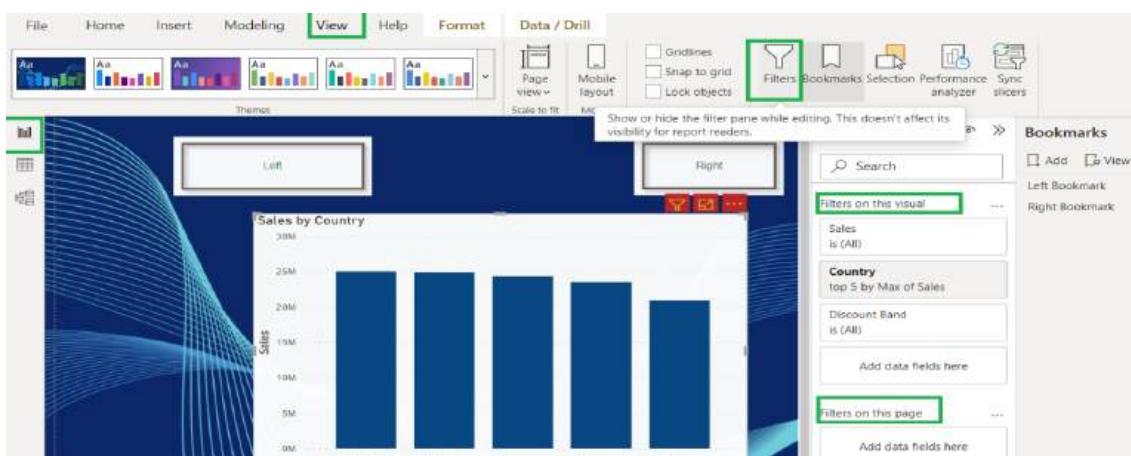
In the below visuals, no object moved from that location to another location.



- **Filters**: It will show or hide the filters pane while editing. This doesn't affect its visibility for report readers. It will filter the column values.

You can filter in visual level, you can filter in page level as well. There are 5 filters. By default we have only 3 filters, by selecting any visual we can see rest of the 2 filters.

- **Filters on this page**-----→Default
- **Filters on all pages**-----→Default
- **Filters on this visual**
- **Drill through**-----→Default
- **Drill-Down and Drill-Up**



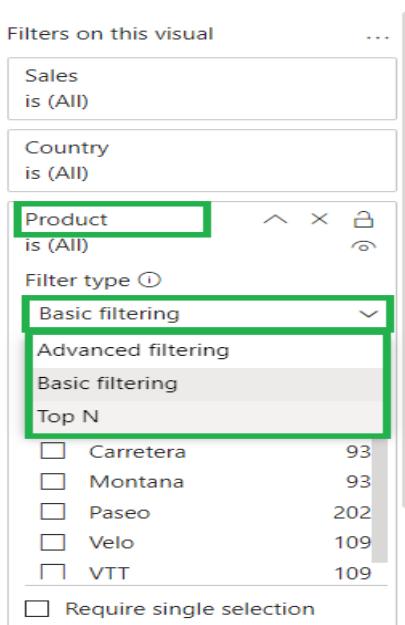
➤ **Filter on this visual:**

If you want to apply filters on visual level, first you have to select that visual and see the filters pane. There you can find the columns which you have used in that visual. You can filter their values in that filters pane.



If you expand any column, you will see the values of that column and you will see the filter type as well.

Filter type having three options. Those are i) Basic filtering ii) Advanced filtering iii) Top N



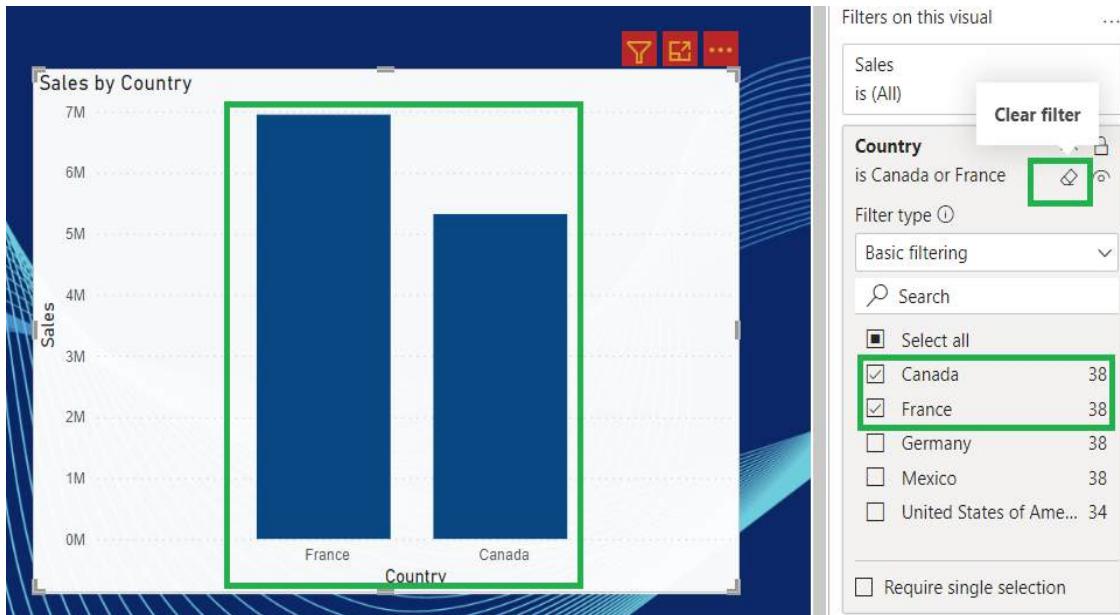
i) Basic Filtering: If you choose this, it will show you the list of values present in that column. If you select any value in that, automatically that corresponding value will be highlighted in the visual as well. See the below screenshot.



That means whatever the values you have selected, those related values bars only visible in the visual. This is called filter on this visual.

It will apply only for that visual only. If you don't select any value, then all values will be visible in the visual.

If you want to clear the filters what you have applied, then you have to click on "**Clear filter**" symbol. It looks like an eraser.

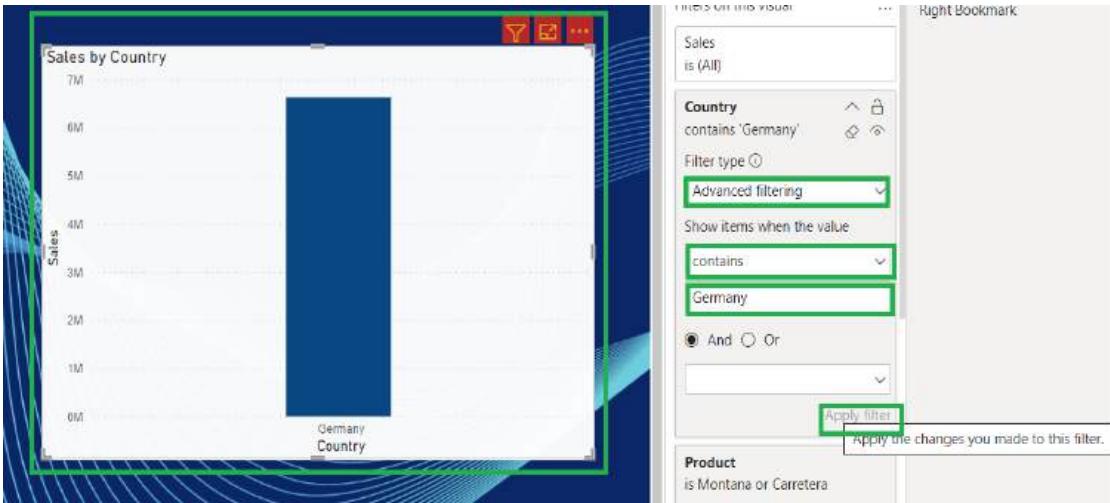


Note: On which column you have to apply filters, that column you need to drag and drop at "**Add data fields**" here option in the filters pane.

i)Advanced Filtering: If you want to apply based on condition, then values will be filtered based on that condition only and visible in the visual. Here you can choose the "**Operator**" and you have to give column value that you want to filter in the "**Text Box**".

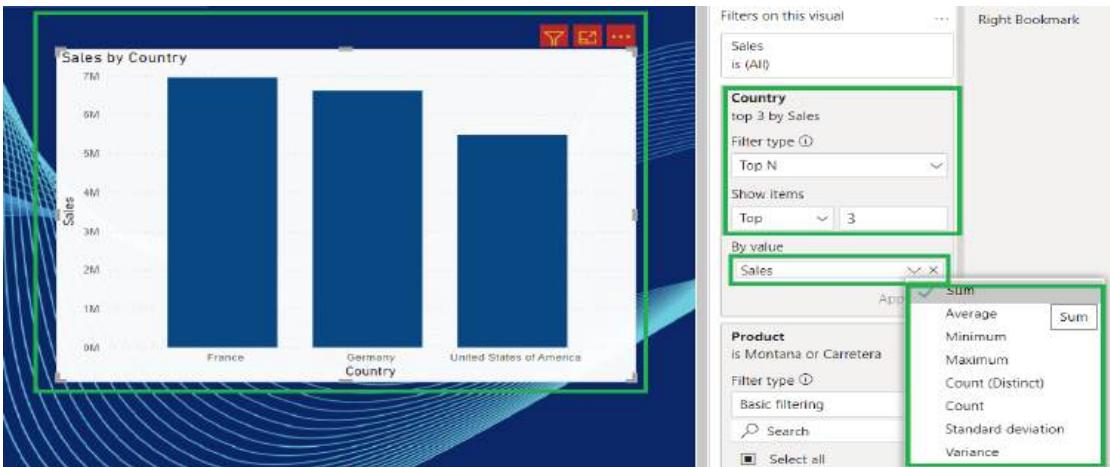
Suppose I am selecting “**Germany**” value from the “**Country**” column and given condition as “**Contains**”. After selection, then you have to click on “**Apply Filter**” option.

Note: For text type of filters it will show operators corresponding to the text. For numeric columns it will show the operators corresponding to the numeric values.



i) **Top N:** Specified number of top values will be visible in the visual by using this option.

Suppose we want to display top 3 highest sales countries. Then drag the sales column into the “**By value**” option. Give the value as 3. Then it will show you top 3 highest sales countries in the visual.

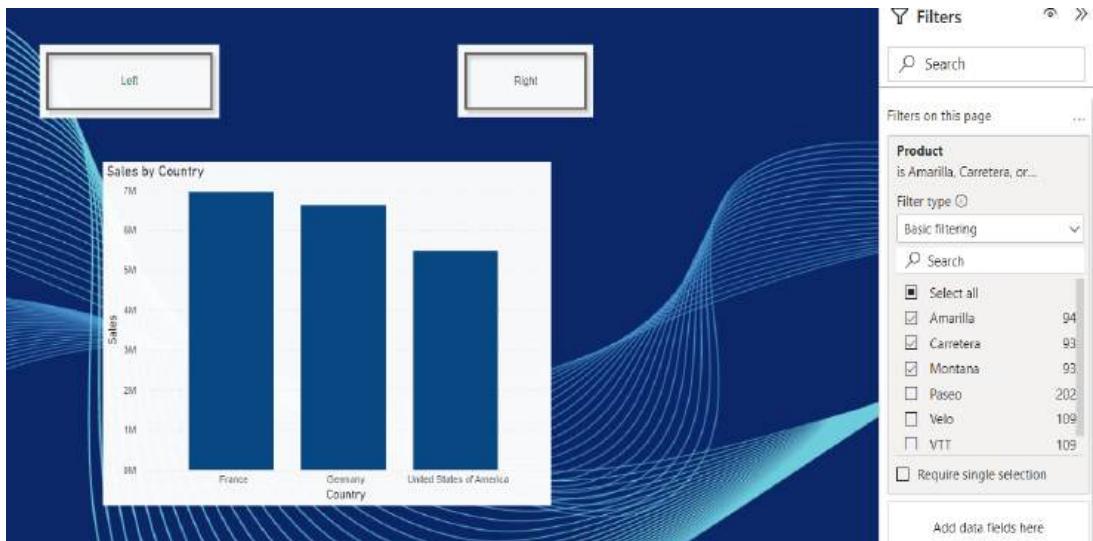


Note: For “**Top N**” and “**Advanced filtering**” you have click on “**Apply filter**” option after you chosen the condition or the value.

→ **Filter on this page:**

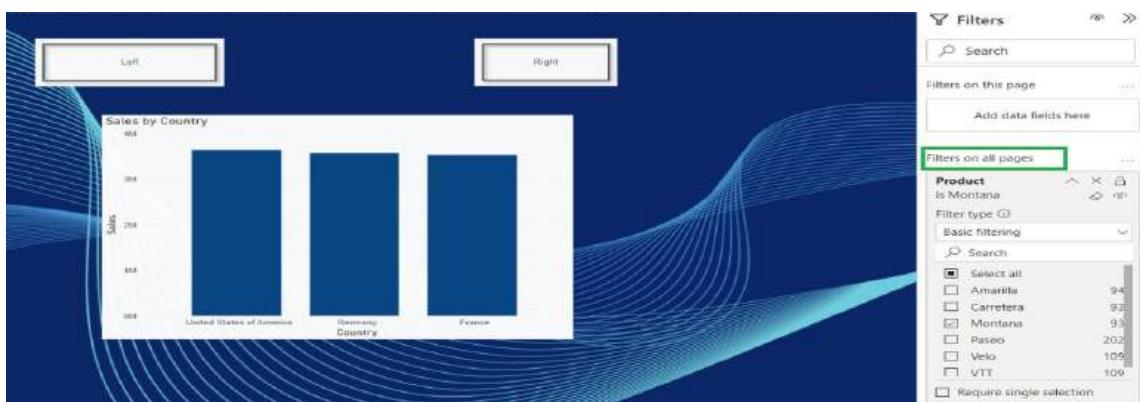
If you want to apply filters on entire page and for all visuals, then you have to use filters on this page option. To filter the data for the entire page, then you have to drag and drop the required column to there.

In this we have only two filter types. Those are i) Basic filtering and ii) Advanced filtering. “Top N” filtering is not available in “**Filter on this page**“ option and “**filter on all the pages**“ option. Whatever the process you have followed in the “**filter on this visual**” option, same will be applied here. But entire page will be affected.



→ **Filters on all pages:**

If you want to apply filters on all pages and for all visuals, then you have to use “**filters on all pages**” option. To filter the data for all the pages, then you have to drag and drop the required column to there.



In this filter option also, only two filter types available “**Basic and Advanced**”.

- **Bookmarks:** Showing or hiding a visual is called as Bookmark. If you want to use bookmarks, you have to call them from “**Buttons**”.

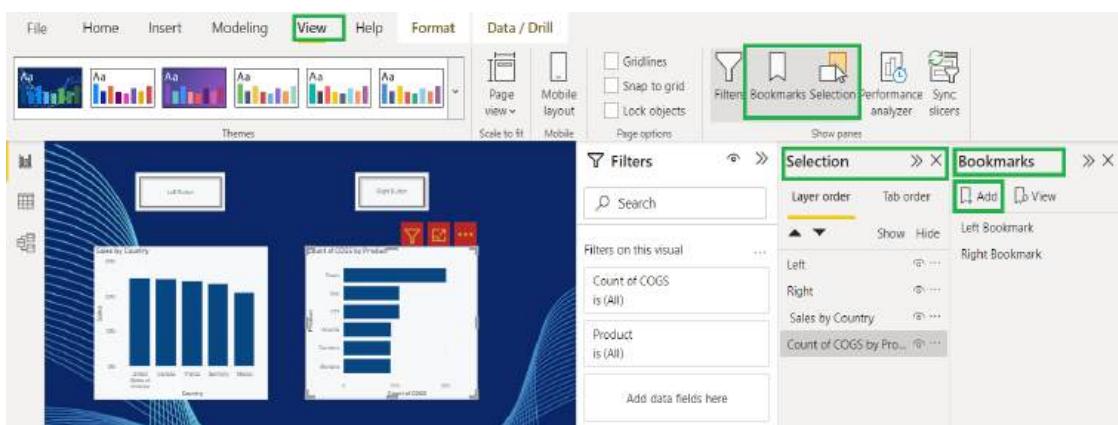
For example, I am taking two buttons for **Left** and **Right**. My scenario is if I press “**Left**” Button, left side visual should be visible and right side visual should be in disable mode, if I press “**Right**” button then right side visual will be visible and left side visual should be in disable mode.

For achieve this scenario, I am taking two bookmarks for these two buttons. One is “**Left Bookmark**” and “**Right Bookmark**”

Note: Along with Bookmarks, we have to use “**Selection**” option, where we have to select the visuals and buttons.

We have two take the book marks corresponding to the number of buttons available. In our case, we have two buttons, that's why I am taking two bookmarks.

If want to add books, click on “**Add**” option in the “**Book marks**” page. We have to give proper naming convention for the book marks by double click on the “**Book marks**” we have added.

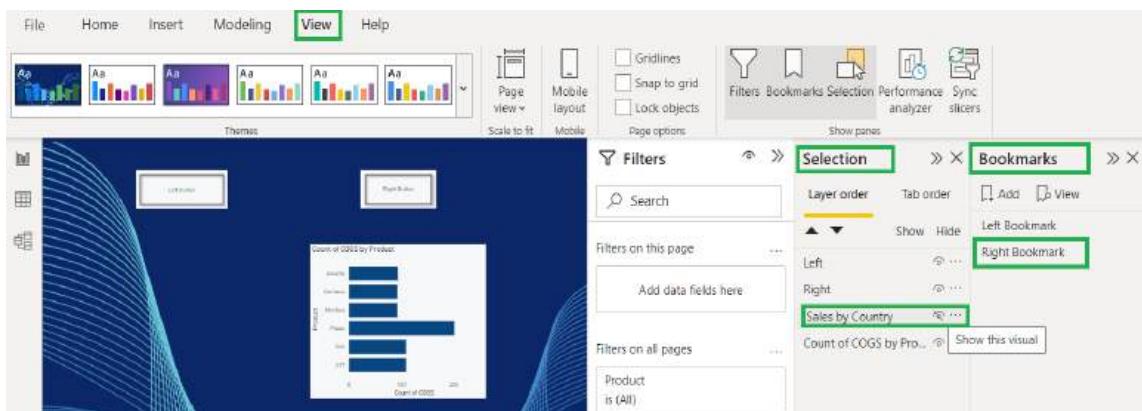
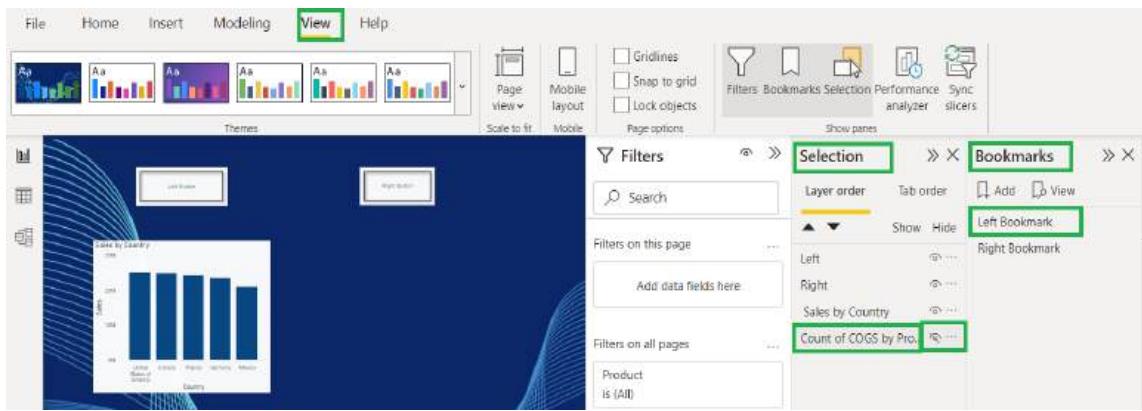


After giving names to the book mark, if I click “**Left Book mark**” corresponding “**Sales by country**” visual should be visible; remaining one should be in disable mode.

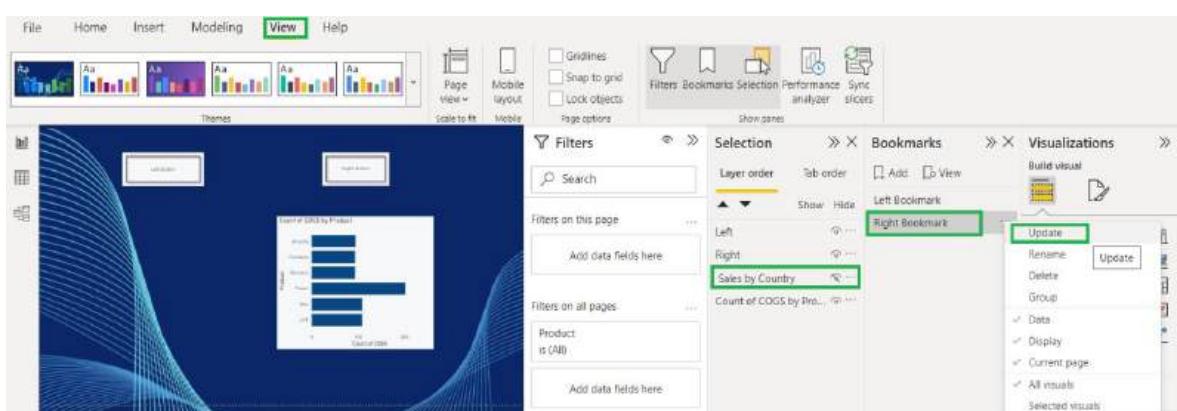
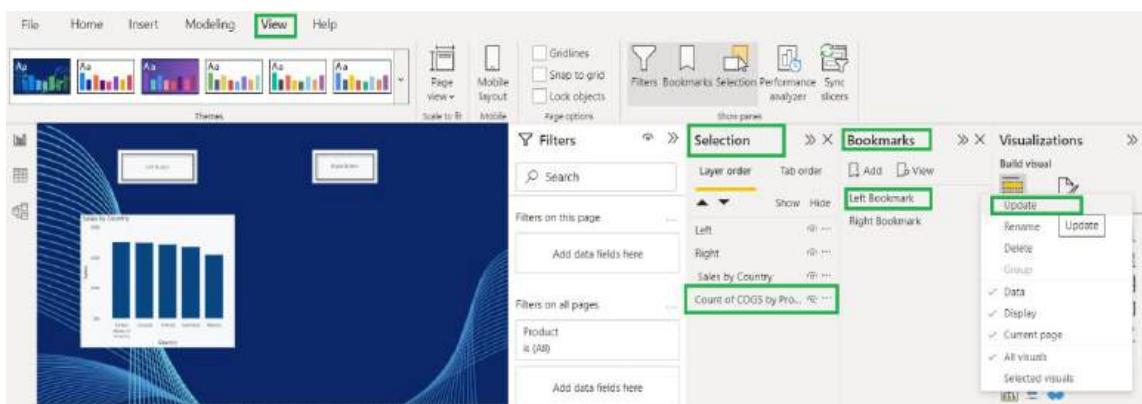
If I click “**Right Book Mark**” corresponding “**COGS by product**” visual should be visible; and remaining one should be in disable mode.

For Left button working, If we don't want to enable “**COGS by product**” visual then click on eye symbol on that visual name in the selection pane.

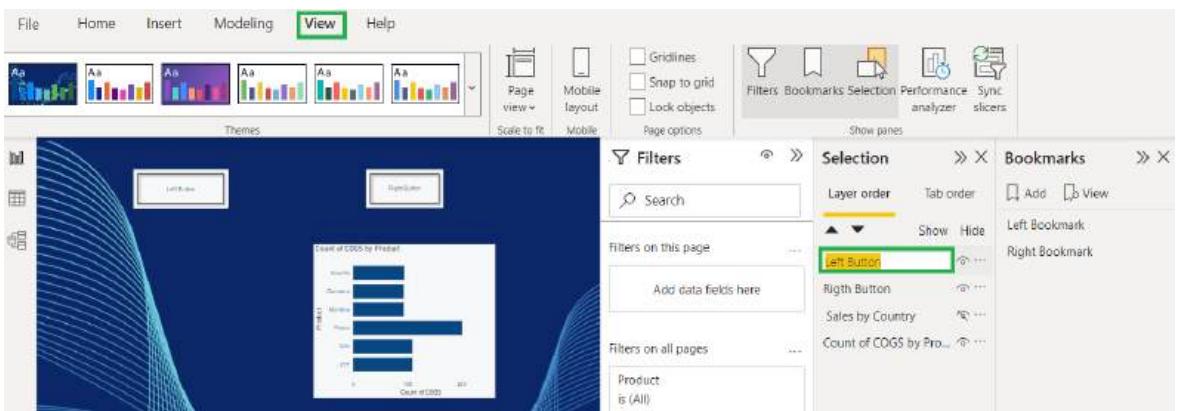
For Right button working; if we don't want to enable “**Sales by Country**” visual then click on eye symbol on that visual name in the selection pane.



Note: For each and every action performed on the bookmark that should be updated. To do that; go to the corresponding book mark and click on dots “...” and click on “Update”.



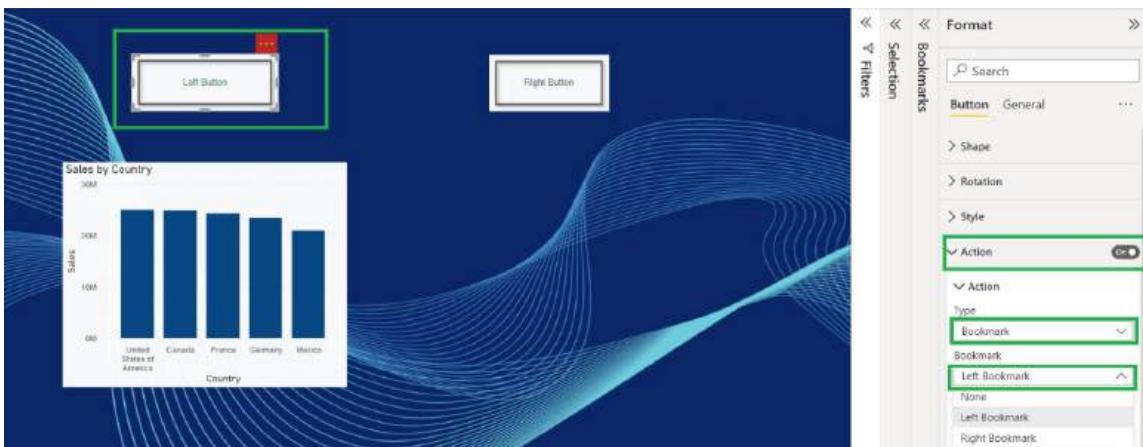
You have to give the proper naming convention for the buttons also by double clicking on the button in the selection pane.



By selecting button name in the selection, you can see that corresponding button is automatically selected.

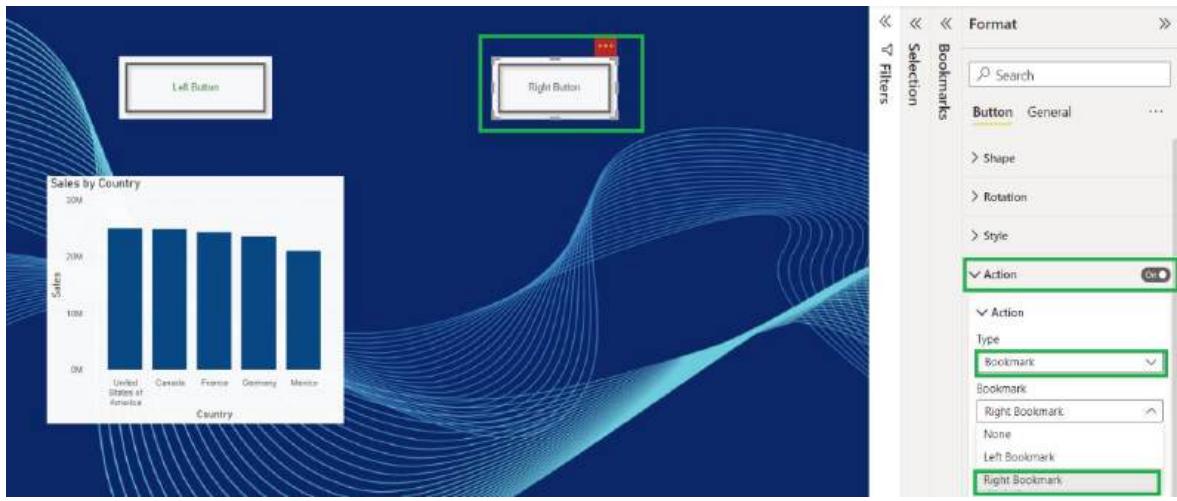
Left Button Action:

Navigation: Select “Left Button”->Go to “Format” pane->Action to be “ON”->Select type as “Bookmark”-> Call the “Book mark” from the drop down menu. For left button I want to call “**Left Book Mark**” here.



Right Button Action:

Navigation: Select “Right Button”->Go to “Format” pane->Action to be “ON”-> Select type as “Bookmark”-> Call the “Book mark” from the drop down menu. For right button I want to call “**Right Book Mark**” here.



Note: if we want to perform an action on button, we have to use “**CTRL+Click**”. If we click on “**Left Button**” corresponding left visual will be visible. If we click on “**Right Button**” corresponding right visual will be visible.



By doing like this, space might be wasted, to overcome this; we need to place visual in proper order and same properties should be copied to another visual. If you click on any button, corresponding visual can be visible at the same place.

So we have to note down the properties of left visual to right visual by just copy and paste.

Navigation: Select “**Left Visual**” -> Go to “**Visualizations**” ->**Format**->**General** ->click on drop down menu of “**Properties**”->copy the values of “**size**” that means “**Height**” and “**Width**”, copy the values of “**Position**” that means “**Horizontal**” and “**Vertical**” to the note pad. Paste these values for the “**Right Visual**” in the properties section.

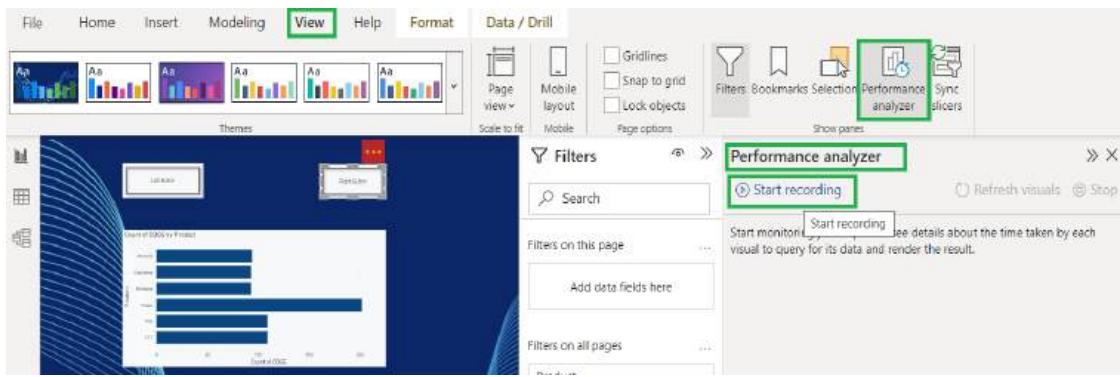


Once properties copied, then if you check by applying “CTRL+CLICK” on any button among them, corresponding visual will be visible at the same place only.

- **Performance Analyzer:** In Power BI Desktop select the **View** ribbon, and then select **Performance Analyzer** to display the Performance Analyzer pane.

The Performance Analyzer helps us to understand how the performance of the dashboard can be optimized.

Performance Analyzer is an easy and accessible way to track the performance of a Power BI report during the development process in Power BI Desktop.



If you want to see how much time is consuming for the report. Then we have to click on “**Start recording**”. Then click on “**Refresh Visuals**” option. The Performance Analyzer captures the Duration time in milliseconds for each visual on the page.

Name	Duration (ms)
⌚ Recording started (31-10-2022 21:43:39)	-
⌚ Refreshed visual	-
⊕ Left Button	20
⊕ Rigth Button	19
⊕ Count of COGS by Product	196

Again click on the Refresh visuals button in the Performance Analyzer pane. After refreshing we see the duration of the visuals for the second execution. The execution time of every visual has been reduced since the first execution.

Name	Duration (ms)
⌚ Recording started (31-10-2022 21:43:39)	-
⌚ Refreshed visual	-
⊕ Left Button	20
⊕ Rigth Button	19
⊕ Count of COGS by Product	196
⌚ Refreshed visual	-
⊕ Left Button	22
⊕ Rigth Button	21
⊕ Count of COGS by Product	117
⌚ Refreshed visual	-
⊕ Left Button	28
⊕ Rigth Button	27
⊕ Count of COGS by Product	128

Performance Analyzer gets all details of all components in the dashboard.

Performance analyzer	
(Start recording) Refresh visuals (Stop)	
Clear Export	
Name	Duration (ms)
Recording started (31-10-2022 21:43:39)	-
Refreshed visual	-
Left Button	20
Visual display	9
Other	11
Copy query	
Rigth Button	19
Count of COGS by Product	196
DAX query	44
Visual display	10
Other	142
Copy query	
Refreshed visual	-

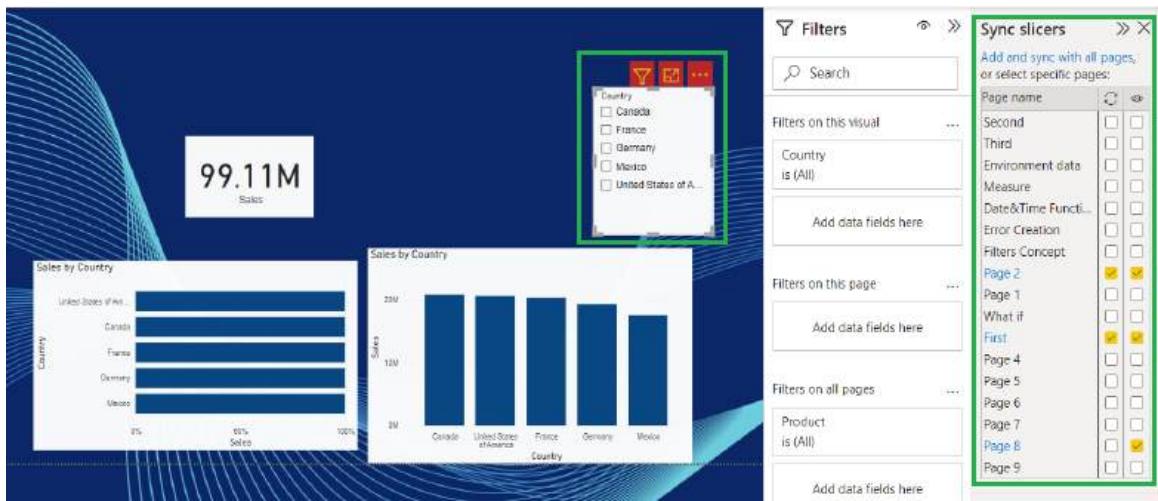
We have to identify most time consuming processes and take an action accordingly. Suppose if we have used dax function in the visuals, try to replace them by using transformation.

Always try to use the less time consuming dax functions for better performance.

- **Sync Slicers:** Synchronize slicers and control their visibility across pages in your report.

That means select any slicer in one page and click on “**sync slicers**” option. Then automatically same type of slicers in all pages will be synched.

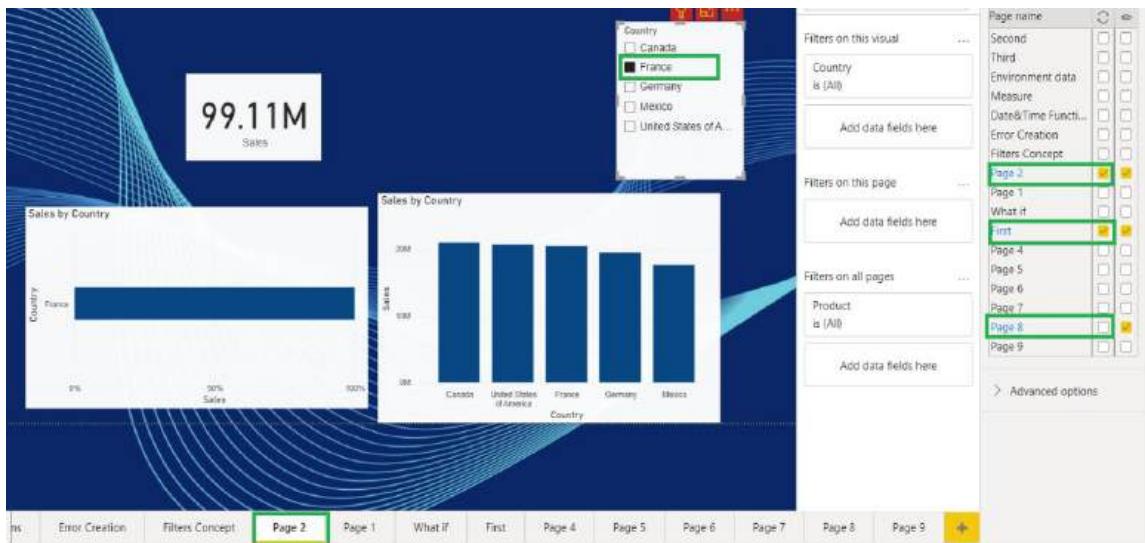
Suppose if we want to see how many pages having the same type of slicers visuals are there, select any slicer in one page then automatically it will display all the pages and show the slicers available pages also. There you can hide or unhide the visual. See the below screenshot.



In the above screen, you can see two lines in “Sync slicers” pane. Those are visibility line and sync line.

In the above screen, Slicer is showing in three pages. One is “First” and second one is “**Page 2**” and third one is “**Page 8**”. If you want to sync both slicers, you have to click on check boxes in sync line that corresponds to the visibility line.

Suppose I am selecting slicers in “**Page 2**” and “**First**” but not in “**Page 8**”. That means if you select any slicer value in either “**Page 2**” or “**First**”. Both pages will be synced. But it will not reflect in “**Page 8**”



The screenshot shows a Power BI dashboard with multiple pages. A green box highlights the 'First' tab in the ribbon. The right side of the interface displays the 'Edit Interactions' pane, which lists various pages and their filter configurations. The 'First' page has several filters applied, including 'Country Is (All)', 'Product Is (All)', and 'Year All_Selected All'. Other pages like 'Page 2', 'Page 3', 'Page 4', 'Page 5', 'Page 6', 'Page 7', 'Page 8', and 'Page 9' also have filter configurations listed.

The screenshot shows a Power BI dashboard with a single page selected. A green box highlights the 'Page 8' tab in the ribbon. The right side shows the 'Edit Interactions' pane, which lists filter configurations for 'Page 8'. The 'Country Is (All)' filter is applied to the current page.

Note: You can specify the pages for syncing the slicers as per your requirement. Those pages only synched. Rest of the pages will not be synched.

- ❖ **Format or Data/DrillTab:** These options will be enabled only when you select any visual.
- **Edit Interactions:** It is used to decide whether you want to apply filter for a particular visual or not.

The screenshot shows a Power BI dashboard with the 'Format' tab selected in the ribbon. The interface displays several charts, including a bar chart for 'Sales by Country' and a pie chart for 'Sales and Count of Gross Sales by Country'. A large numerical value '119M' is prominently displayed on the screen. The 'Edit Interactions' pane is visible on the left, showing options for applying drill-down filters to the entire page or specific visual elements.

In this section, we have 3 options. Those are as follows

- ★ Filter

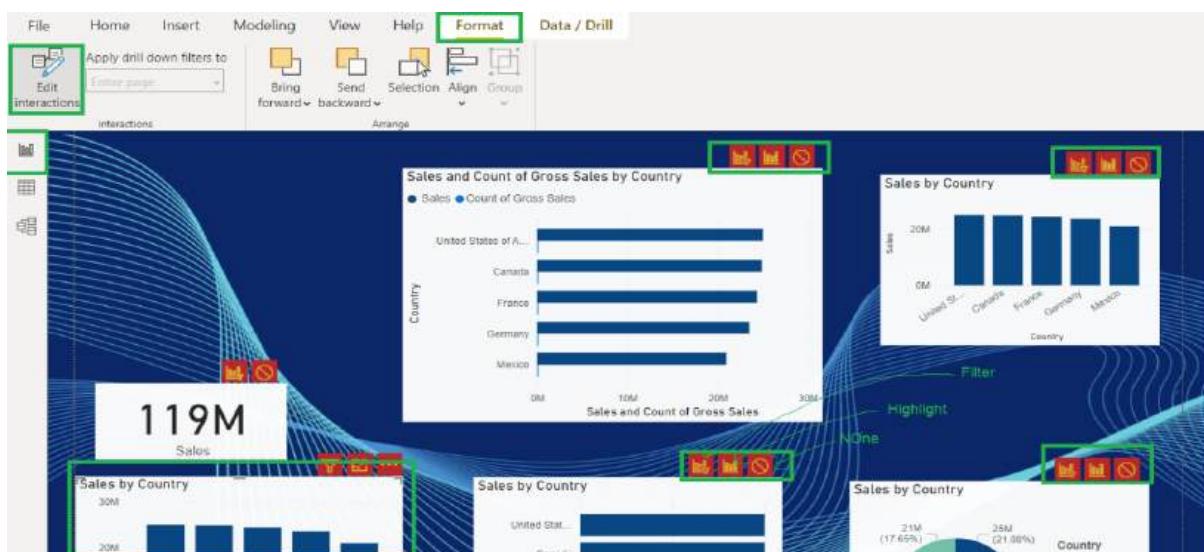
- None
- Highlight

Note: If we want to see the above three options, first, we have to select any visual in the current page. Then click on “**Edit interactions**”. Most of the visuals will have **Filter** and **none** options.

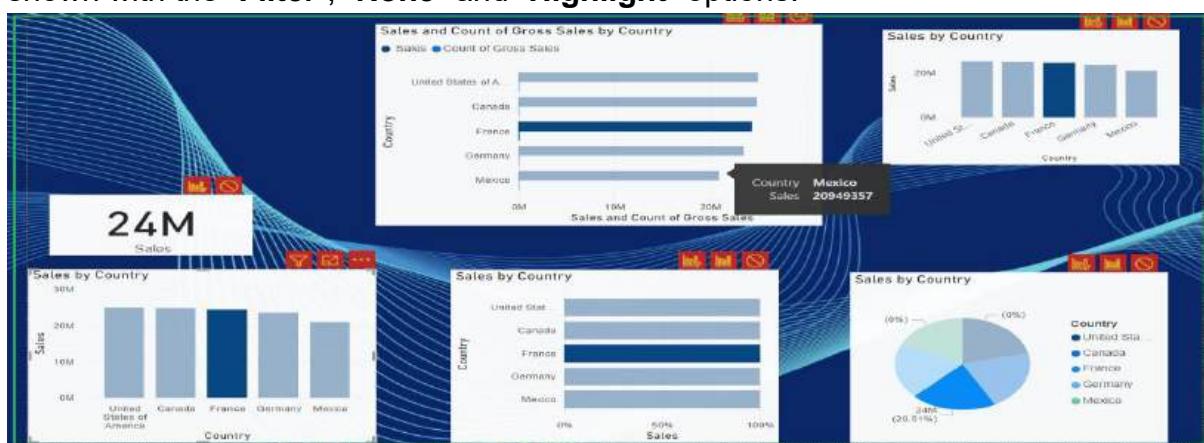
“**Highlight**” filter will be enabled for some of the visuals only. Those are for example, stacked column chart, stacked bar chart, pie chart, clustered bar chart etc.

Filter:

This option is nothing but selected value will be filtered in all the visuals. By default this option will be enabled.



Suppose I am taking 7 visuals in our page, I am selecting “**stacked column chart**” visual. Then I click on “**Edit Interactions**”. Then automatically rest of the visuals are shown with the “**Filter**”, “**None**” and “**Highlight**” options.



If I select any value in the “**Stacked column chart**”, automatically that value will be filtered in the corresponding visuals. Suppose I have selected “**France**” value. Then same value has been filtered in other visuals as well. See the above screenshot.

None:

If you enable this option in any visual, filters will not be applied on that visual.

That means whatever the value you have selected in one visual, that value will not be filtered on the other visual, because it is already **none** option is activated.

First select the “**Stacked column chart**”, then automatically it shows all visuals with the above three options.

Now, you can enable the “**none**” option for whichever visuals you don’t want to affect the value.

Once apply “**none**” option for a particular visual, then select any value in the “**Stacked column chart**”. If you observe value you have selected, that value will not be filtered on whatever the visuals having “**none**” option has enabled.



Here, for “**Stacked Bar chart**” and “**Clustered column chart**” visuals **none** option has enabled. I have selected “**Germany**” value un “**Stacked column chart**” that value has not been filtered in “**Stacked Bar chart**” and “**Clustered column chart**”.

Note: Whatever the visuals enabled “none” option, those visuals only will not be filtered. Rest of the visuals automatically filtered.

Highlight:

This option will be available for some of the visuals only. Like **stacked column chart**, **stacked bar chart**, **clustered bar chart**, **clustered column chart** and **pie chart** etc...

If you enable this option in any visual and if you select any value in other visual that corresponding value will be highlighted in this visual.

Suppose I am enabling “**highlight**” option for “**Pie chart**” and “**Clustered Bar Chart**” and I will select value “**France**” in “**Stacked Column Chart**”.

Then this “**France**” value will be highlighted in “**Pie Chart**” and “**Clustered Bar Chart**”. Highlighted value will be visible with dark color.



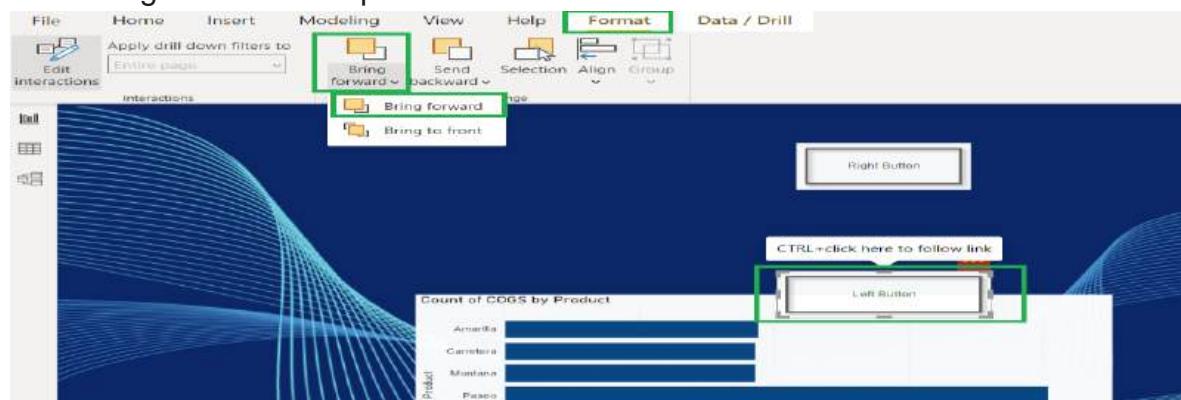
Note: If you enable filter option, then only filtered value will be visible, rest of the values will not be visible. But if you enable “**Highlight**” option, then selected value will be visible in dark color, rest of the values will be visible with light color.

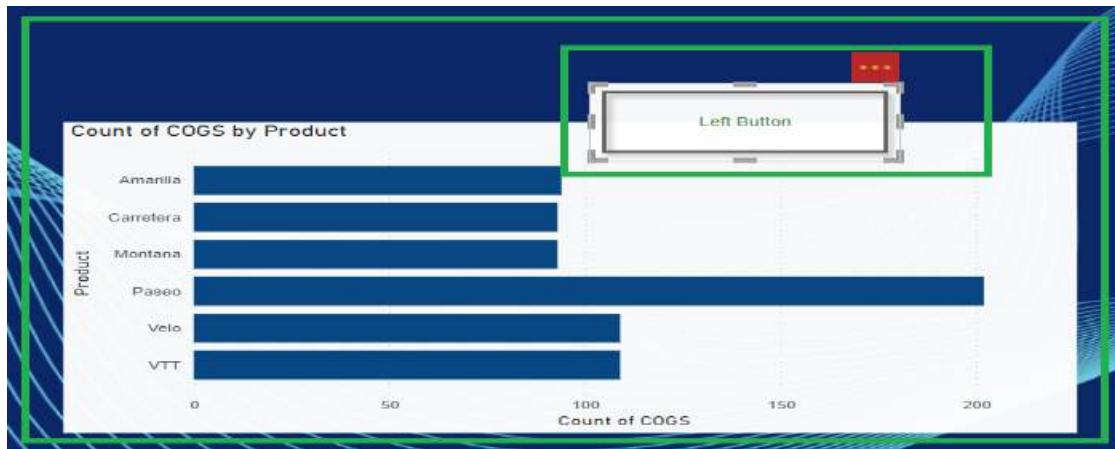
- **Bring forward:** It will bring the visual to the forward. That means if you select any visual that is in behind to another visual can be bring to the forward place. In this option, we have 2 options.
 - ★ Bring forward
 - ★ Bring front

Bring forward: It will bring the visual to the forward closer to the front. If you want to bring forward a visual you have to select that visual follow below navigation.

On the **Home** tab in the **Arrange** group, click the arrow next to or under **Bring Forward**, and then click **Bring Forward**.

Here, I have selected “**Left Button**” that is behind the “**Stacked column chart**” and I click on “**Bring forward**” option. Then automatically “**Left Button**” visual will be coming to the one step forward to the front.



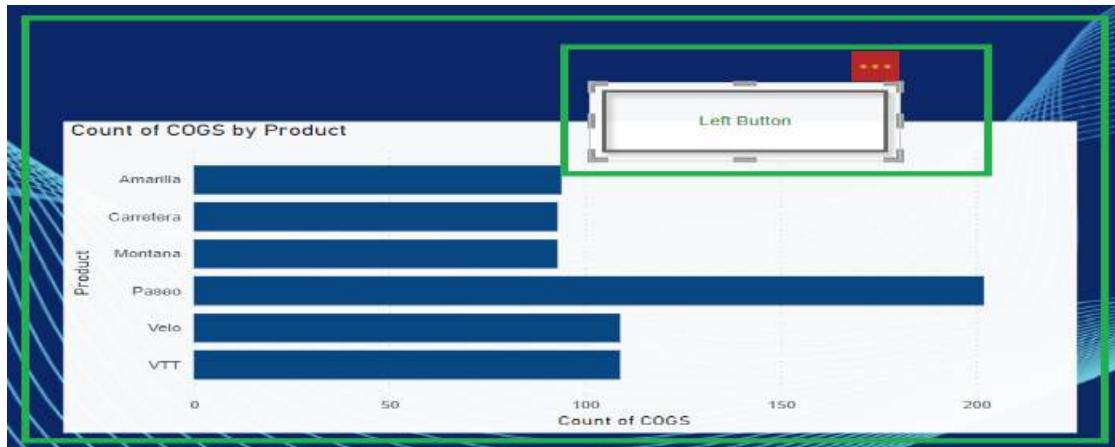


- ➊ Bring front

Bring front: It will bring the visual to the front place of the stack. If you want to bring front a visual you have to select that visual follow below navigation.

On the **Home** tab in the **Arrange** group, click the arrow next to or under **Bring Forward**, and then click **Bring Front**.

Here, I have selected "**Left Button**" that is behind the "**Stacked column chart**" and I click on "**Bring front**" option. Then automatically "**Left Button**" visual will be coming to the front of the stack.

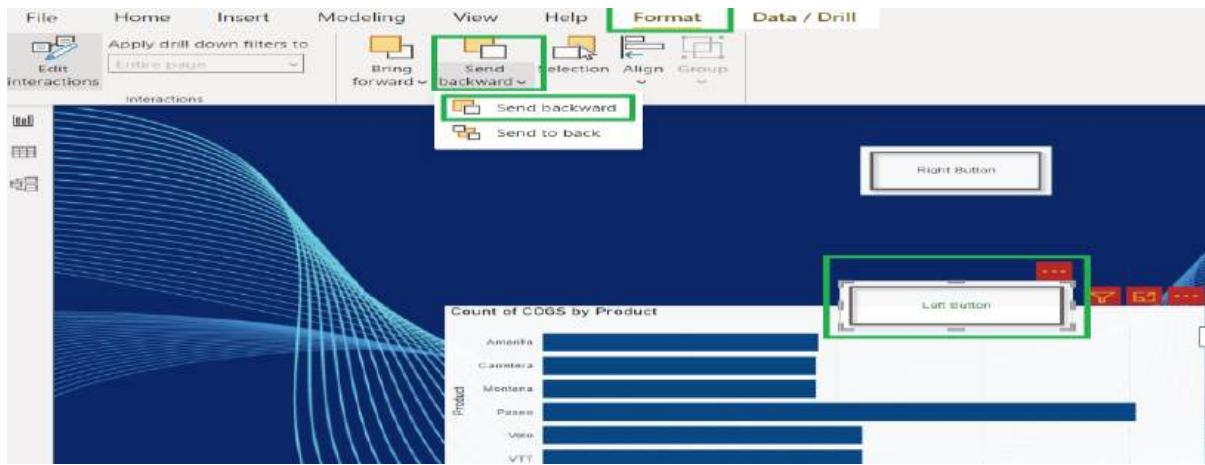


- **Send backward:** It will send an object to the backward. That means if you select any visual that is in front of another visual. That can be sent back to the backward position. In this option, we have 2 options.

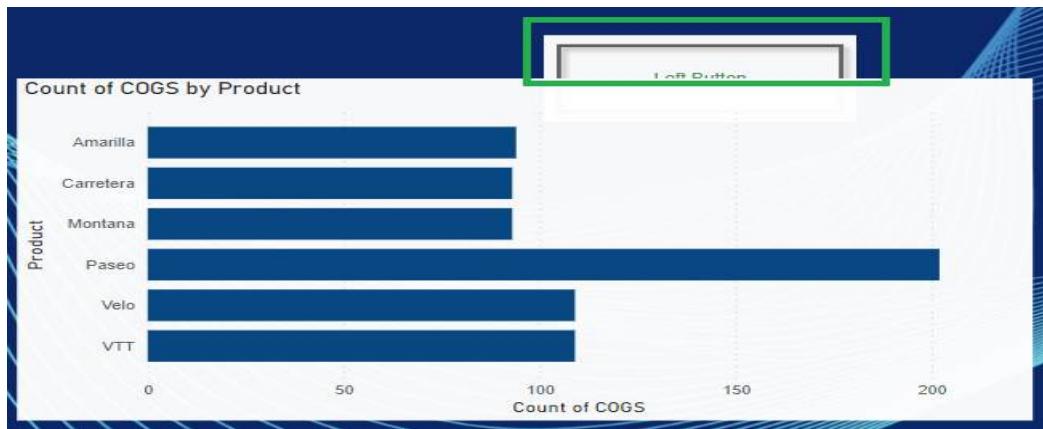
- ➊ Send backward
 - ➋ Send back

Send backward: It will send the visual one step toward the back. If you want to bring forward a visual you have to select that visual follow below navigation.

On the **Home** tab in the **Arrange** group, click the arrow next to or under **Send Backward**, and then click **Send Backward**.



Here, I have selected “**Left Button**” that is already in top place and click on “**Send Backward**”. Then it went one step towards the back.

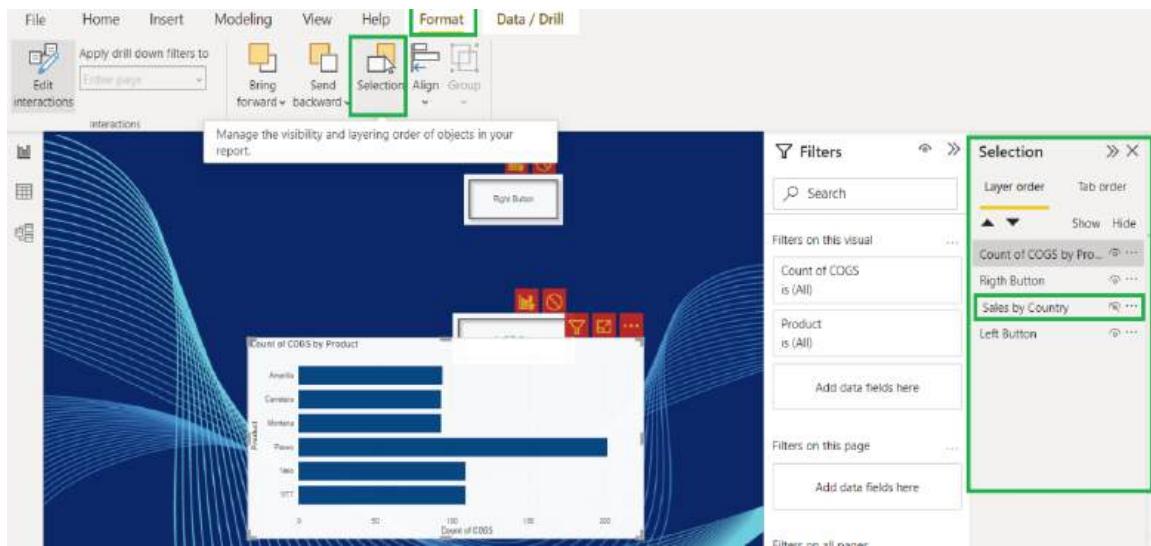


Send back: It will send the visual to the back that is last place. If you want to send a visual to the back position you have to select that visual follow below navigation.

On the **Home** tab in the **Arrange** group, click the arrow next to or under **Send Backward**, and then click **Send Back**.



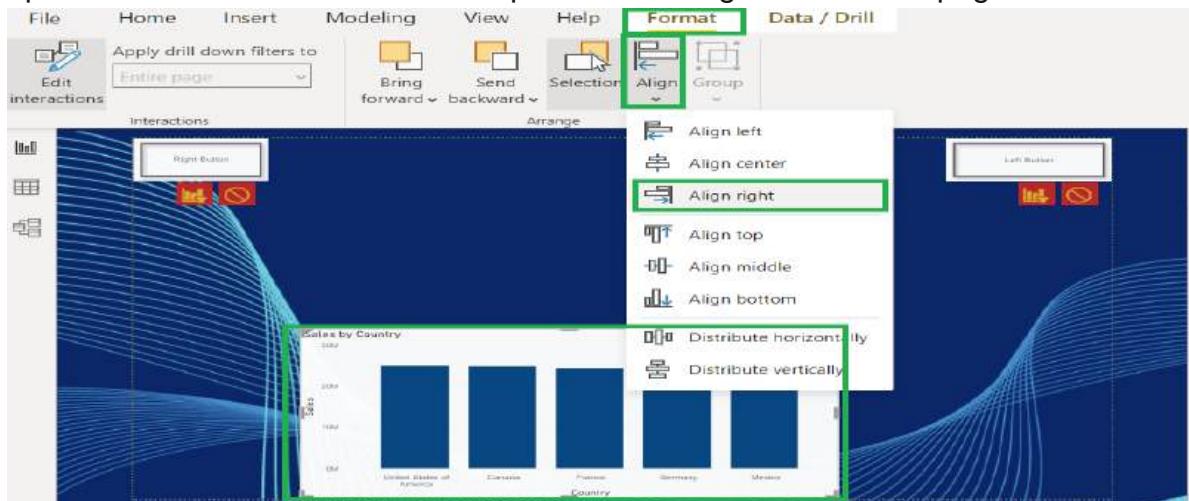
- **Selection:** It will manage the visibility and layering order of objects in your report. Most of the times we will use this option along with book marks. By using this option you can hide or unhide the visual by selecting.



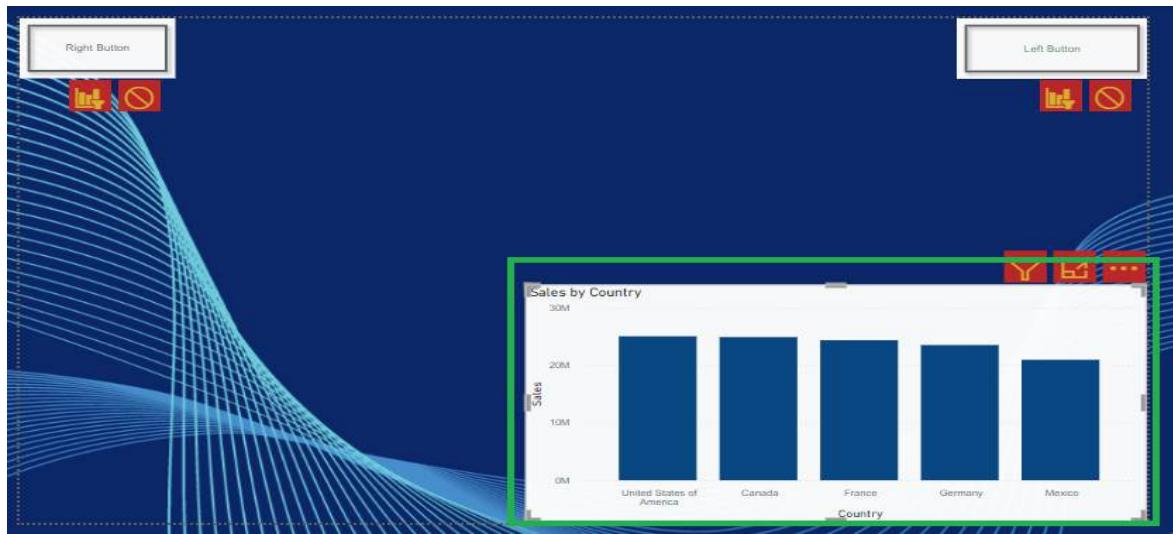
- **Align:** This will align the objects at the bottom, top, right, left, center or middle. That means we can arrange the visuals in an order. Select single visual or multiple visuals by pressing **CTRL** button.

Navigation: Select visual and go to “Format” option and click on drop down menu of “Align” and select the option that where you want to put the visual.

Suppose I have selected “Sales by country” visual and click on “Align right” option. Then selected visual will be placed in the right side of that page.



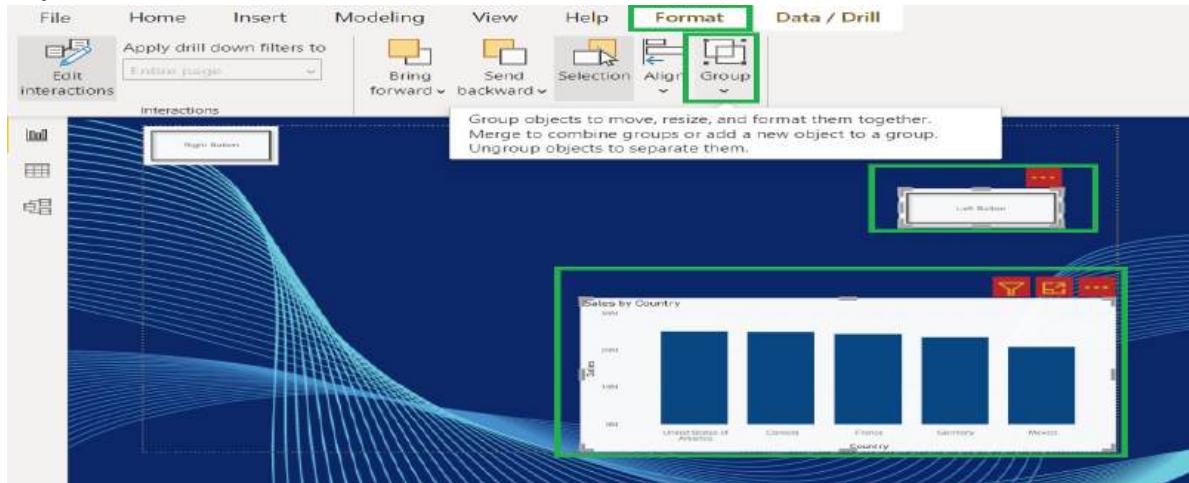
Result:

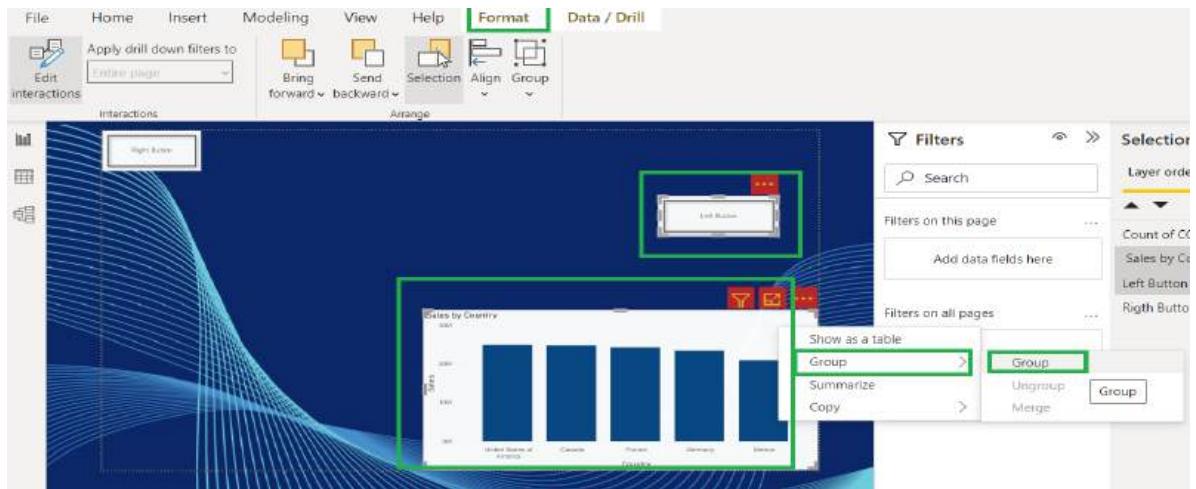


- **Group:** Group objects to move, resize and format them together. Merge to combine groups or add new objects to a group. Ungroup objects to separate them.

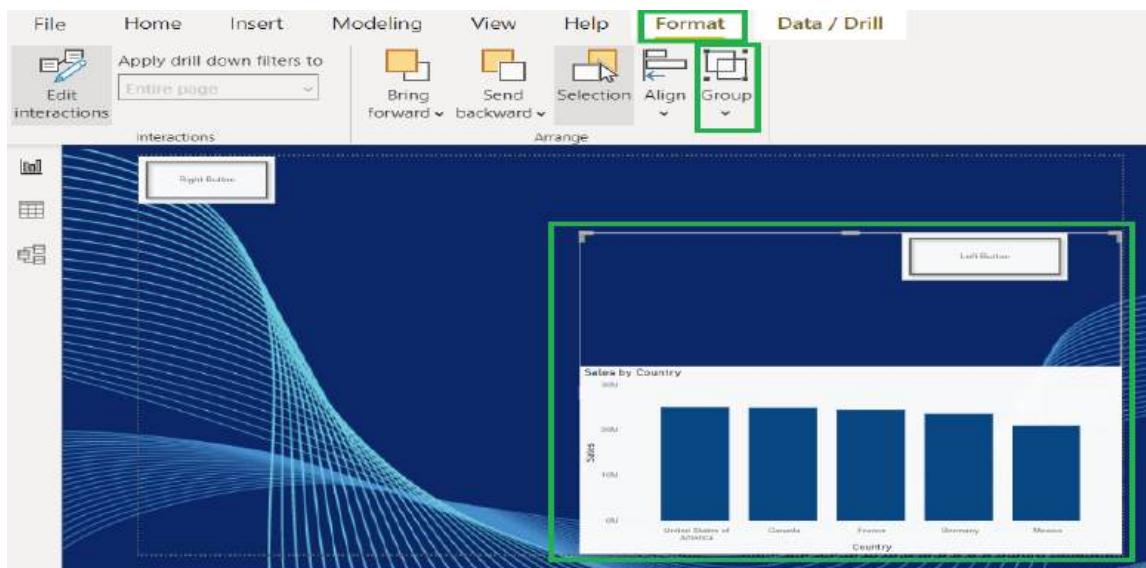
Navigation: Select the objects that you want to group and go to “Format” option and click on “Group” option. Then automatically selected objects will be grouped.

If you want to use “Group” option, you can directly right click on the selected objects.





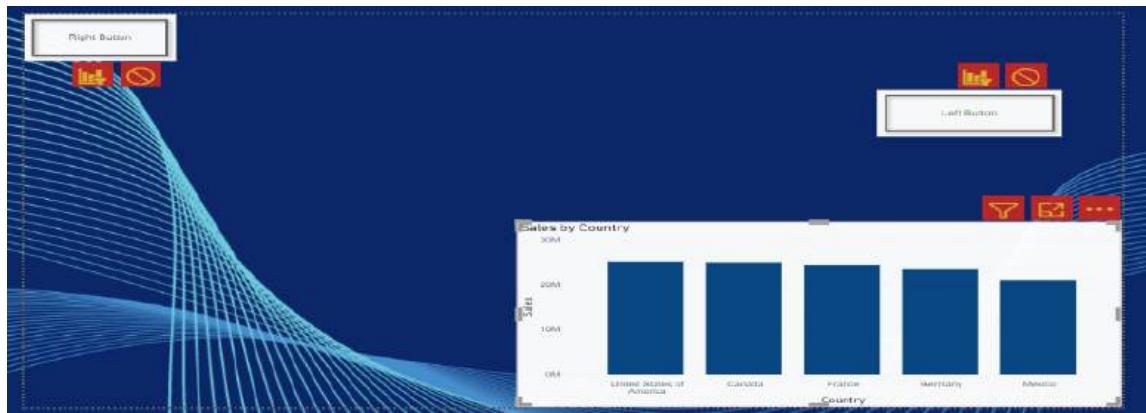
Result:



Ungroup: You can ungroup the grouped object by right click on the selected grouped object.

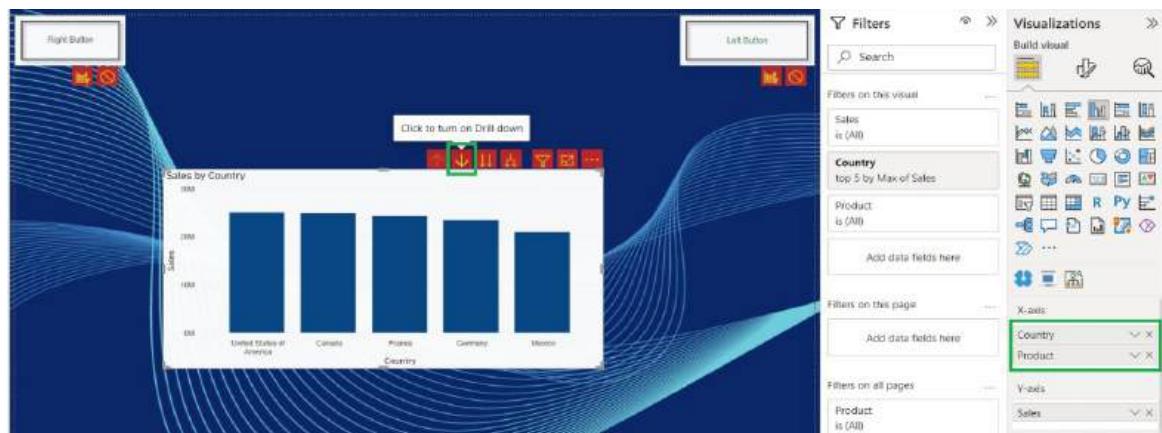


Result:



- ❖ **Data/Drill Tab:** These options will be enabled only when you select any visual.
- **Drill Down:** It will show the different hierarchies in the same visual only. That means if you select any visual and put more than one column in X-Axis, then you can see the “**Drill Down**” option.

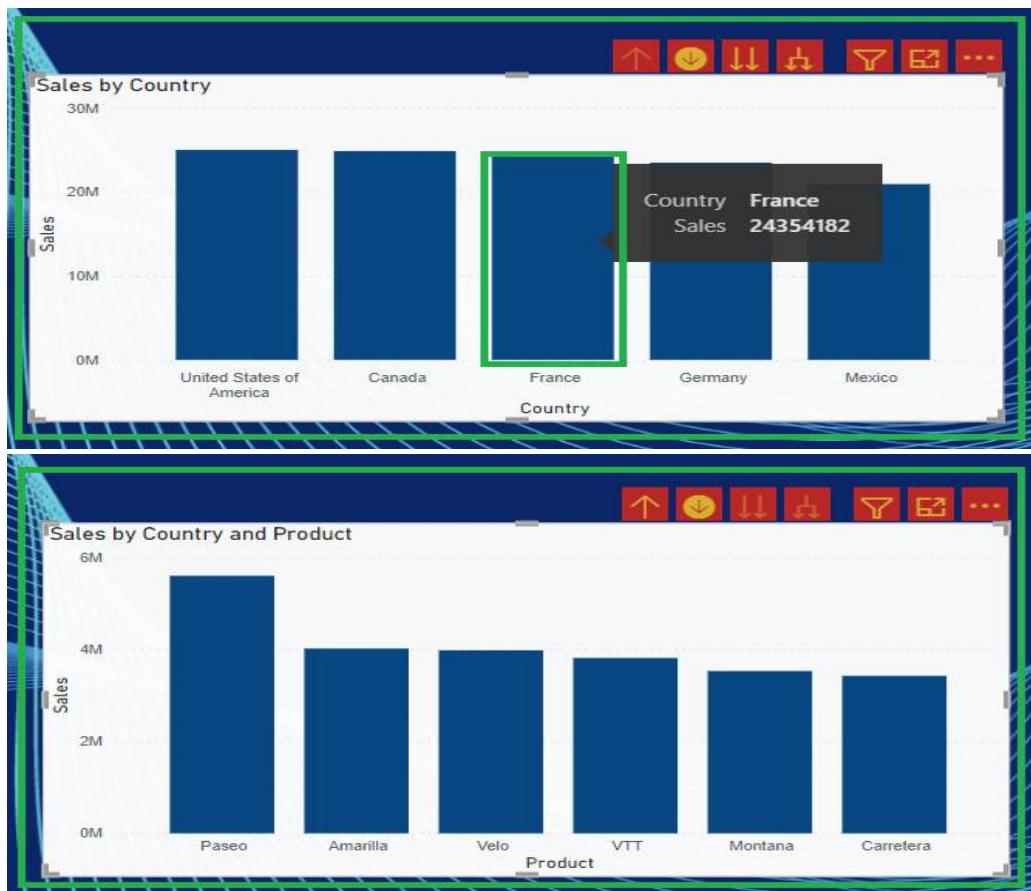
Activate it by clicking on the down arrow symbol. If you click on any value in the selected visual then you can see the next hierarchy of the data. Like this, data will be visible.



Here, I have selected “**Sales by Country**” visual and given “**Country**” and “**Product**” columns in X-axis, sales in Y-axis. Then you can see the down arrow symbol. That symbol you can call as **Drill Down**.

If you click on that symbol, **Drill down** option will be enabled. Then if you click on any value in that visual, next level of hierarchy will be visible.

That means “**Sales by Country and Product**” details for the selected value will be displayed. If you want to go back to the previous visual then you can click on “**Up arrow**” button.



Note: We can only enable the **Drill down** option for the visuals having X-axis and Y-axis options.

Ex) Stacked column chart

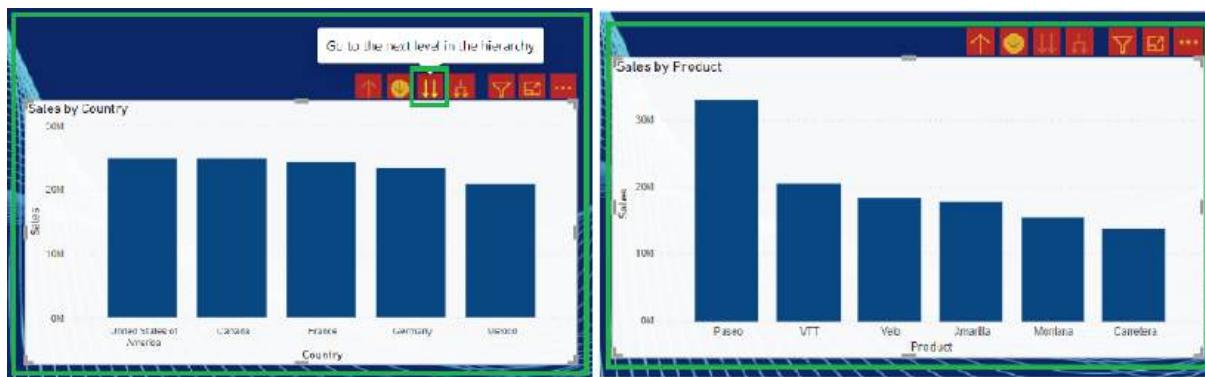
Note: We can put the only calculated columns in the X-axis only. But we can't put measures. We should put Character or date related columns in the X-axis. In Y-axis, we can put measures, numeric values or decimal columns.

Ex)

X-axis→ Country, Product

Y-axis → Sales

Note: If you select single down arrow, selected value corresponding product details will be visible. If you click on double down arrow symbol, then the next level of hierarchy will be visible. Then overall product values will be visible.



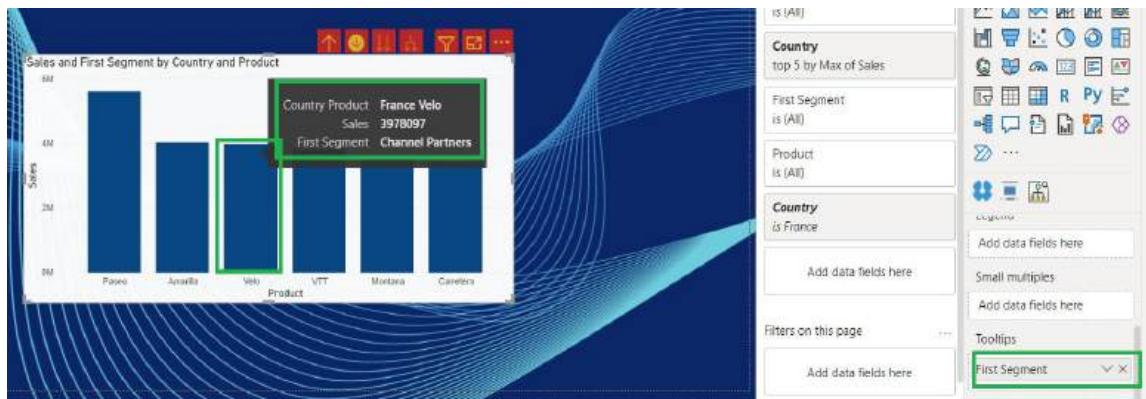
Note: You can see the hierarchy in single level by clicking on the “**Expand all down one level in the hierarchy**” option.



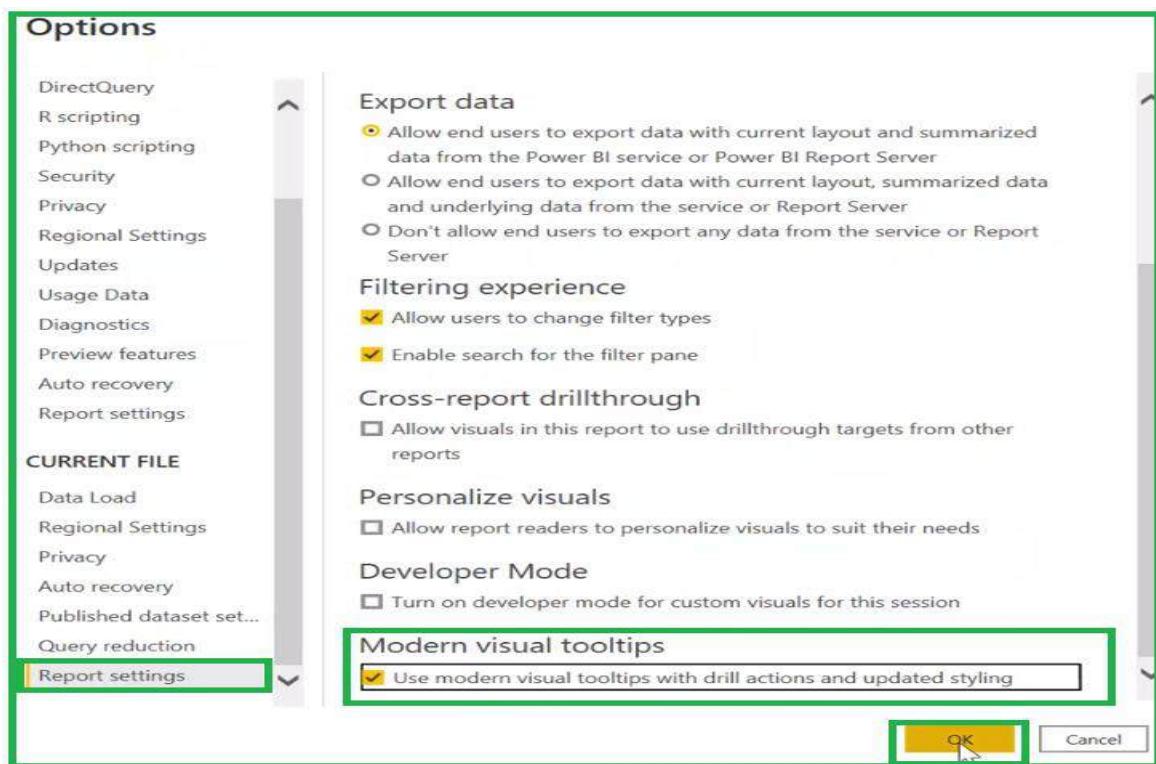
Tool tip: If you mouse over on any object, then automatically short description of that object will be visible on that object. That is called as “**Tooltip**”.



Note: If you want to add any other column values to see as in tooltip, you have to drag that column to the “**Tool tips**” option in the visualization pane.



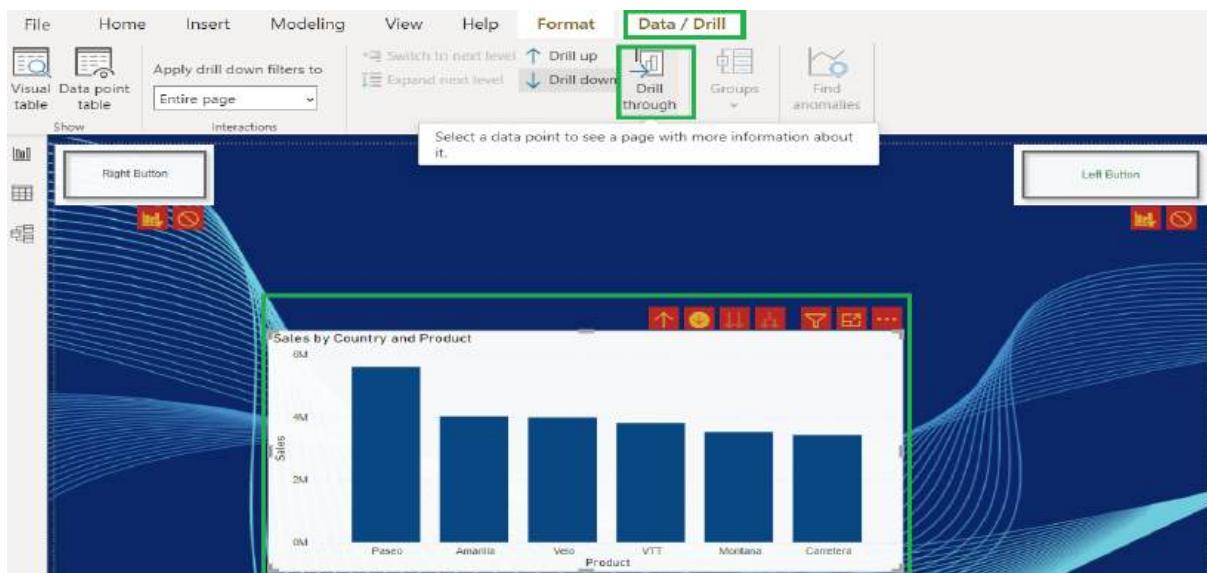
Note: If you want to see the **drill down** options by mouse over the particular bar in the visual, then you have to go to the “File” tab and go to “Options and Settings”. Then go to “Options” and then select “Report Settings” option in “Current File”. Then check on the “Modern Visual Tooltips”.



Result: If you click on “Drill up” option, you can go to the previous level, if you click on “Drill down” option, you can go to the next level of hierarchy.



- **Drill Through:** This is also one kind of filter. That means filter will be happened from one page to another page. Definition of Drill Through is “**Select a data point to see a page with more information about it.**

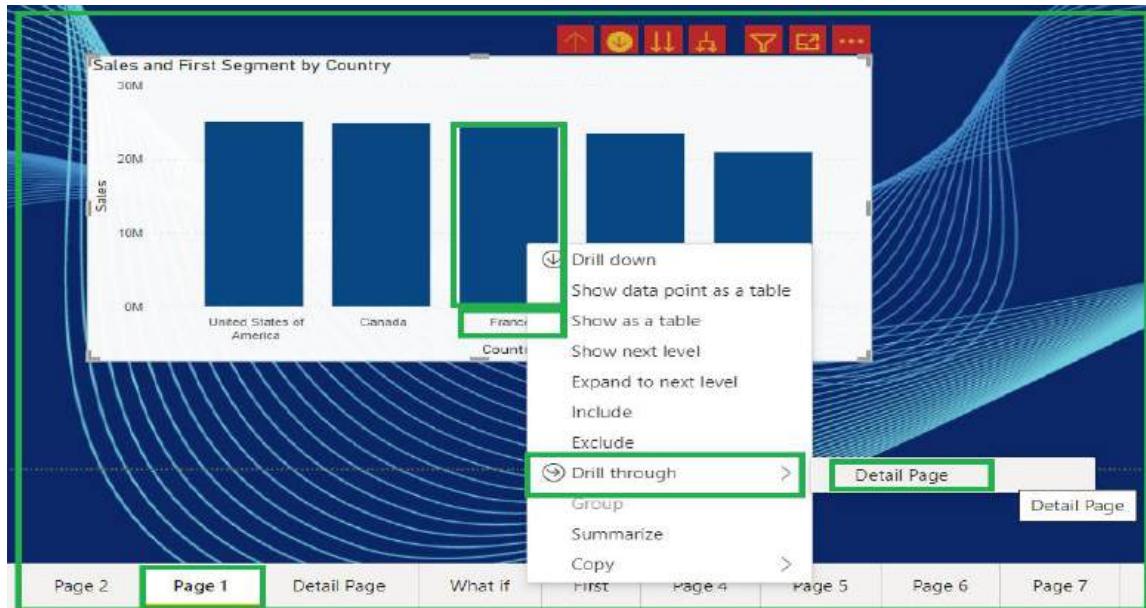


For example, I have selected “**Sales by Country and Product**” visual in one page and if you select any value in that visual then that value detailing will be filtered in another page.

The table data is as follows:

Country	Product	Region	Sales
Canada	Amarilla	Channel Partners	67177
Canada	Amarilla	Enterprise	371647
Canada	Amarilla	Government	257972
Canada	Amarilla	Midmarket	44741
Canada	Amarilla	Small Business	702401
Canada	Carrilera	Channel partners	116619
Canada	Carrilera	Enterprise	306716
Canada	Carrilera	Government	1332157
Canada	Carrilera	Midmarket	120001
Canada	Carrilera	Small Business	654630
Canada	Montana	Channel Partners	45100
Canada	Montana	Enterprise	376313
Canada	Montana	Government	829861
Canada	Montana	Midmarket	57095
Canada	Montana	Small Business	1392191
Canada	Paseo	Channel Partners	165371
Canada	Paseo	Enterprise	905143
Canada	Paseo	Government	3956879
Canada	Paseo	Midmarket	108132
Canada	Paseo	Small Business	2320353
Canada	Velo	Channel Partners	34315
Canada	Velo	Enterprise	70700
Canada	Velo	Government	1374574
Total			119726385

Suppose if I want to filter “**Country**” Column in “**Detail Page**”, then you have to drag and drop that column from fields pane to “**Drill Through**” Section.



I have selected “France” value bar and right click on that value. Then I went through “Drill through” and then click on “Detail Page”. You can see the detailed data about “France” value in the “Detail Page”.

Country	Product	Segment	Sales
France	Amarilla	Channel Partners	43260
France	Amarilla	Enterprise	392952
France	Amarilla	Government	2686042
France	Amarilla	Midmarket	32739
France	Amarilla	Small Business	861435
France	Carretera	Channel Partners	65787
France	Carretera	Enterprise	1187521
France	Carretera	Government	1397092
France	Carretera	Midmarket	39432
France	Carretera	Small Business	733491
France	Montana	Channel Partners	48265
France	Montana	Enterprise	367283
France	Montana	Government	1140658
France	Montana	Midmarket	62486
France	Montana	Small Business	1908693
France	Paseo	Channel Partners	108602
France	Paseo	Enterprise	1250380
France	Paseo	Government	2532866
France	Paseo	Midmarket	241131
France	Paseo	Small Business	1464776
France	Velo	Channel Partners	33887
France	Velo	Enterprise	379544
France	Velo	Government	2332438
Total			24354182

Page 2 | Page 1 | Detail Page | What if | First

Note: You can filter N number of columns in Drill through option.

- **Table Tools Tab:** If you select any data set in the field's pane. Then “Table Tools Tab” will be visible. In this section, you can use “Manage Relationships” or “New Measure”, “New Column” and “New Table” options as per your requirement.

The screenshot shows the Power BI desktop interface with the 'Table tools' tab selected in the ribbon. On the left, there's a visual titled 'Sales and First Segment by Country' showing sales values for five countries. The Fields pane on the right lists various columns, with 'Aggregation' highlighted.

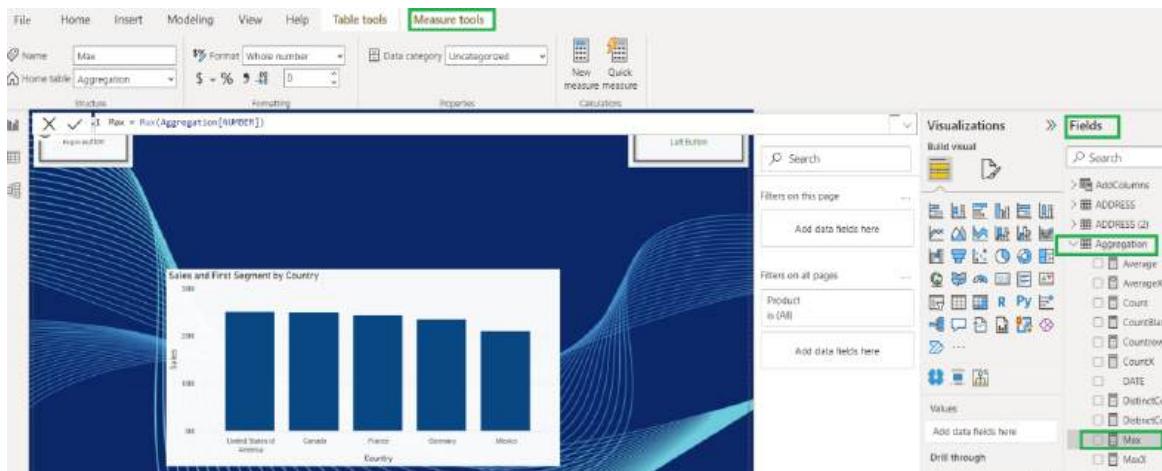
- **Column Tools Tab:** If you select any column in a data set in the **field's pane**. Then “**Column Tools Tab**” will be visible. In this section, you can use “**Manage Relationships**” and “**New Column**” options as per your requirement.

You can change the data type and format of the column in this section. You can change the numeric value into decimal also.

The screenshot shows the Power BI desktop interface with the 'Column tools' tab selected in the ribbon. The Fields pane on the right shows the 'Aggregation' column selected, with its properties visible in the ribbon tabs: Data type (Text), Format (\$ - %), Summarization (Don't summarize), and Sort by column.

Note: Columns we are viewing in the data tab are called as “Calculated Columns”.

- **Measure Tools Tab:** If you select any measure in a data set in the **field's pane**. Then “**Measure Tools Tab**” will be visible. In this section, you can use “**New Measure**” option as per your requirement.



Note: You can't change the data type for measure values. But you can change the format of the measure in this section. You can change the numeric value into decimal also.

VISUALIZATIONS

Visualization is a process of taking raw data and transforming it into graphical or pictorial representations such as charts, graphs, diagrams, pictures etc. So users can quickly analyze the data prepare reports to make business decisions effectively.

Visualization is more important because it is a quick and easy way to convey concepts to the end-users.

We are inherently in the visual world where pictures or images speak more than words. So it is easy to visualize a large amount of data using graphs and charts than depending on reports or spreadsheets.

In Visualizations pane, we have two tabs. One is “**Build Visuals**” and “**Format your report page**”. If you want to take visuals then we can use “**Build Visuals**” tab and you can add data fields into the visuals here. If you want to change the format of that visual you can use “**Format**” tab.

We have another tab also called as “**Analytics**” tab. This will describe the bench mark line or margin or max value for the report.

In **Build Visuals** tab, we can see **Default visuals**. If you want to use **Custom visuals**, then you can fetch visuals from the **market store**. Most of the times we will use **Default visuals** only because custom visuals will impact the performance.

Note: Custom visuals will occupy more space than the default visuals.

We have lot of default visuals in the visualizations pane.

Common Chart Types

- Column Chart
- Line Chart
- Stacked Chart
- Combination Chart
- Pie Chart / Bar Chart
- Tree Map
- Map

Power Bi Specific Charts / Visuals

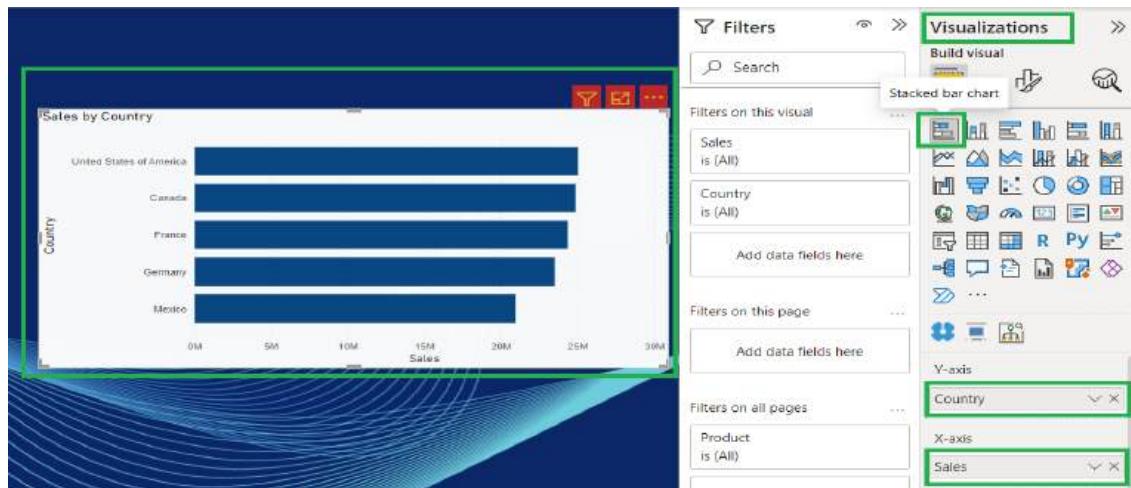
- Card
- Multi Row Card
- Filters
- Q/A
- Table / Matrix

1. **Stacked Bar Chart:** Stacked bar charts are useful to compare multiple dimensions against a single measure. In this chart, we have Y-Axis and X-Axis. Here, Horizontal lines are called as X-axis, and Vertical lines are called as Y-axis.

Suppose we are taking **Financials2** data set, and putting Character type of columns in Y-axis and numeric type of values in X-axis.

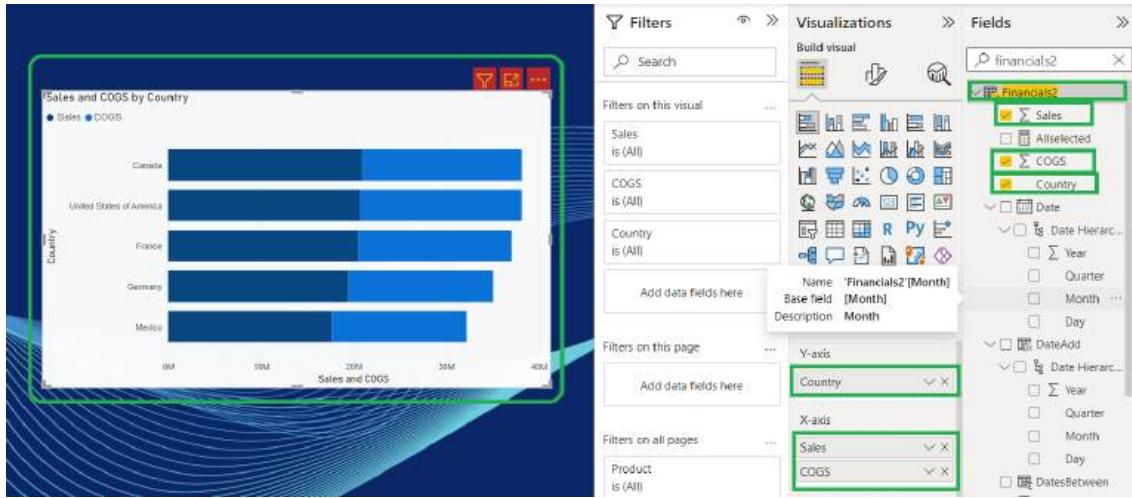
Go to Build Visuals and Double click on “**Stacked Bar Chart**” symbol from the default visuals.

For example, I am taking “**Country**” column in Y-axis and “**Sales**” column in X-axis.



Whatever the column we have mentioned in Y-axis, those column values will be visible in Y-axis. Whatever the column we have mentioned in X-axis can be visible as horizontal bars.

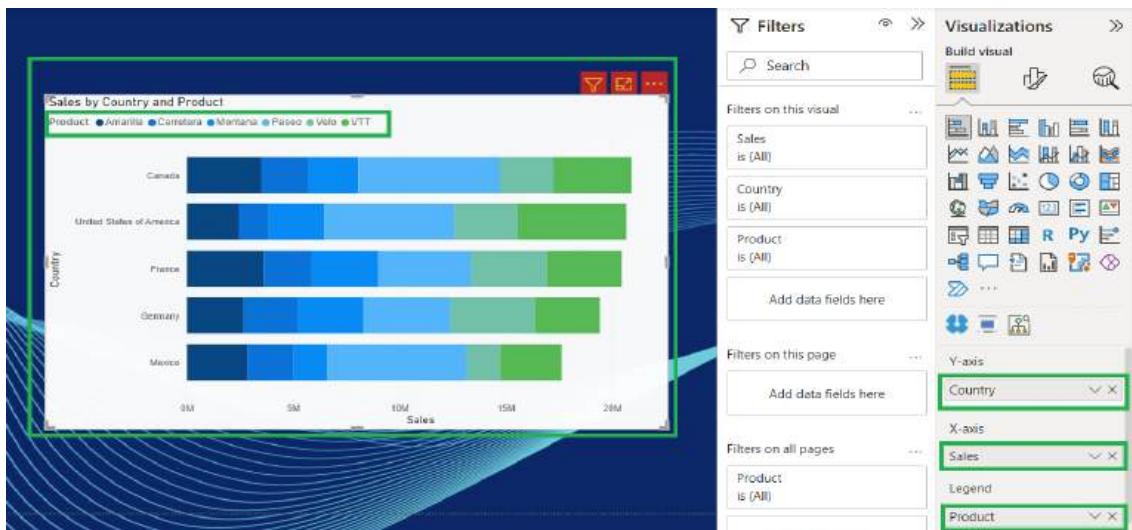
If you put multiple columns in X-axis then Drill Down option will be activated. If you put multiple columns in Y-axis then Values will be visible as stacked to one by one. See below screen shot.



Legend: A legend is used to identify data in visualizations by its color, size, or other distinguishing features. If you put any field in legend, that corresponding values will be shown in different colors for single bar.

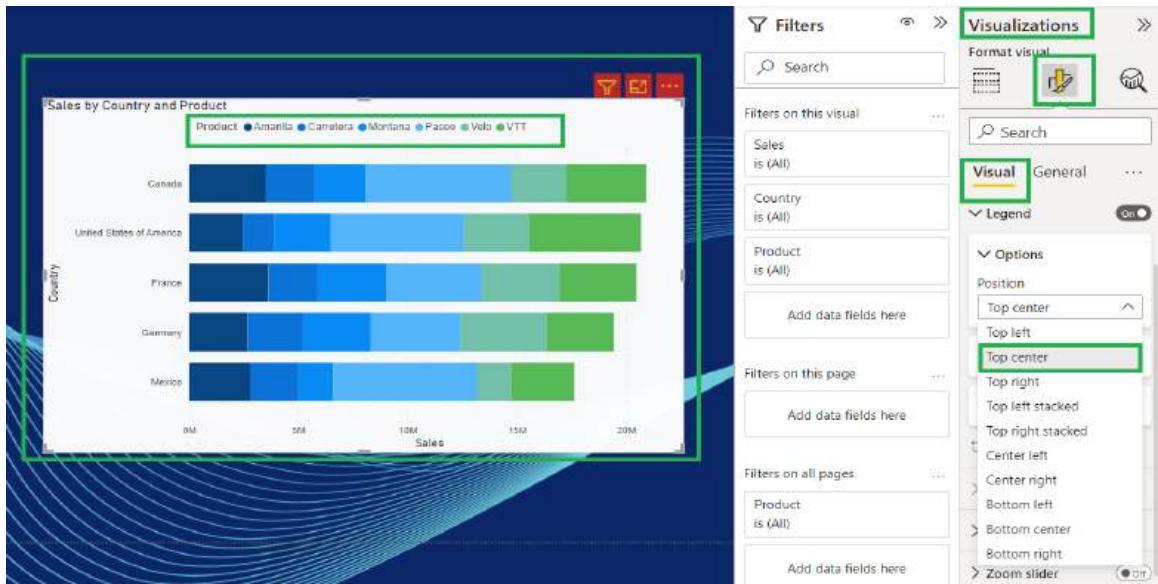
Note: Only one column should put in Legend to represent data in different colors.

I am taking “**Product**” column in **legend** that means sales column will be divided into multiple segments according to the product.



By default legend values will be aligned to Top Left, if you want to display those values in different positions, then you can change it in “**Format**” option.

Go to “**Format**” then click on drop down menu of “**Legend**”. Then you can select the alignment.



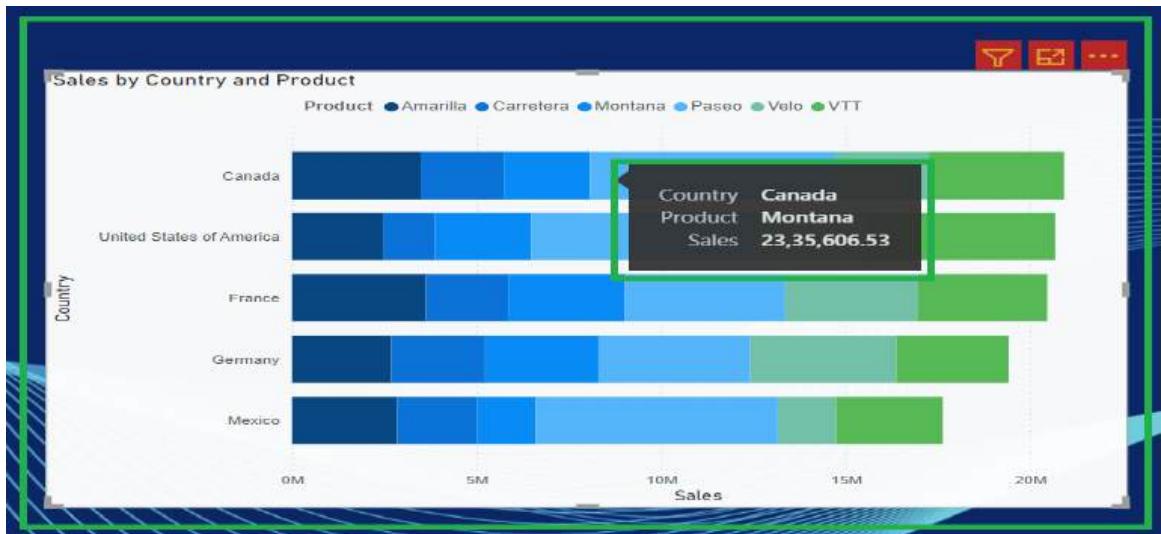
You can change the font, size of the text, Bold or Italic, Underline and Color of the text for the legend by simply click on drop down menu of **text** option in **Legend** drop down of **Format** section in **Visualization** pane.



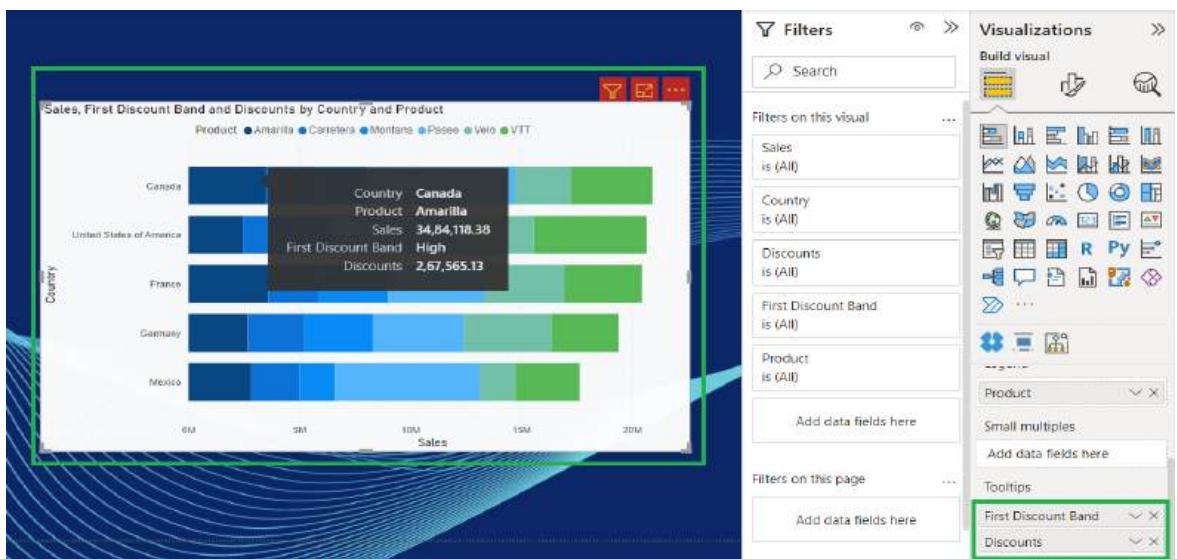
You can change the legend name in visualization pane and you can off the legend also.

Tooltips: A tooltip is a brief, informative message that appears when a user mouse-hover on a particular element. Tooltips can be visible when you put the columns in either X-axis or Y-axis.

All columns information will not be visible by doing like this. For some columns; tooltip values automatically displayed when you mouse-hover on any object.



If you want to see more columns information, then you have to put required column in tooltips option in visualization pane.



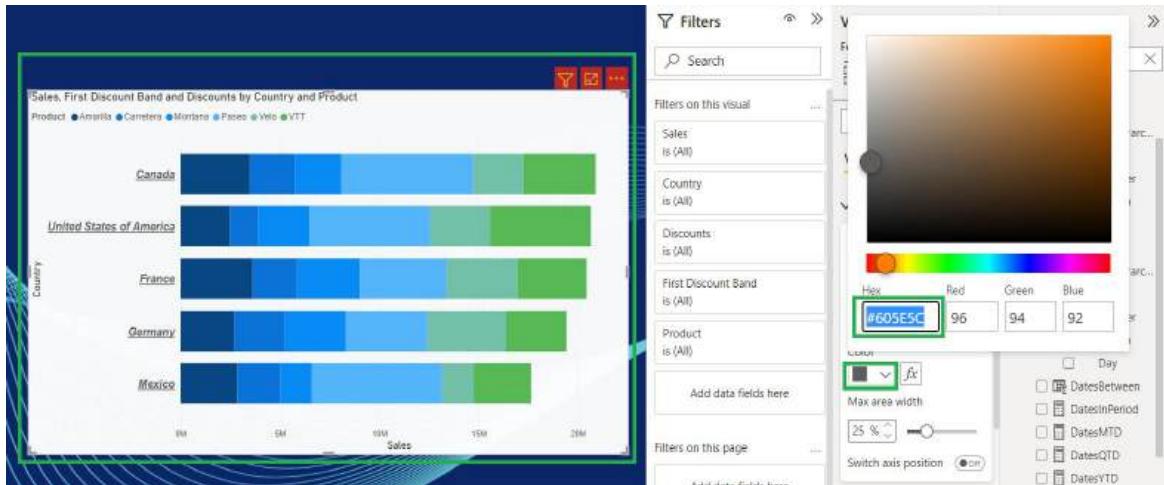
Whatever the columns you have put in tooltips option, then visualization doesn't have effect in structure but if you put the columns in either X-axis, Y-axis, Legend, and then visualization will effect in terms of display.

Changing design part of the visual: If you want to change the design part of the visual then you have to select the visual and go to the “**Format this Visual**” option in “**Visualizations**” pane. Then change the required details. Here you can see two options one is “**Visual**” and another one is “**General**”.

If you want to change size or font or color of the X-axis related values then click on X-axis drop down menu and change. If you want to change Y-axis related things then go to Y-axis drop down menu.



If you want to put the user-defined color for the Y-axis or X-axis values, then go to “**More colors**” option in **Color** section. Then you have to provide the code given by the user. These codes will be stored as hexadecimal format.



If you want to provide color dynamically for the values then you use functions by just click on “**fx**” option in **color** section.

You can OFF or ON the title name of the X-axis or Y-axis by dragging scroll bar corresponding to the axis.



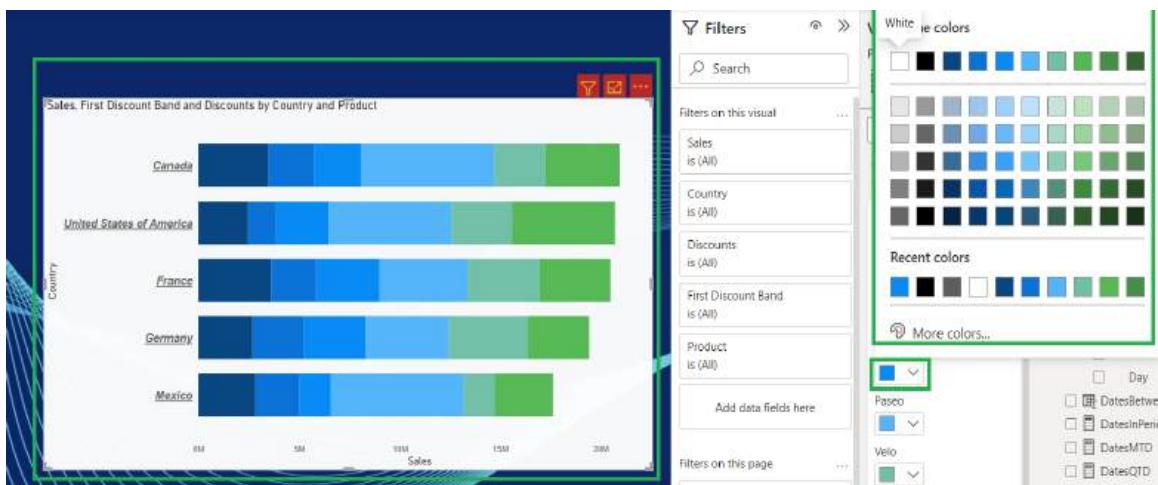
Gridlines: if you want see the dotted lines in background of the visual, and then you can select **ON** option in visualization pane. You can change the color of the gridlines and width of the lines also as well as style also.



Bars: For each and every bar in the visual has different color and that is set by the Power BI by default. If you want to change the colors of any bar, then you can do in this option.

User defined colors also we can give by using color codes decided by the user and pate those into the “**More colors**” option.

You can give the space between the bars also by simply drag the scroll bar in “**inner padding**” option and you can give the “**Minimum category width**” also.

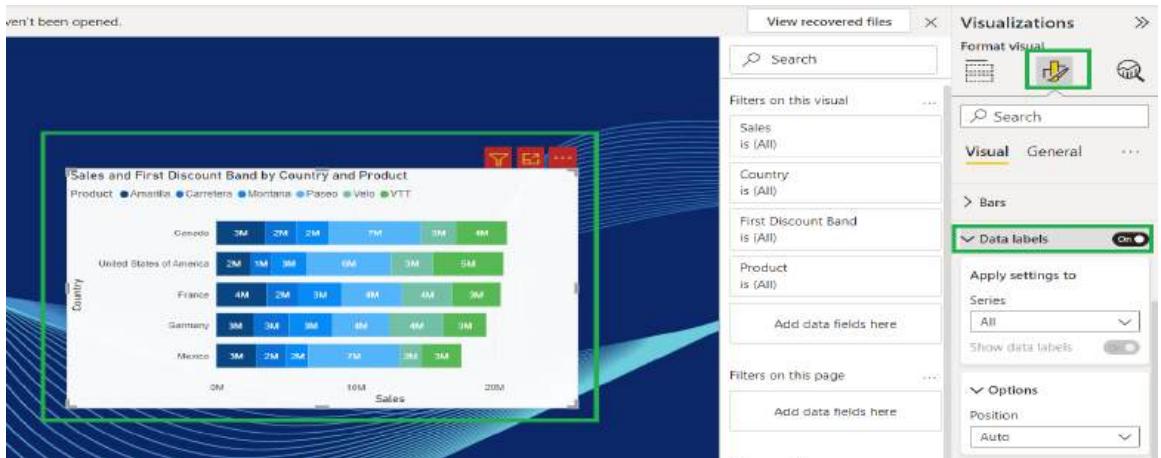




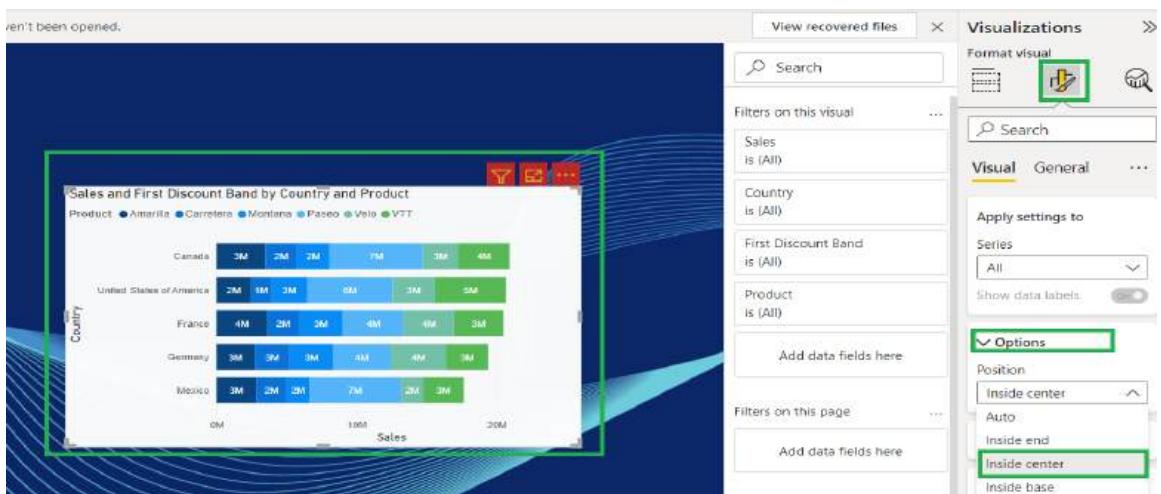
Data Labels: If you want to see all the values (Numbers) within the charts then we can use data labels.

By default, data labels will be in **OFF** mode only. Whenever required then we can **ON** them.

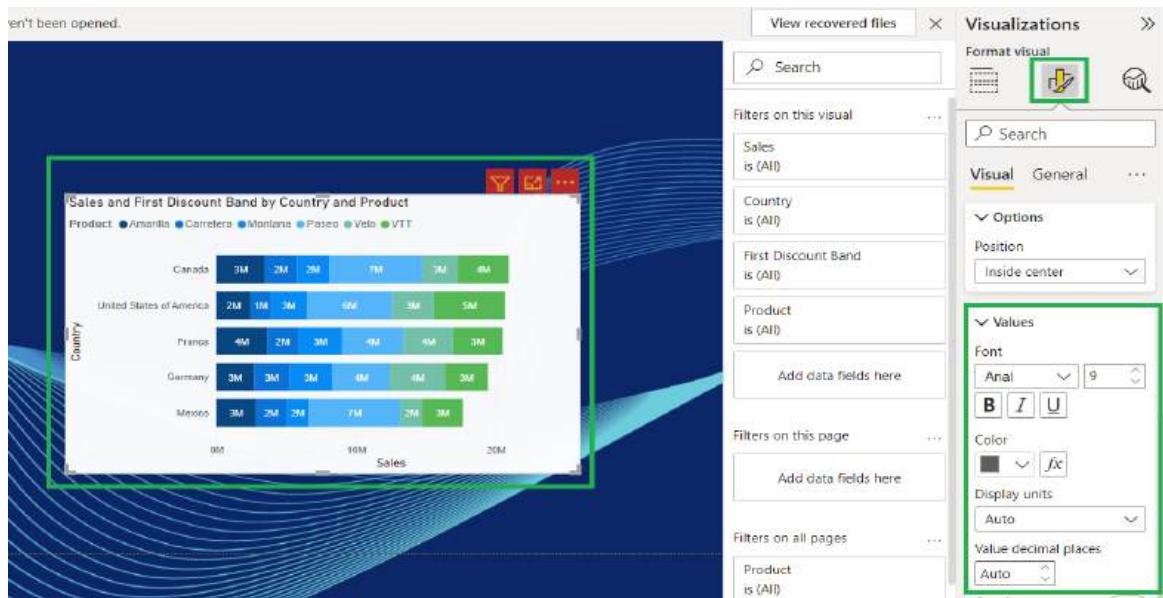
For example, I want to display sales values on the bars, then we should enable data labels option.



You can change the position of the data labels also within the bars.



We can change the font, color, size or display units by using “**Values**” option in the data labels section from “**Visualization**” pane.



Total Labels: If you want to display sum of the label values along with labels then we can use this option.

There you can change the size of the total values, color and font etc.

Display units you can change to thousands, millions, trillions, Auto etc.



General: If you want to change the properties of the visual, title, effects, tooltips, Alt text then we have go for “**General**” tab.

Properties: If you want to change the properties of the visual, then we can use this option. Suppose if we want place one visual on another visual place, and then we can take properties of one visual and copy them to another visual.



Title: If you want to change the name of the title for the visual, then you can use this option.

You can change font, color, size, text color, background color, alignment as you like.



Effects: If you want to make some effects for the visual for better visibility, then we can use this option. That means borders for the visual and size of the border we can change and shadow can apply.

By applying shadow, look and feel is good and representation will be visible as good.



Toolips: Whenever you mouse-hover on any object then that corresponding object short description will be shown as in a text box. That is called tooltip.

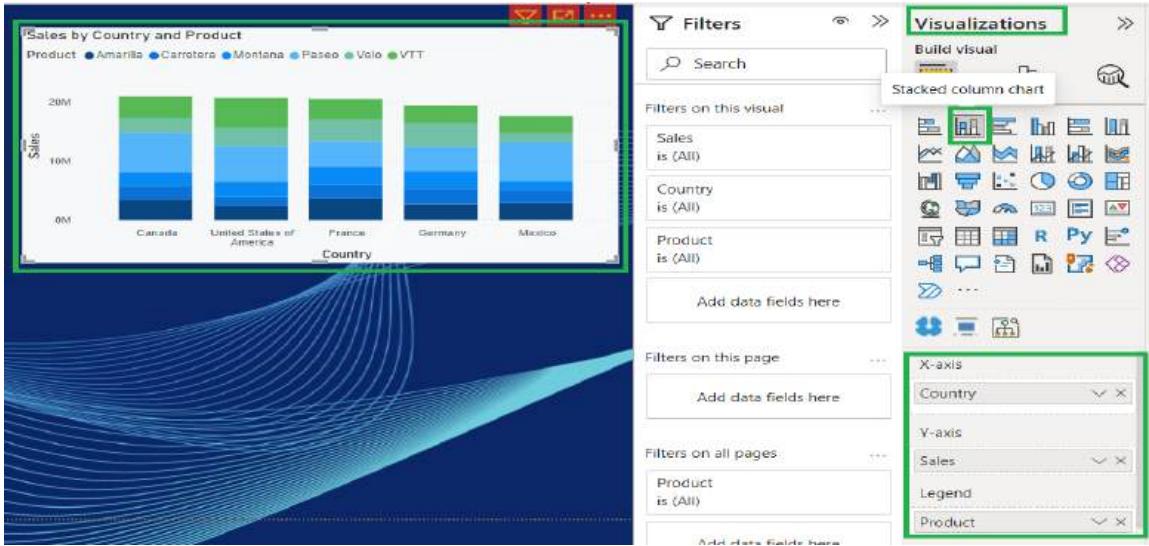
Alt Text: Alternative text we can provide for the visual to identify the visual belongs to.



2. **Stacked Column Chart:** A stacked column chart is a **form of bar chart that shows the composition and comparison of a few variables, either relative or absolute, over time**. Also called a stacked bar or column chart, they look like a series of columns or bars that are stacked on top of each other.

This is also having X-axis (Horizontal) and Y-axis (Vertical). Always need to put the calculated columns in X-axis and measures in Y-axis. It looks like opposite to the stacked bar chart.

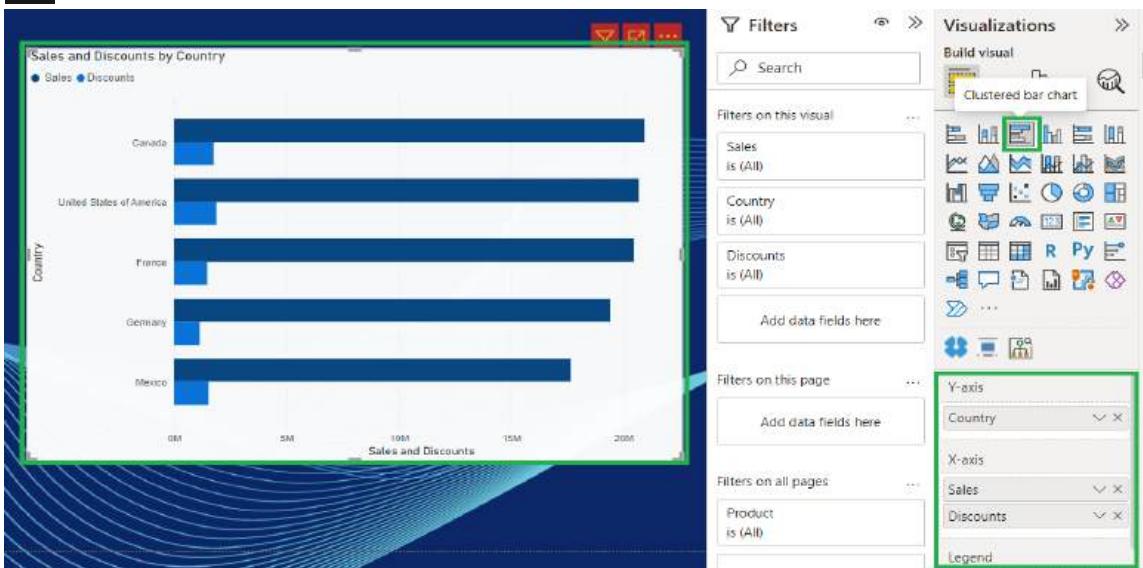
Note: Stacked column chart most of the times we will use for showing data over date and we can use this for representing time data, having same intervals.



Note: Whatever the options we have applied in stacked bar chart same options will be available in this stacked column chart as well. Same options we can use here.

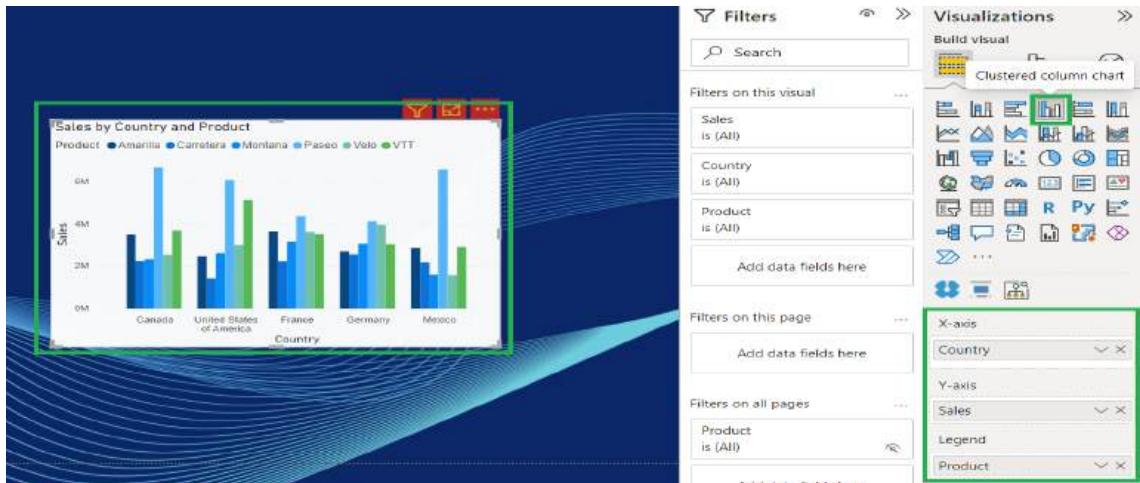
3. **Clustered Bar Chart:** Clustered Bar Chart (also known as Grouped bar chart, Multi-series bar chart) is great for displaying and comparing multiple sets of data over the same categories (like sales revenue of various departments of the company over several years).

Ex)



Note: Format options are same as above.

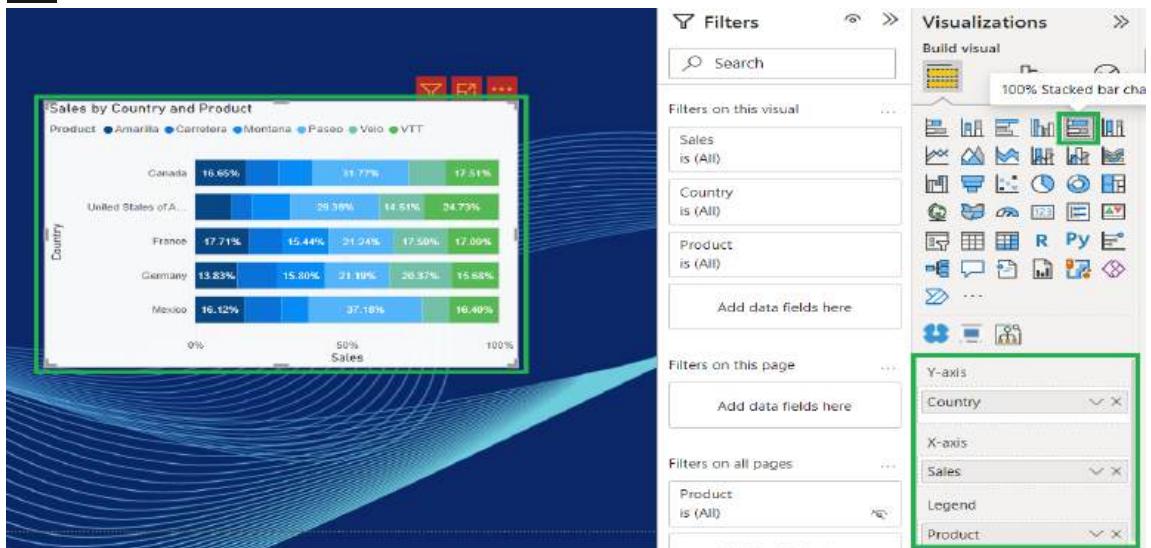
4. Clustered Column Chart: On Power BI, the Clustered column chart is useful to display the comparison of multiple series as in the vertical axis. We can describe as a Clustered Column Chart is used to represent the vertical bars of multiple regions against a single Metric.



Note: Format options are same as above.

5. 100% Stacked Bar Chart: Power BI 100% stacked bar chart is used to display relative percentage of multiple data series in stacked bars, where the total (cumulative) of each stacked bar always equals 100%. In a 100% stacked bar chart, Axis is represented on Y-axis and Value on X-axis.

Ex)

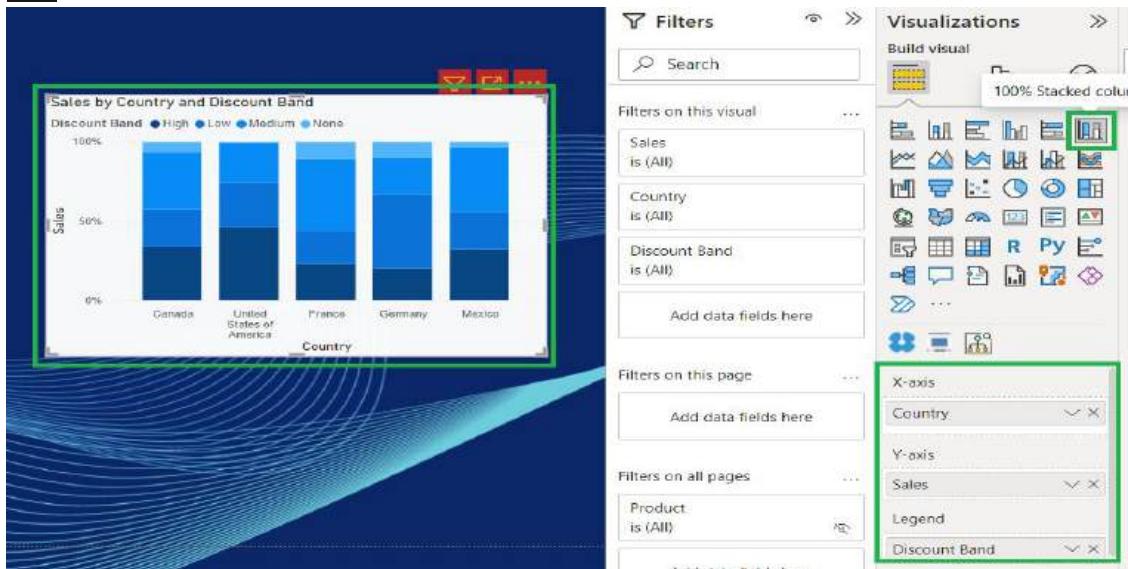


Note: Format options are same as above.

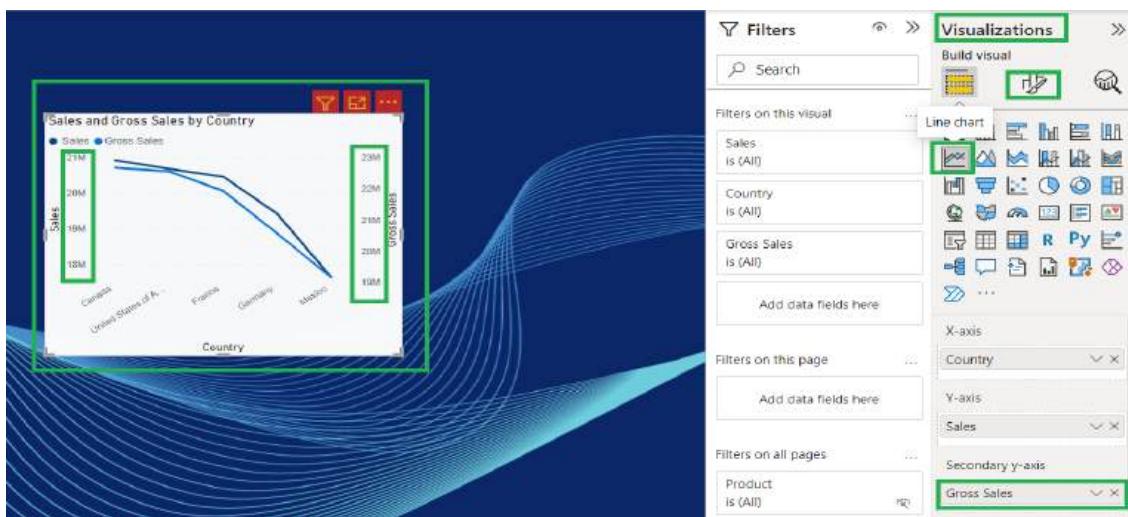
6. 100% Stacked Column Chart: Power BI 100% stacked column chart is used to display relative percentage of multiple data series in Stacked columns, where the total (cumulative) of each Stacked columns always equals

100%. In a 100% Stacked column chart, Axis is represented on X-axis and Value on Y-axis.

Ex)



7. **Line Chart:** A line chart is a type of chart used **to show information those changes over time**. Line charts are created by plotting a series of several points and connecting them with a straight line. Line charts are used to track changes over short and long periods.



Along with X-axis and Y-axis, additionally "**Secondary Y-axis**" option is there in this chart. That means you can see values of the field in Y-axis in left side of the visual and right side of the visual.

I have put **Sales** column in **Y-axis** and **Gross Sales** column in **Secondary Y-axis**. See the above screen shot.

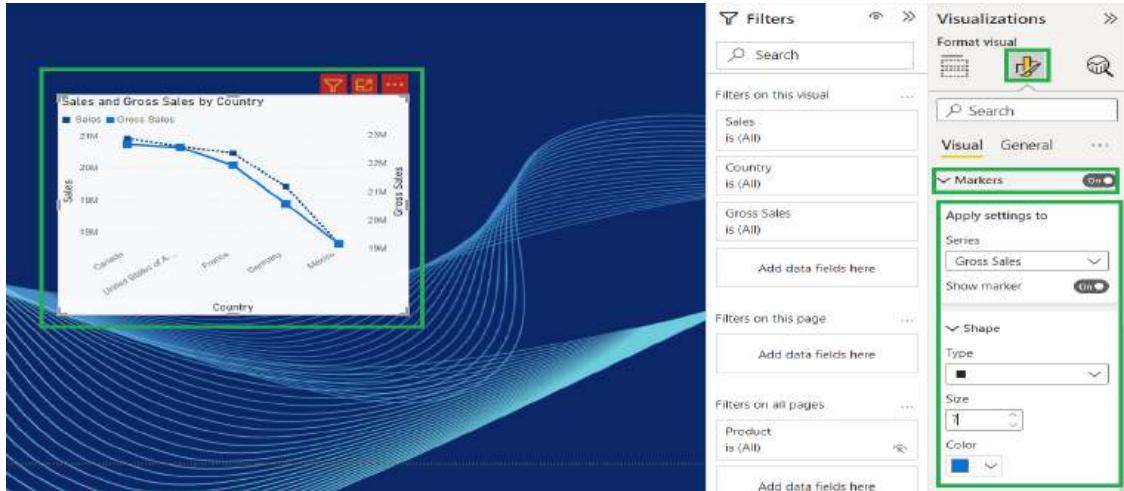
Charts will be shown as lines in between the values with several spot points.

If you want to change the shape of the lines, spots of the join type, color etc. Suppose I want to change the line of the sales column to dotted line. Then I have changed the style of that line as dotted by selecting sales column in format this visual option line section.



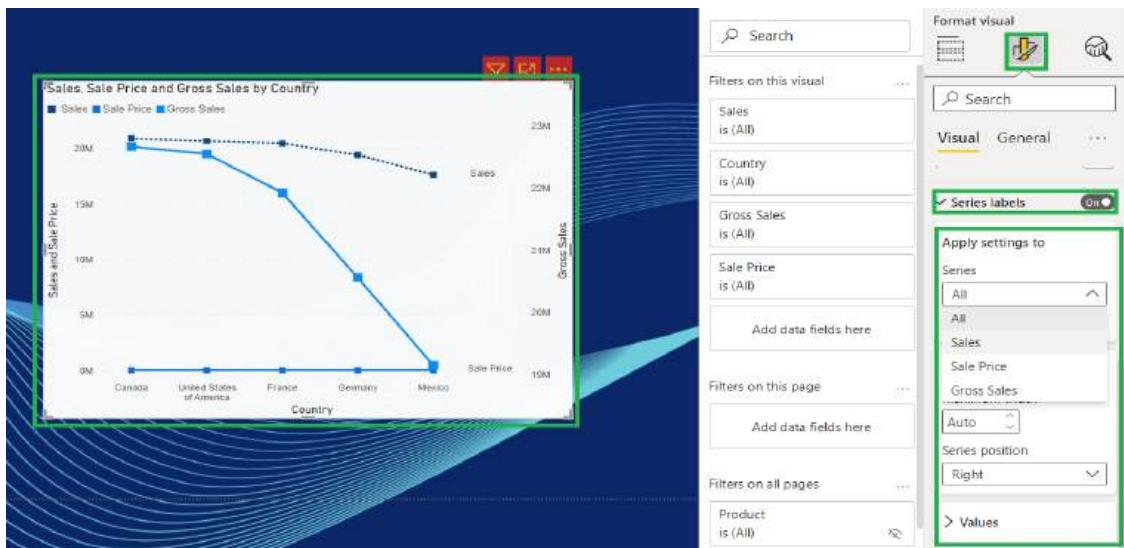
You can increase the stroke width. That means line thickness will be increased.

Markers: It shows data points with dark highlighted dots. By default, these spots will be shown as round symbols. You can change the data points type to square etc.



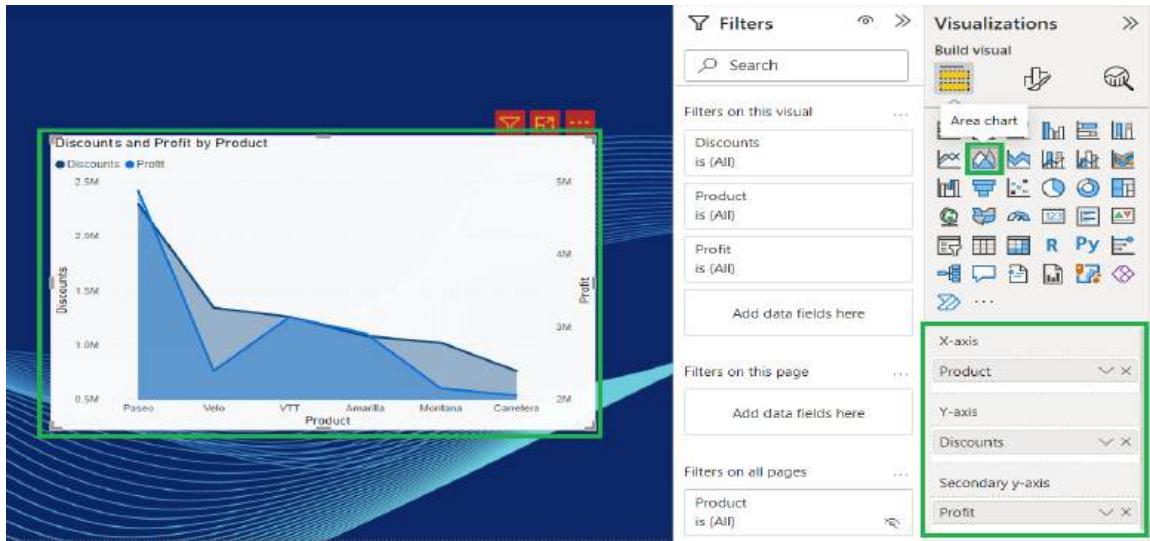
Note: By default, markers are disabled mode, if you want them, make it enable.

Series Labels: Series labels describe which line belongs to which field. You can **ON** this if required.



Note: All the rest of the options are same as above.

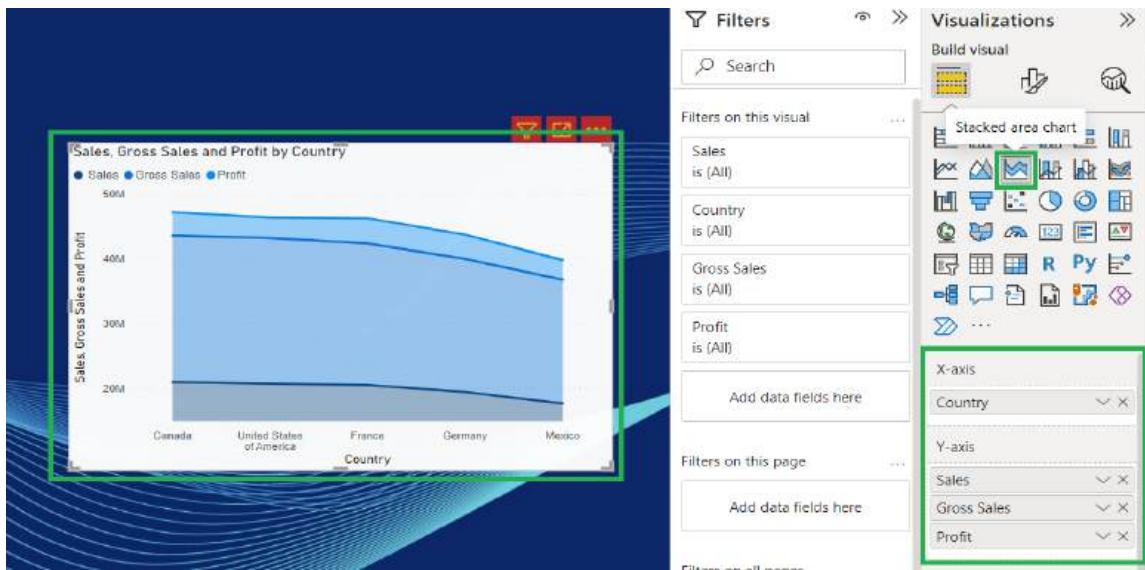
8. **Area Chart:** An area chart is a graph that combines a line chart and a bar chart to show changes in quantities over time. It's similar to a line graph in that data points are plotted and connected by line segments. However, the area below the line is colored in or shaded.



Note: An area chart is similar to a line chart, except the area under the line is shaded in.

Note: Whatever the options we have in line chart are applicable here as well.

9. **Stacked Area Chart:** Stacked Area Chart is plotted in the form of several area series stacked on top of one another. The height of each series is determined by the value in each data point.



Note: All options for formatting this visual are same as with area chart.

10. Line and Stacked Column Chart: The Line and Stacked Column Chart is a combo charts that combines the Line chart and Column chart together in one visual. By combining these two visuals together, you can make a very quick comparison between two sets of measures.

The main benefit of this type of chart is that you can have one or two Y axes. What this means is that you can either display two measures that would have the same Y axis, something like Total Sales and Profit.

Or, we could show two measures that are based on completely different values such as Units Sold and Profit.



11. Waterfall Chart: Waterfall charts are useful to show the running total as values where they are added or subtracted.



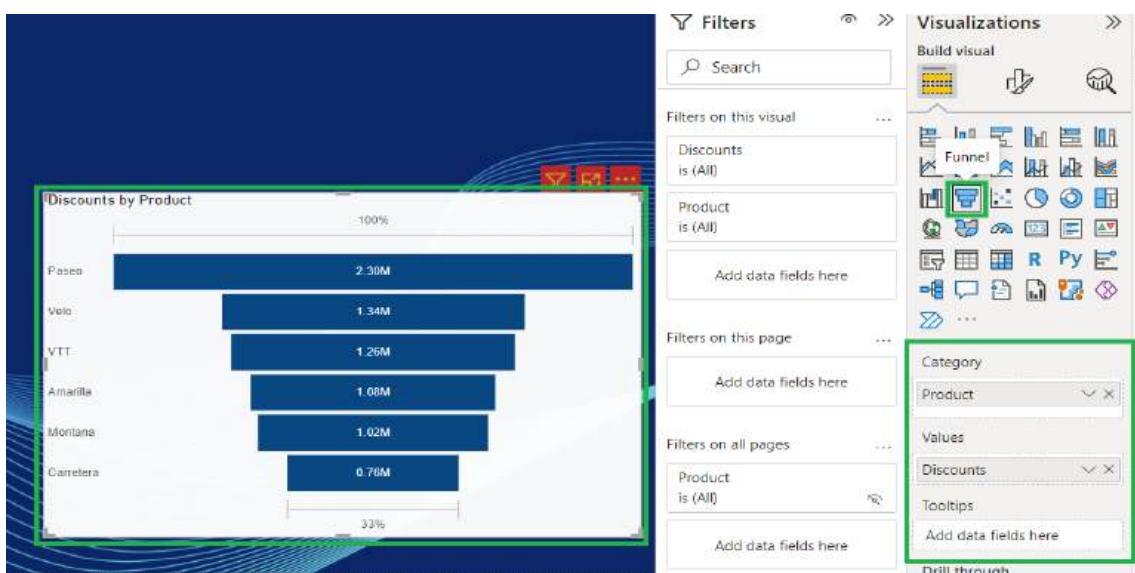
In Power BI, we also have a waterfall chart where the columns in the waterfall chart are color-coded for the view of increase or decrease in the values.

Breakdown: If you put any calculated column in this option, then total values will be split into multiple chunks according to the column you have mentioned here.

Instead of X-axis, we have only one option here is "**Category**". Based on this field data will be shown as in water fall looks like.

12. Funnel: A funnel chart shape like a funnel having the first stage being the largest and each subsequent stage is smaller than its predecessor.

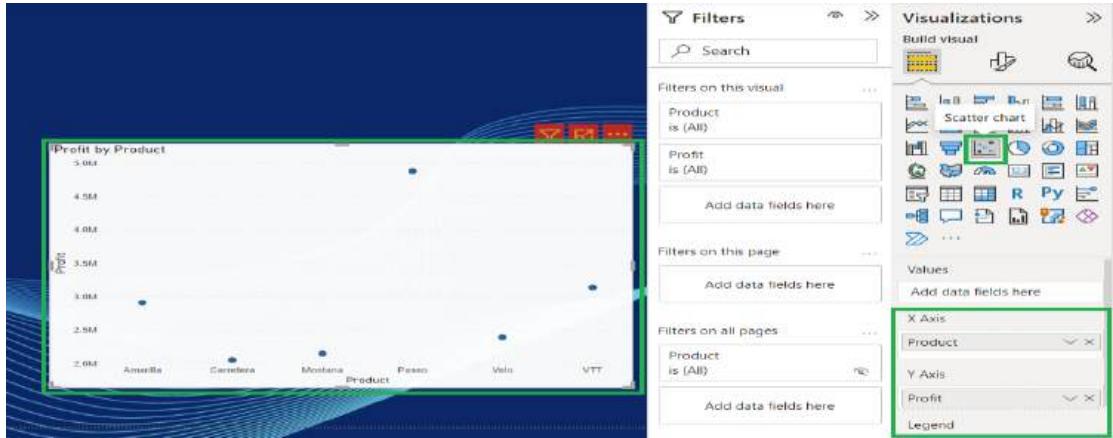
Here data will be shown as bars in descending order and it display the value in percentage also.



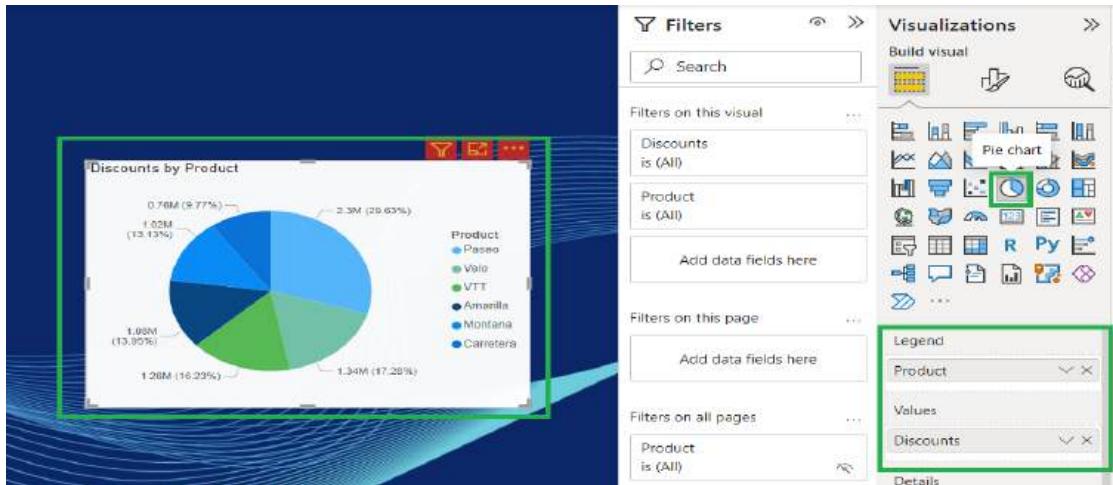
Note: By default, data labels option will be enabled for this chart.

13. Scatter Chart: A scatter chart shows the relationship between two numerical values. It shows the data with dotted spots.

In this chart, you can change the spots shape, color, size in “**Markers**” option in “**Format**” pane.



14. Pie Chart: The pie chart is a round-shaped circle chart where each category data set is shown in a pie shape based on the value of each data label.

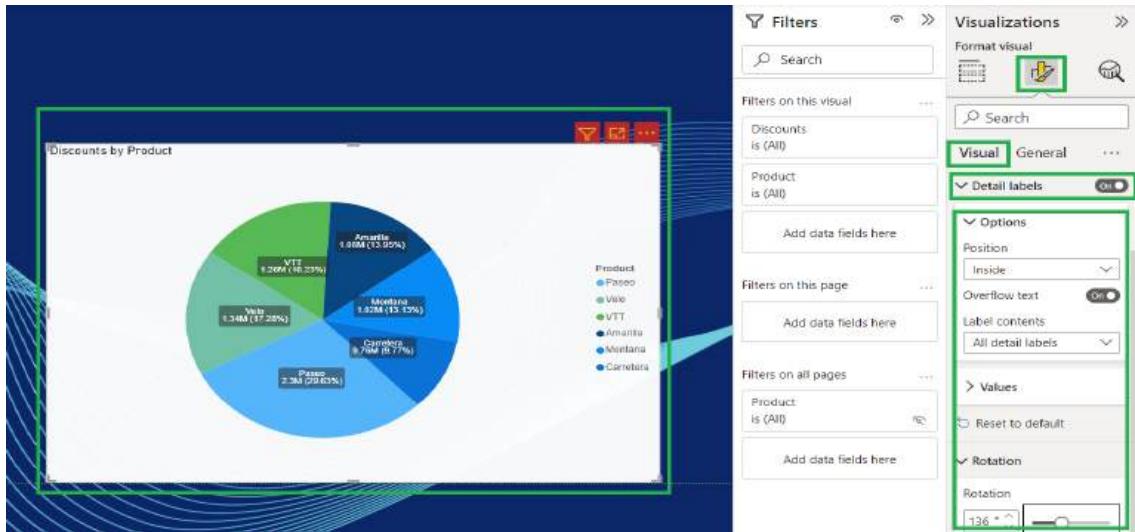


Here each slicer is displaying different color for the different values. If you want to change the color of the each slicer then you can go to format and select slicer option.

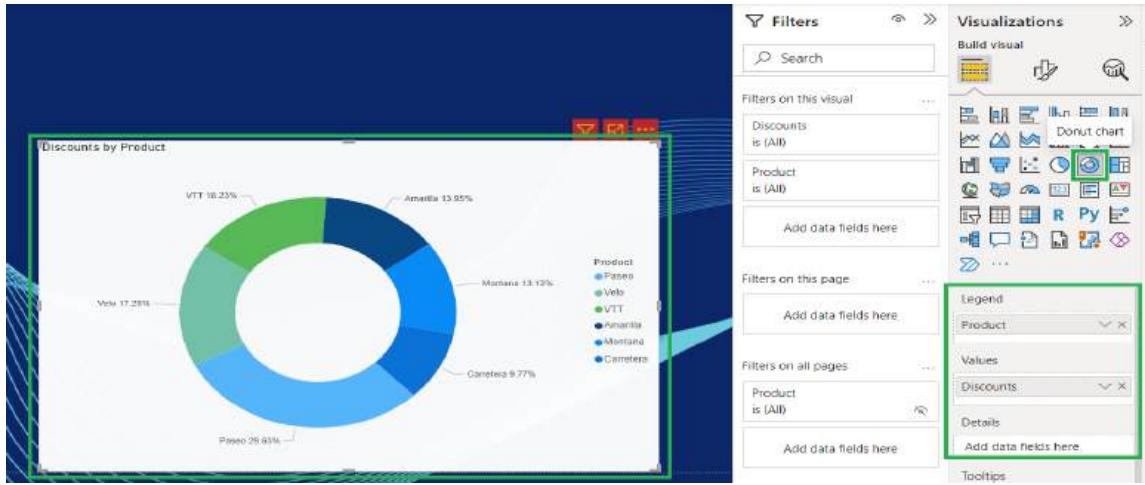
If you want to change format of this visual, then you can go to format this visual option, then choose the option as you like.

Detail Labels: By default, values will be displayed as value and its percentage. You can make changes to display values differently by using this option.

That means values you can place inside or outside and you can display details of the slicer. You can rotate the visual also. See below screenshot.

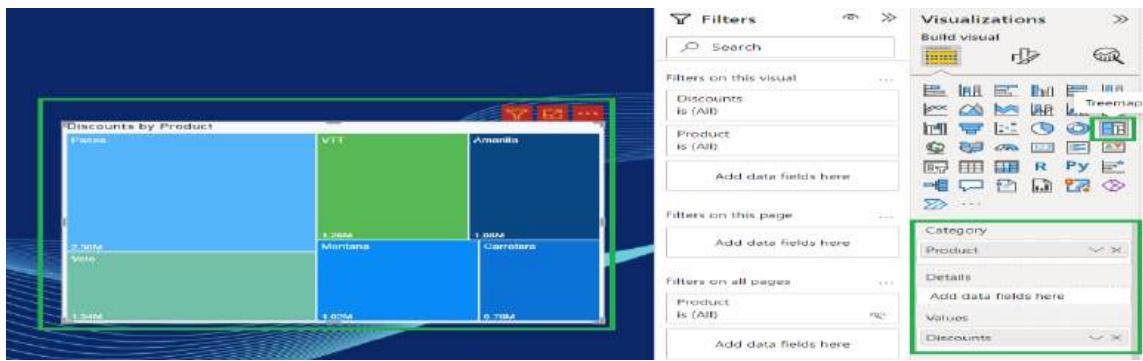


15. Donut Chart: A doughnut chart is similar to a pie chart in that it **shows the relationship of parts to a whole**. The only difference is that the center is blank and allows space for a label or icon.



16. Tree Map: Tree map display hierarchical data as a set of nested rectangles. Based on the value, then rectangle size will be depended.

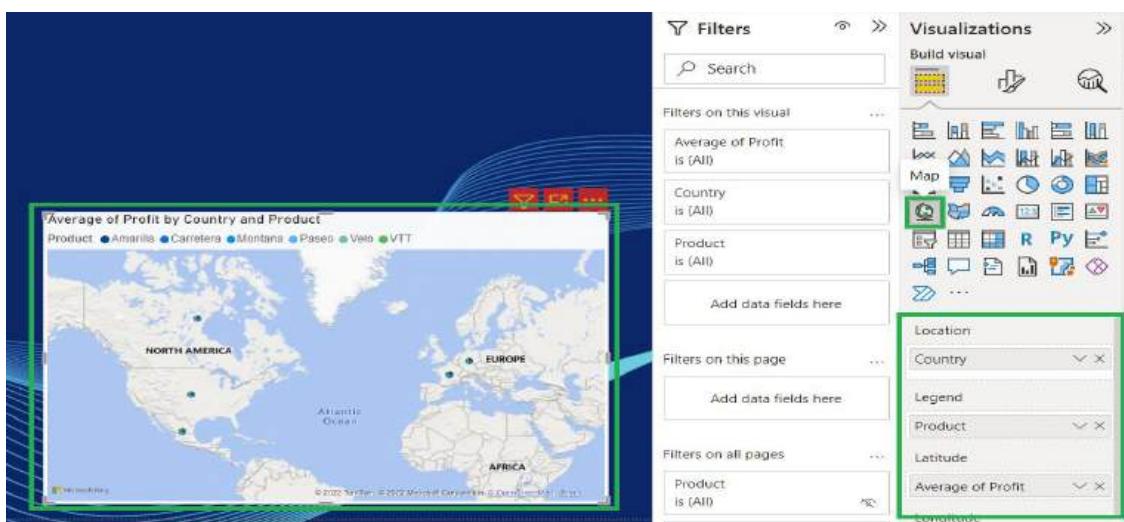
Highest value shows in big rectangle, lowest value shows in small rectangle.



17. Map: Map visualization is used to analyze and display the geographically related data and present it in the form of maps.

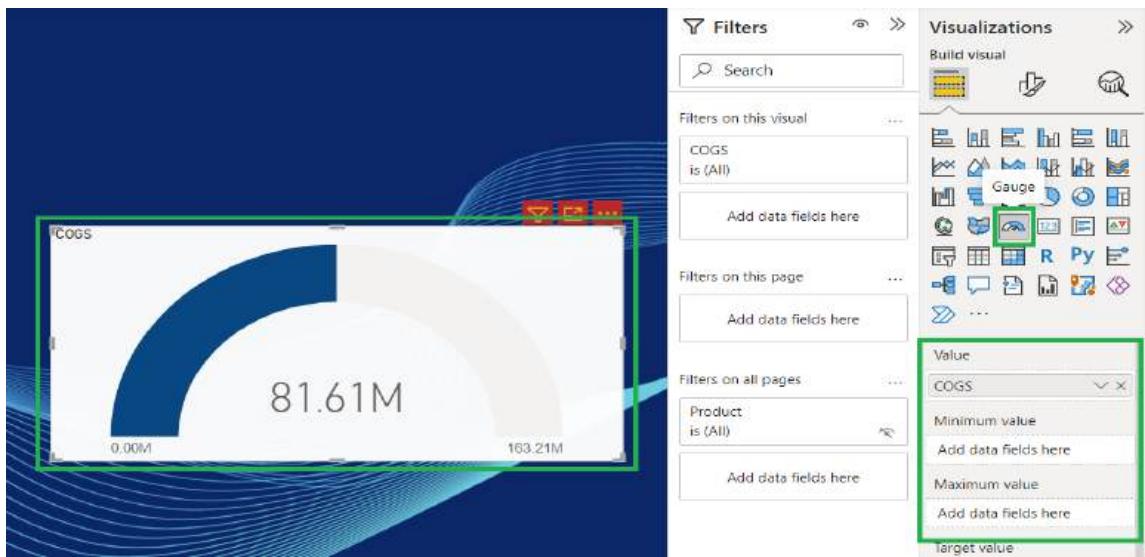
This kind of data expression is clearer and more natural. We can visually see the distribution or proportion of data in each region.

In this, we have to provide latitude and longitude to show the google map look like.



18. Gauge: Gauge charts are used to show progress towards a particular goal. It helps understand to what extent a goal has been completed.

It will show minimum value and maximum value.

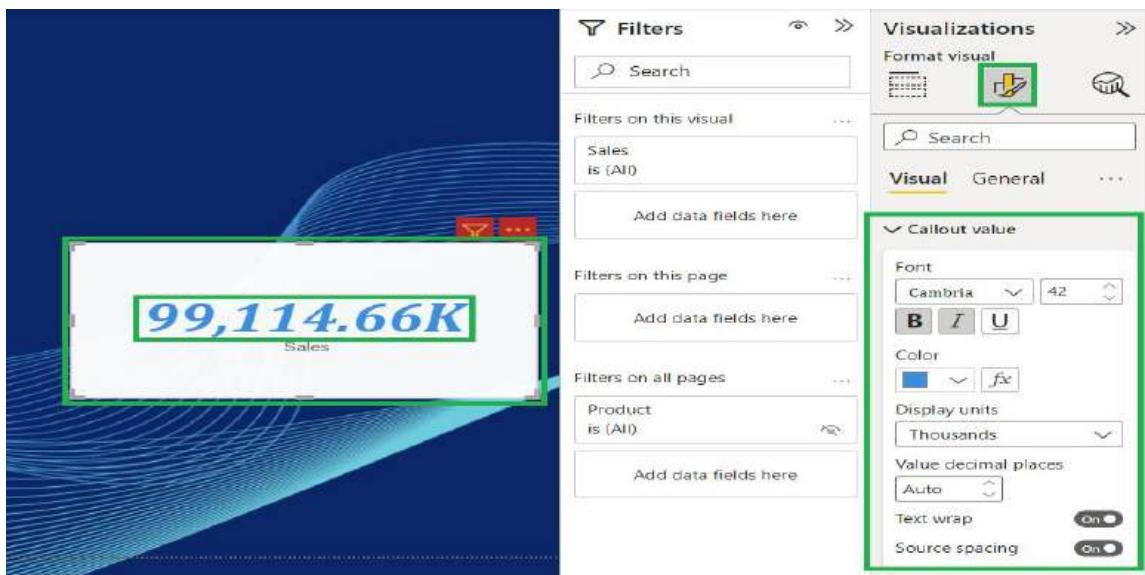


19. Card: Measure values will be mentioned in this visual. It will display numeric single value such as total sales, market share year over year, or total opportunities, is sometimes the most important thing you want to track.

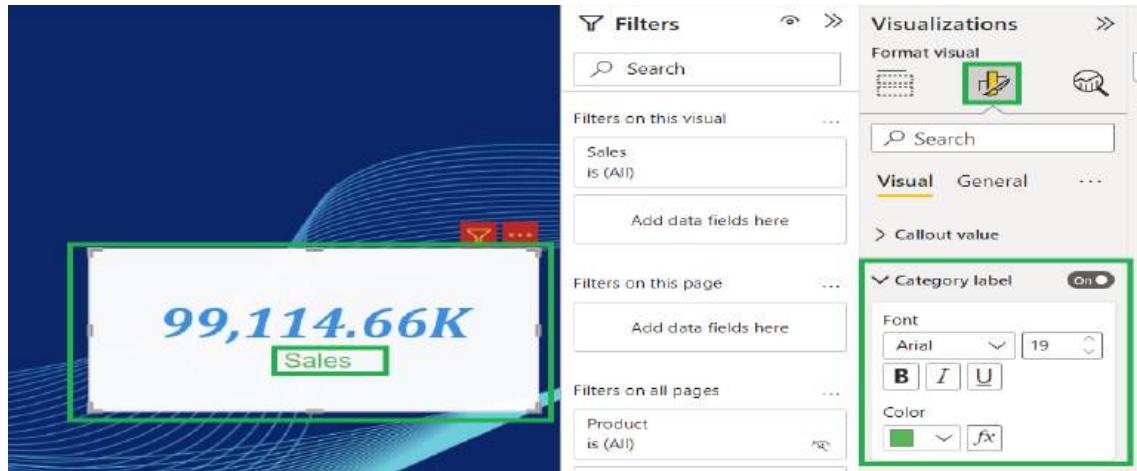
In this visual, we have only one “**fields**” option to mention the measure value. Suppose I am mentioning “**Sales**” column to the “**fields**” option.

In format option, we have two options those are “**Callout value**” and **Category label**.

Callout Value: In this, we can change the color of the value, size, font, style, display units and we can represent the value with decimals.



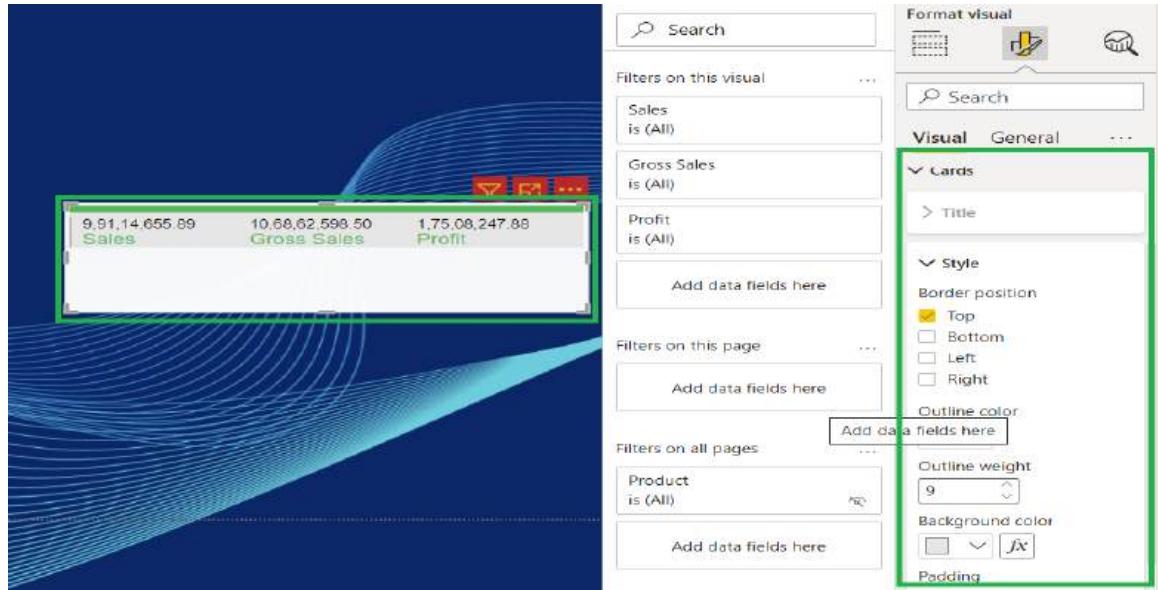
Category label: This will show the measure name that we have mentioned in the “**fields**” option. If you want, you can turn off, you can change the color of this string and size, font etc.



Note: In card visual, only one measure can be shown. We can't put multiple measure values in this visual. For that we have to use **Multi-row-card**.

20. Multi-row Card: This is used to represent multiple measure values. That means we can put multiple measure values in the “**fields**” option.

Whatever the options we have in “**Card**” visual in format section will be applicable here as well. Additionally we have another option called “**Cards**”.



In this **cards** option, we can change the boarder position in style, outline color, weight, back ground color etc.

21. KPI: KPI means, Key performance indicator which is used to communicate the amount of progress toward a measurable goal.

The screenshot shows a Power BI report with a KPI visual. The KPI visual displays the value '20.68M' with a goal of '120 (+17232587.64%)'. The 'Visualizations' pane on the right lists various chart types. A callout box highlights the 'Format visual' pane, which contains settings for 'Value', 'Trend axis', 'Country', and 'Trend'.

Optionally, format the KPI by selecting the paint brush icon to open the **Format visual** pane.

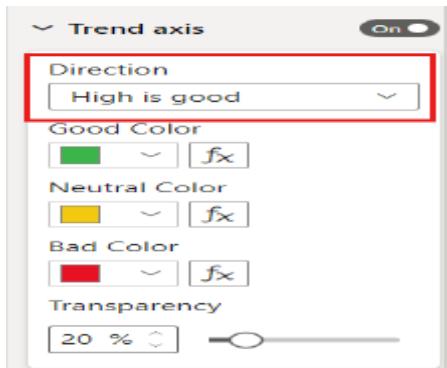
The screenshot shows the 'Format visual' pane in Power BI. It includes sections for 'Callout value', 'Icons', 'Trend axis', 'Target label', and 'Date', each with an 'On' toggle switch.

Callout value - controls the indicator's display units, decimal places, and text formatting.

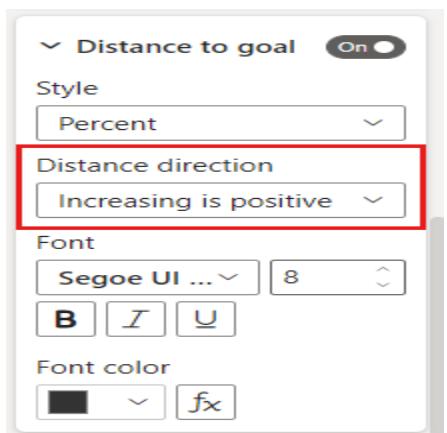
Icons - when set to **On**, the visual shows small icons next to the value, a green checkmark for an increasing value, and a red exclamation point for a decreasing value. The value's direction is set by **Trend axis**.

Trend axis - when set to **On**, the visual shows the trend axis as the background of the KPI visual. People consider some KPIs better for *higher* values and consider some better for *lower* values.

For example, earnings versus wait time. Typically a higher value of earnings is better versus a higher value of wait time. For this report, select **Direction > High is good**. Optionally, change the color settings.



Target label - when set to **On**, the visual shows the value's label. **Distance to goal** sets the style and direction of the distance from the goal.

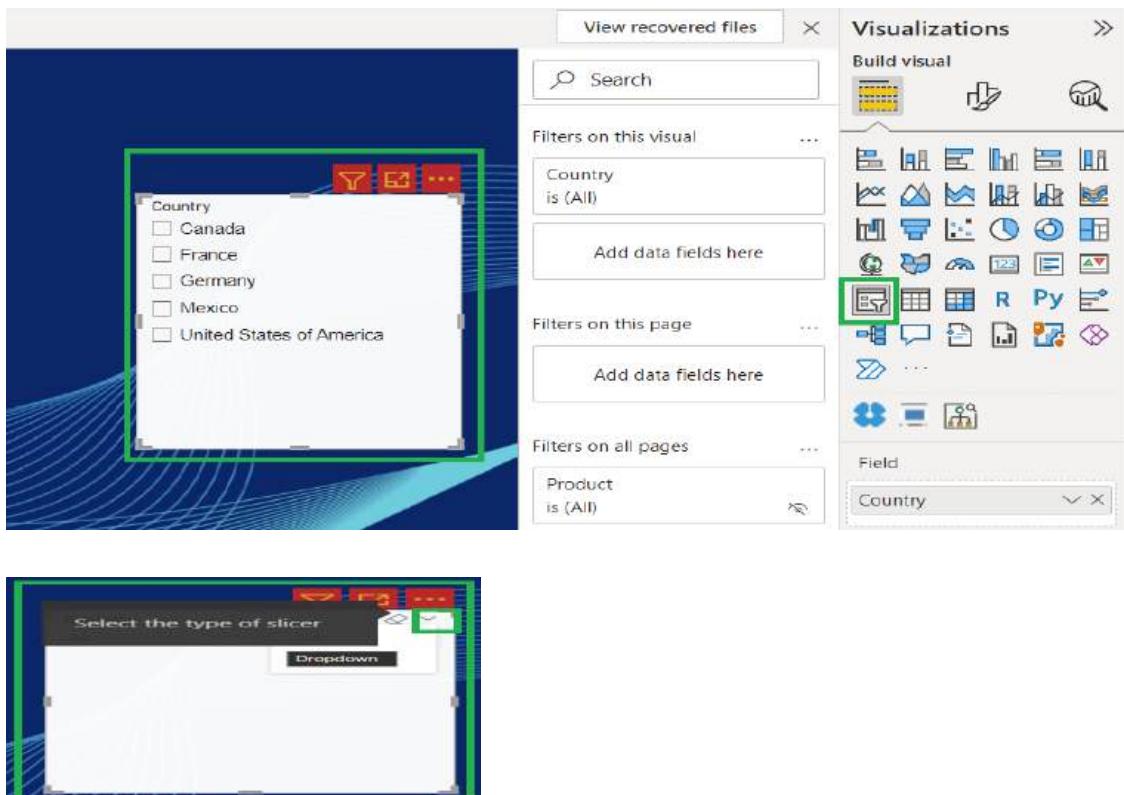


Date - when set to **On**, the visual shows the date. Optionally, change the font, and its style and color.

22. Slicer: Slicers in Power BI are a type of on canvas visual filters. The slicers enable a user to sort and filter a packed report and view only the information they want. Unlike filters, the slicers are present as a visual on the report and let a user select values as they are analyzing the report.

It is also one type filter. We can filter the value in the slicer corresponding value will be shown in other related visuals in the same canvas.

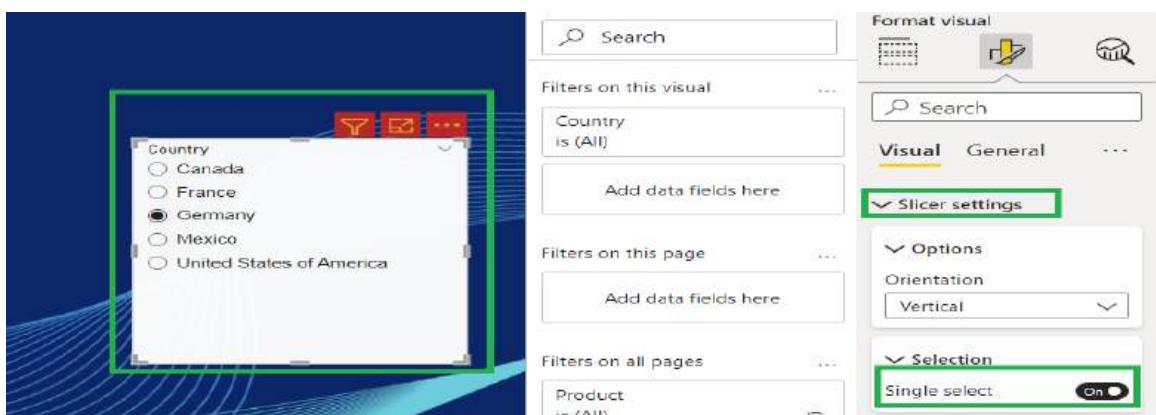
We can put multiple columns in slicer as well.



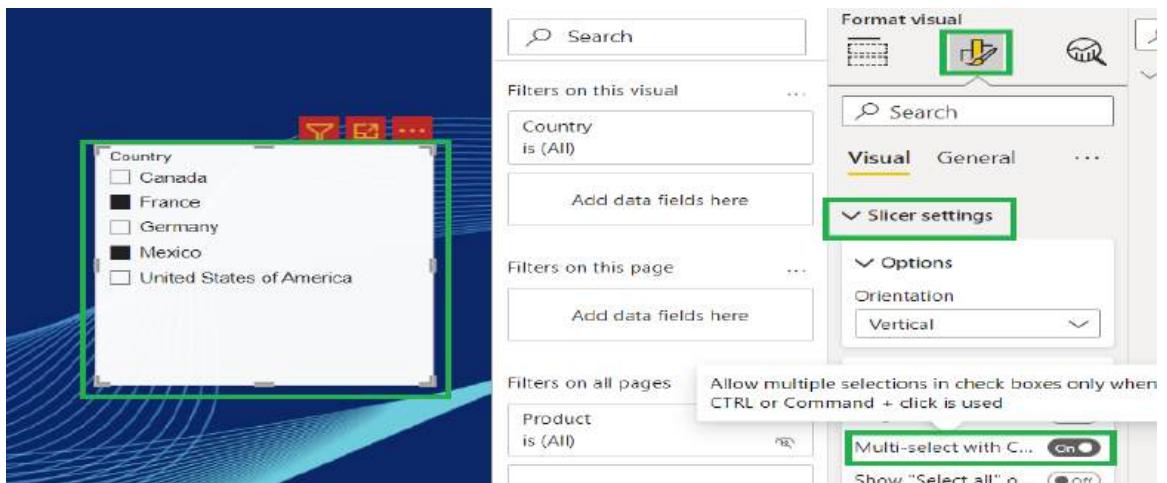
If you want to change the values as drop down menu, then you have to click on drop down arrow of the top right corner of the slicer.

By default, slicer set as single selection, if you want to make it multi select, then go to format option and then go to slicer settings.

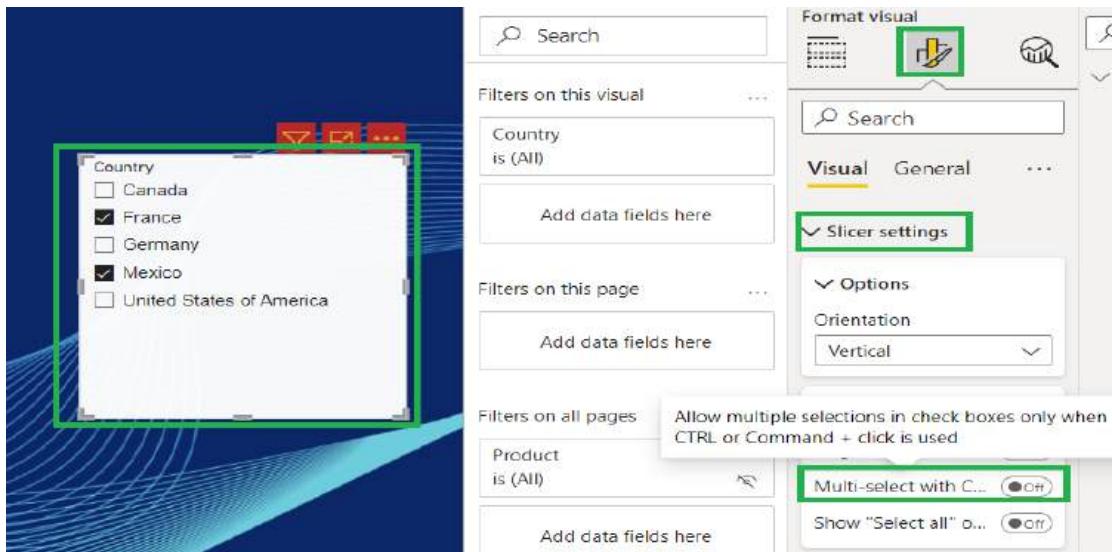
If you select single button ON then slicer will be viewed as radio button form. If you put it OFF then multi-select with control you can select.



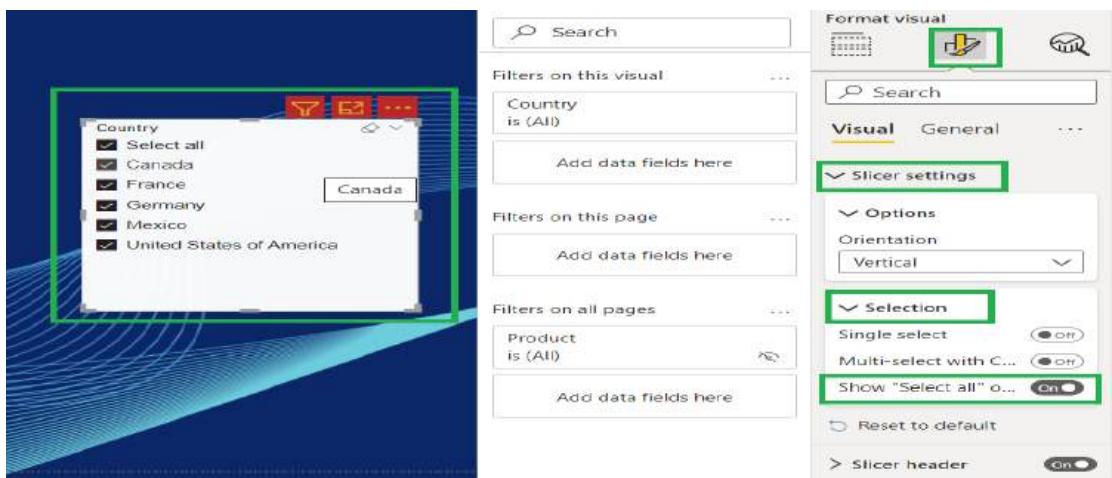
If you want to select multiple values by using **CTRL+CLICK** option then we have to put “Multi-select-with control” option as **ON**.



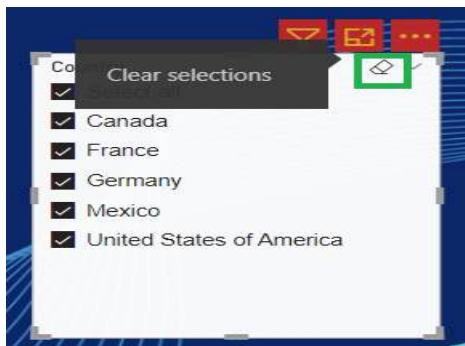
If you put multi-select with control option as in OFF, then without pressing CTRL button, you can select multiple values. You can see those values with tick mark.



If you want, you can put the select all option in the slicer directly, by this, you can select all the values at a time.



If you want to deselect the selected values, you just click on the clear selection symbol at the top right corner of the slicer visual.



You can put the orientation as horizontal as well just by changing it in format orientation section.

A screenshot of a Power BI report. On the left is a horizontal slicer visual with four items: 'Select all', 'Germany', 'Canada', and 'Mexico'. The 'Select all' item is highlighted with a green border. To the right of the slicer is a 'Format visual' pane. In the 'Slicer settings' section, under 'Orientation', the 'Horizontal' option is selected. Other options like 'Vertical' and 'Single select' are also shown. The background of the report features a blue and white abstract design.

You can select the values in this horizontal orientation. Those will be highlighted.



This country related slicer. That's why it is showing “Country” as title name. If you want to hide this name you can simply go to slicer header option and make it OFF. Values you can increase or decrease. Colors of the values you can change in **Values** option.

23. Table: A table is a grid that contains related data in a logical series of rows and columns. It may also contain headers and a row for totals.

Whatever the columns you have mentioned in the **Columns** option, those many columns created and their corresponding aggregated sum will be calculated automatically and total value also displayed in the last. If you want you can **OFF** the total value in the **totals** section of format pane.

If you don't want to summarize and you want data for each country wise then you can simply click on sales column down arrow button and click on “**don't summarize**” option.

Note: You can put the multiple columns also in this table visual, but the relationship should be there for the tables.

24. Matrix: Matrix visualization is very similar to table visualization. In table visualization, data is present in a two-dimensional format. But Matrix visualization can display data in multiple dimensions. It provides full support for stepped layout.

The Matrix aggregates the data itself and lets users drill down. Users can create matrix visuals by using Power BI Desktop and cross-highlight elements within the matrix.

The screenshot shows a Power BI report with a Matrix visual. The Matrix visual displays sales data for various products across different countries. The columns represent countries: Canada, France, Germany, Mexico, and United States. The rows represent products: Amanita, Carrerera, Montaña, Paseo, Vida, and VTT. The total sales for each product across all countries are shown in the bottom row. The 'Build visual' pane on the right shows the configuration for the Matrix visual, with 'Rows' set to 'Product', 'Columns' set to 'Country', and 'Values' set to 'Sales'. The 'Matrix' icon in the pane is highlighted with a green box.

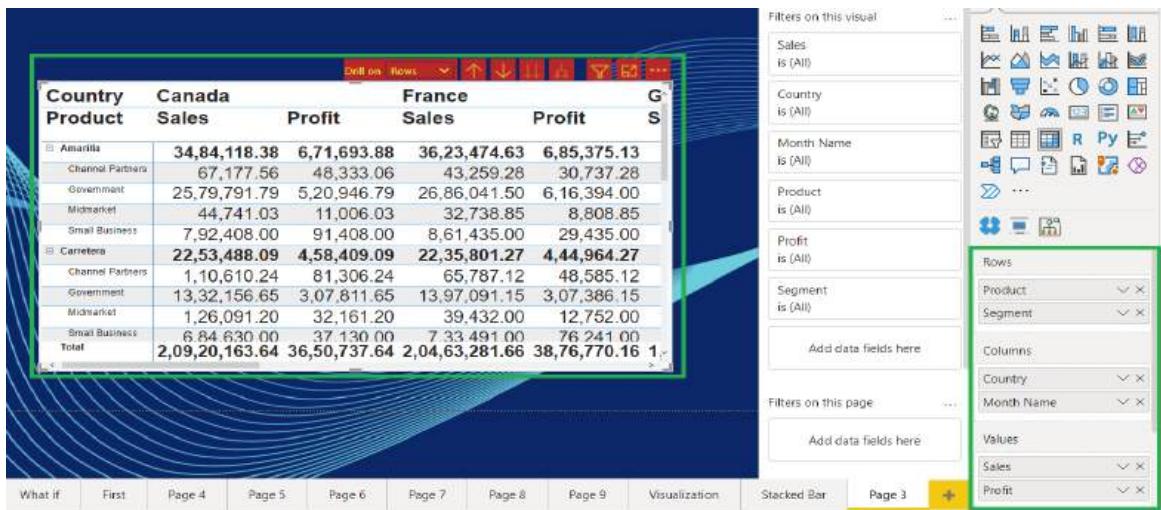
	Canada	France	Germany	Mexico	United States
Amanita	34,84,118.38	36,23,474.63	26,86,344.01	28,43,321.64	
Carrerera	22,53,488.09	22,35,801.27	25,48,843.18	21,62,083.92	
Montaña	23,35,606.53	31,60,099.87	30,68,566.87	16,01,419.31	
Paseo	66,46,378.49	43,47,371.06	41,14,429.74	65,56,447.64	
Vida	25,36,646.59	35,98,552.49	39,55,359.50	15,78,650.39	
VTT	36,63,925.56	34,97,982.34	30,44,971.27	28,91,547.96	
Total	2,09,20,163.64	2,04,63,281.66	1,94,18,514.57	1,76,33,470.86	

Whatever the columns you have mentioned in the **columns** option, those columns will be appeared as column headers and whatever the columns you have mentioned in the **rows** option; those are appeared as row headers.

Whatever the columns you have mentioned in **values** option, those are appeared as cell values to the corresponding columns and rows.

Multiple columns you can put in rows option, columns option and values option. Visual will be generated automatically with the corresponding columns and values.

You can see the total values like category wise that means column wise totals and total values also. See the below screenshot.



POWER BI SERVICE

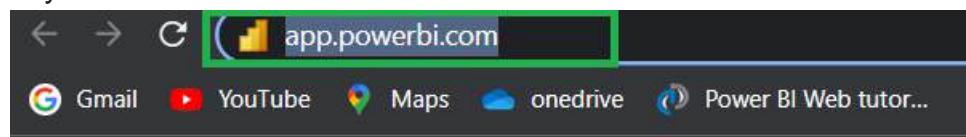
Once development part for report is completed in power bi desktop, then we need to place it to one environment. That environment is called as **power bi service**. From the service itself you can share the reports.

Power BI service is a cloud-based business analytics and data visualization service that enables anyone to visualize and analyze data with greater speed, efficiency, and understanding.

If you want to login to the **Power BI service**, then you have to follow below navigation.

Navigation:

Go to Google and type “app.powerbi.com” and press on “Enter” button in the keyboard



Provide the organizational mail id and click on “Submit”.

The screenshot shows a sign-up form for Power BI. At the top, there is a 'Power BI' logo. Below it, a message says 'Enter your email, we'll check if you need to create a new account.' A large input field labeled 'Email' contains the text 'noyey49538@wirese.com'. To the right of the input field, there are three overlapping yellow bars of increasing height. Below the input field, a small note states: 'By proceeding you acknowledge that if you use your organization's email, your organization may have rights to access and manage your data and account. [Learn more about using your organization's email.](#)' At the bottom of the form, there is a note: 'By clicking Submit, you agree to these [terms and conditions](#) and allow Power BI to get your user and tenant details. [Microsoft Privacy Statement](#)'. A yellow 'Submit' button is located at the bottom left.

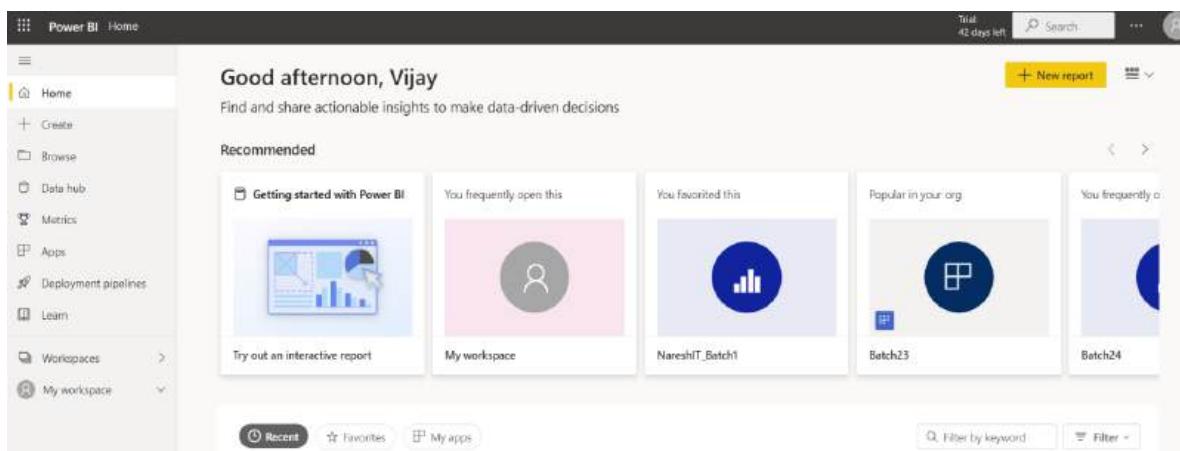
Enter password of the organizational mail id you have given and click on “Sign in”



After that, click on “Yes” for always stay as signed in option.



Now you can see the below screen. That is called power bi service screen.



- 1) **Licenses of the power bi service:** For each and every software; we should require license right. For power bi service there are two types of licenses.
 - Power BI Pro
 - Power BI Premium

Power BI Pro: Power BI Pro is an individual user license that allows access to all content and capabilities in the Power BI service. You can even share content and collaborate with other Pro users.

Only Pro users can publish content to other workspaces, share dashboards, and subscribe to dashboards and reports.

In **Power BI Pro**, you can store up to **10GB** data. That means you can publish reports up to 10GB allocation space. The Pro plan **costs \$9.99/user/month**. Users can try it a **free trial for 60 days** before purchasing the subscription.

If you want to refresh data sets in power bi service, then you can schedule it for 8 times per day only. If you want to refresh more than 8 times scheduled refresh that is not possible with **Power BI Pro**.

Power BI Premium: Power BI Premium is a capacity-based offering that includes: Flexibility to publish reports broadly across an enterprise, without requiring recipients to be licensed individually per user.

With Power BI premium, greater scale and performance than shared capacity in the Power BI service.

In **Power BI Pro**, you can store up to **100GB** data. That means you can publish reports up to 100GB allocation space.

The Premium plan starts at **\$4,995** a month per dedicated cloud compute and storage resource. This does not include the cost of licensing for individual Pro and Free licenses required for each user in your organization.

If you want to refresh data sets in power bi service, then you can schedule it for 48 times per day only. If you want to refresh more than 48 times scheduled refresh that is not possible with **Power BI Premium**.

Note: Along with Pro and Premium versions trial version also there, then you can use that for 60days. That you can find out whether it is trial version or anything. See below screenshot. It will also occupy 10GB of data.

The screenshot shows the Power BI 'My workspace' interface. At the top, it displays 'Vijay Rama Raju Pro trial user'. Below this, a progress bar indicates '41 MB of 10 GB used' (41 MB). A dropdown menu on the right is open, showing options like 'Manage personal storage' (highlighted), 'Notifications', 'Settings' (also highlighted), 'Admin portal', 'Manage gateways', and 'Settings'. The main area lists datasets owned by the user, including '01-query-editor-start', 'Credit_Card_Defaults', 'Example', 'End to End Project', 'Opportunity Analysis Sample', 'sample report', and 'Caloris Xtra Lucky Draw Data'. Each dataset entry includes its name, size, type, last refreshed date, associated dashboard, and manage embed codes link.

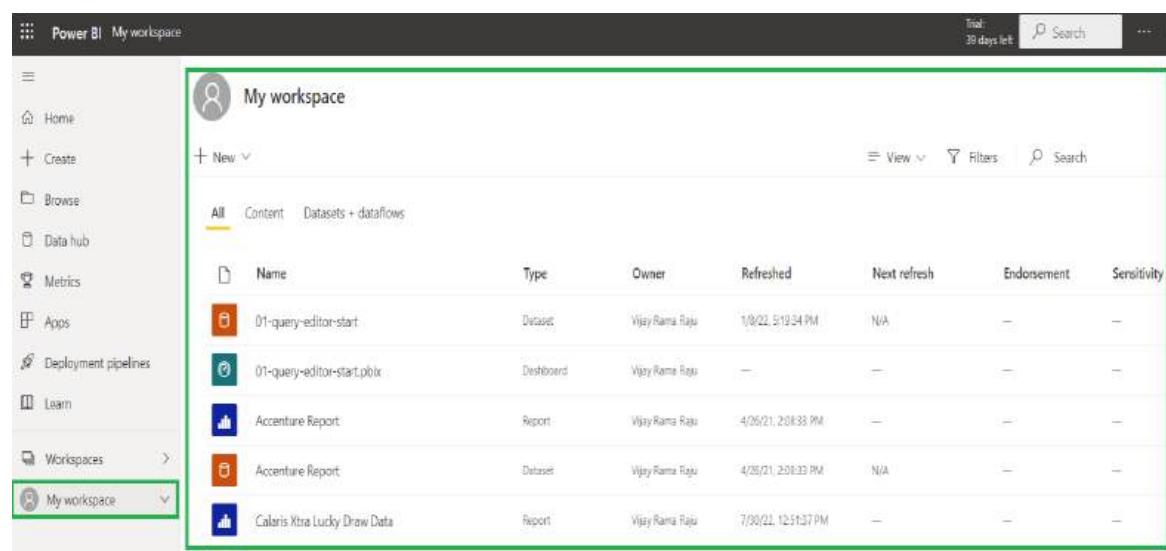
2) Workspaces: The place where your reports put into power bi service is called as workspace. Workspaces are used to collaborate and share content with colleagues.

You can add colleagues to your workspaces and collaborate on dashboards, reports, workbooks, and datasets. With one exception, each workspace member needs a Power BI Pro or Premium Per User (PPU) license.

After login to the power bi service, we have to access the workspace to put your reports.

There are two types of workspaces in power bi service. One is **My Workspace** and another one is **Create Workspace**.

My Workspace: This is your personal workspace. You can publish your reports into my workspace but you can't share your reports to the multiple users. So in organizational level, we will not use My workspace always we will use "**Create Workspace**" option.



The screenshot shows the 'My workspace' section of the Power BI service. On the left, there's a sidebar with navigation links: Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, Learn, Workspaces (with a dropdown menu), and My workspace (which is selected and highlighted with a green border). The main area is titled 'My workspace' and contains a table of contents. At the top of the table are buttons for '+ New' (with options for Content, Datasets + dataflows, and Reports), 'View' (dropdown), 'Filters' (button), and 'Search' (input field). The table has columns for Name, Type, Owner, Refreshed, Next refresh, Endorsement, and Sensitivity. The data includes:

Name	Type	Owner	Refreshed	Next refresh	Endorsement	Sensitivity
01-query-editor-start	Dataset	Vijay Rama Raju	1/8/22, 5:19:34 PM	N/A	—	—
01-query-editor-start.pbix	Dashboard	Vijay Rama Raju	—	—	—	—
Accenture Report	Report	Vijay Rama Raju	4/26/21, 2:08:33 PM	—	—	—
Accenture Report	Dataset	Vijay Rama Raju	4/26/21, 2:08:33 PM	N/A	—	—
Caloris Xtra Lucky Draw Data	Report	Vijay Rama Raju	7/30/21, 12:51:37 PM	—	—	—

Create Workspace: This is used to create a new workspace. This is useful for holding the power bi reports, dash boards, apps etc. and you can share the reports to the multiple users.

To create workspace, go to power bi service and then **Home** tab. Then you can find "**My Workspace**" option. Then click on "**Create Workspace**" option. It will pop up a window.

There you have to provide **Workspace name**. That should be unique and you have to give the **description** of that workspace.

Type	Owner	Refreshed	Next refresh	Endorsement	Sensitivity
Dataset	Vijay Rama Raju	1/8/22, 5:19:34 PM	N/A	—	—
Dashboard	Vijay Rama Raju	—	—	—	—
Report	Vijay Rama Raju	4/26/21, 2:00:33 PM	—	—	—
Dataset	Vijay Rama Raju	4/26/21, 2:00:33 PM	N/A	—	—
Report	Vijay Rama Raju	7/10/22, 12:51:37 PM	—	—	—
Dataset	Vijay Rama Raju	7/10/22, 12:51:37 PM	N/A	—	—
Report	Vijay Rama Raju	7/13/22, 8:30:57 PM	—	—	—
Dataset	Vijay Rama Raju	7/13/22, 8:30:57 PM	N/A	—	—
Report	Vijay Rama Raju	7/20/22, 10:02:10 AM	—	—	—

Create a workspace

Workspace image


Workspace name

Available

Description

[Learn more about workspace settings](#)

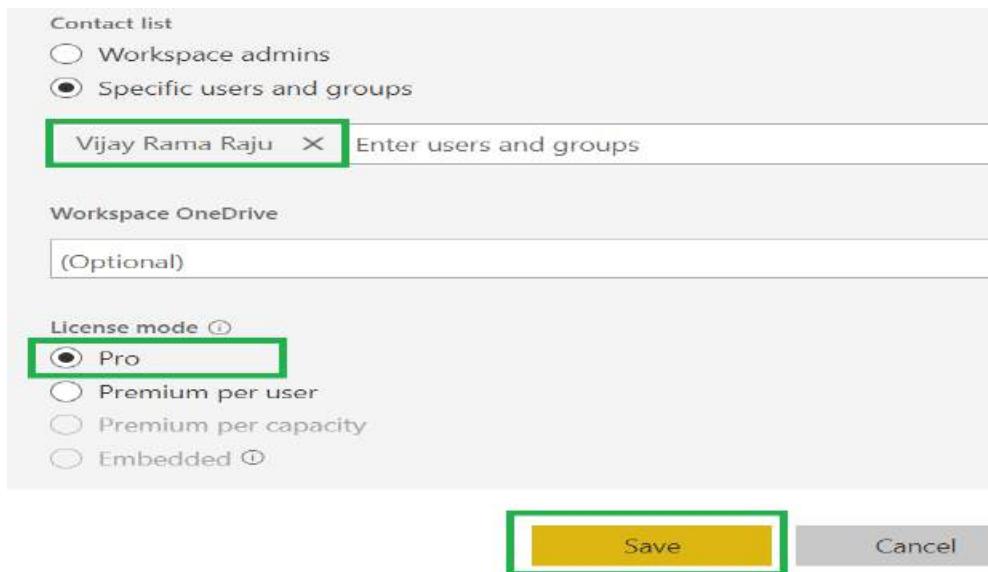
Advanced: If you want to provide access of this workspace to another user then you can use these advance options.

Workspace admins: By default, this option will be enabled. With this, Admin only can access the workspace.

Specific users and groups: We can mention the specific users mail ids to provide an access the workspace.

At the time of creation of workspace, only we should provide the mail ids of the users to give an access to that workspace.

Workspace OneDrive: We can provide the OneDrive link to save the workspace name in one drive.



If you want to see the created workspace, then you can go to “**Workspaces**” option and search for the **name** of the workspace.

Type	Owner	Refreshed	Next refresh	Endorsement	Sensitivity
Dashboard	Batch261	—	—	—	—
Report	Batch261	10/15/23, 9:07:32 AM	—	—	—
Dataset	Batch261	10/15/23, 9:07:32 PM	N/A	—	—

Once workspace created, then you have to give the roles of the users to access this workspace. There are different roles for users to access workspace.

⋮ Power BI Batch261

Owner
Batch261
Batch261
Batch261

If you want to give the access to workspace to the user, then you have to **select the workspace and click on ellipse (dots) and click on workspace access**.

Now you have to provide mail id of the user to whom we need to give the access and you have to assign a proper role to that user. Then only workspace access will be provided to that user as per the role assigned to him.

Workspaces allow you to assign roles to individuals, and also user groups such as security groups, Microsoft 365 groups, and distribution lists.

NAME	PERMISSION
Vijay Rama Raju	Admin

Note: You can put N number of reports in a single workspace. You can put the reports up to 10GB.

3) Roles for workspace to access: There are 4 different roles available.

- **Viewer** - This role provides read only access to workspace items. Read access does provide report / dashboard consumers the ability to not only view, but also interact with visuals.

Interaction does not mean changing a visual. Also note that users in this view do not require a Pro License to view reports if the workspace is in Premium mode. Without Premium content a Pro License is required. They can't share the reports; they can just view the reports.

- **Contributor** - This role can access and interact with reports and dashboards. Additionally, this role can create, edit, copy, and delete items in a workspace, publish reports, schedule refreshes, and modify gateways. But they cannot share reports through app. That means text box option will not be in enable mode just like viewer.
- **Member** - This role can access and interact with reports and dashboards. Additionally, this role can create, edit, copy, and delete items in a workspace, publish reports, schedule refreshes, and modify gateways. Share the reports via app as well and mail ids will be in enable mode in this role.

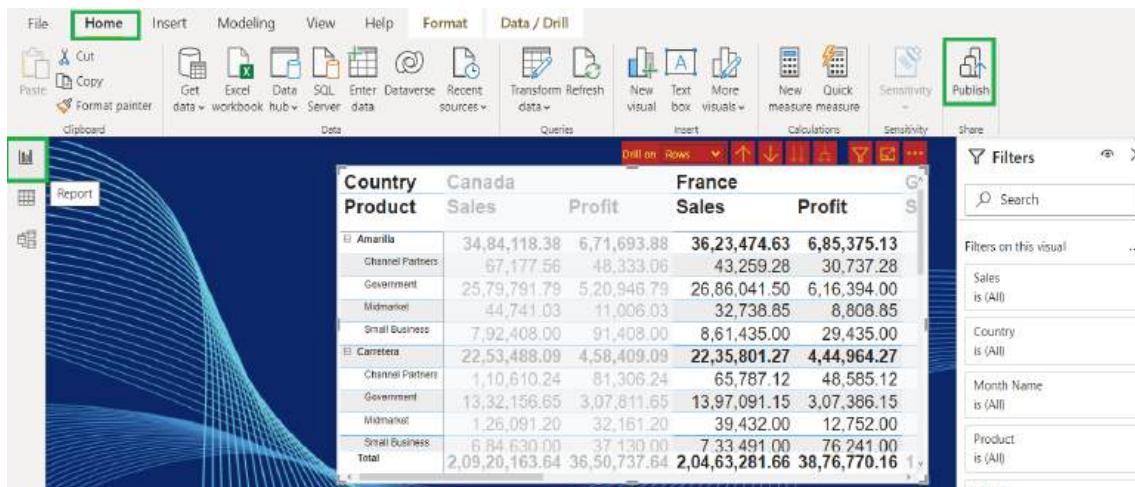
Finally, members of this role can also feature dashboards on the service, share items, allow others to re-share items, publish or republish items. This role is also able to add other users to the viewer or contributor role.

- **Admin** - This role can do all the functions above plus add and remove all users including other Admins. Members of this role can create, edit, copy,

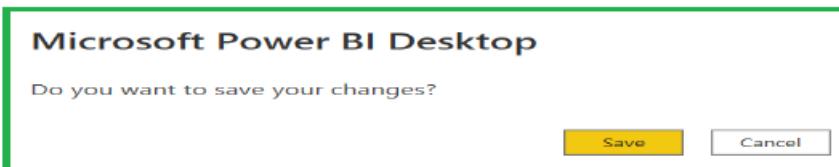
delete items and **Add Admins** and do all the things. They can share the reports via **App** as well.

- 4) **Sharing a report using share option:** Once create a workspace on power bi service, and then we have to publish reports from power bi desktop to this newly created workspace.

Navigation: Go to “**Power BI Desktop**” -> **Report** tab -> Select “**Home**” ribbon -> Click on “**Publish**” button. Immediately it will ask to “**Save**” the report. Click on “**Save**” button.



The screenshot shows the Microsoft Power BI Desktop interface. The ribbon is at the top with the "Home" tab selected. Below the ribbon is a report area containing two tables: "Canada" and "France". The "Canada" table has columns for Product, Sales, and Profit. The "France" table also has columns for Sales and Profit. On the right side of the screen, there is a "Filters" pane with several dropdown menus for Sales, Country, Month Name, and Product, all set to "(All)".



Again **new window** will open. That window will tell you on which workspace we need to publish your report. Select the workspace name “**Batch26**” and click on “**Select**” button. Finally click on “**Got it**”. Then go to power bi service and check whether the report is published or not.



Note: For the first time when you want to publish your report, then it will ask credentials like mail id and password. Then provide the workspace name where we want to publish the reports.

Once report is published in to the workspace of power bi service, then you can see three tabs for that report.

⇒ All → Reports, dashboards, data sets will be available in this tab.

⇒ Content--→ Reports will be available in this tab.

⇒ Datasets + Dataflows → Data sets and data flows only will be available in this tab

(When you publish report from desktop, along with report data sets will be published into the workspace. But data sets will be combined together and visible as single dataset. Here data set name is by default created as same as the report name in power bi service)

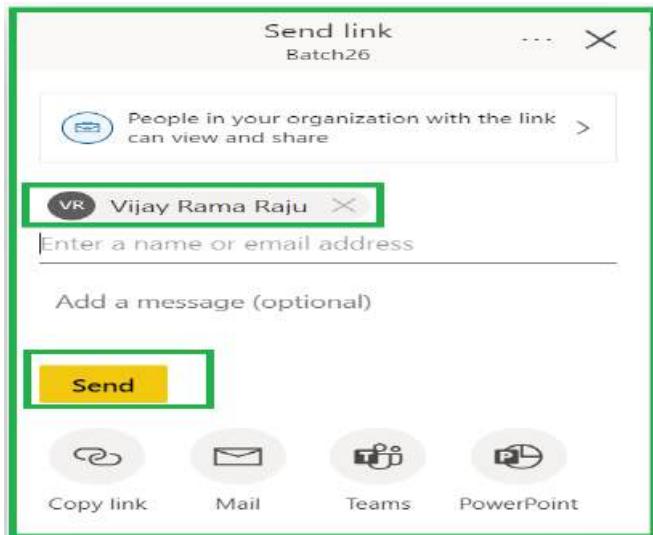
If you want to see all the visuals and pages of the report, then you have to go to the content tab and click on “batch26” report.

The screenshot shows the Power BI Content page. On the left, there's a navigation menu with options like Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, Learn, Workspaces, and a dropdown for the current workspace, "Batch26". The main area is titled "Batch26" and shows a list of items under "Content". There are two items listed: "Batch26" (Dashboard) and "Batch26" (Report). The "Report" item is highlighted with a green border. At the top right, there are buttons for View, Filters, Settings, Access, and Search.



Note: If you want to share a report, you must have to mouse-hover on that report, and then only share symbol will be appeared. There you have to provide mail ids of the users and click on “Send” button.

The screenshot shows the Power BI Content page again, with the "Batch26" workspace selected. The "Batch26" report item is highlighted with a green border. A "Share" icon is visible next to the report item. The rest of the interface is similar to the first screenshot, with the navigation menu and content list.

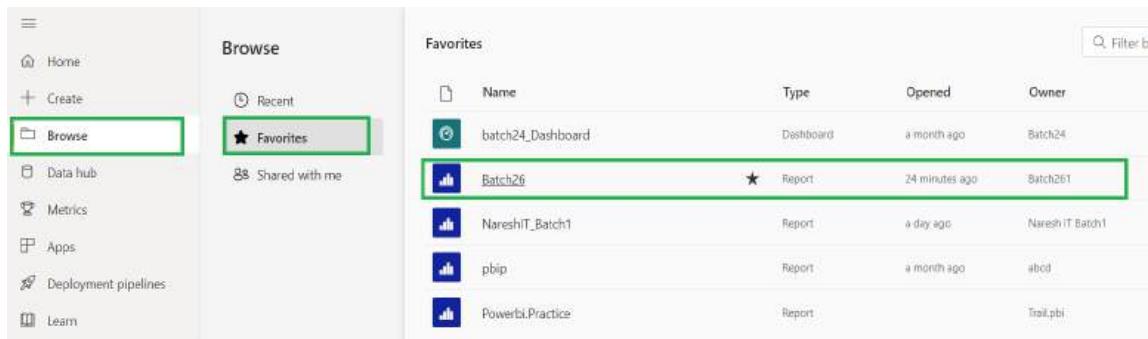


You can mention multiple user mail ids and click on send to share the reports. They will see the reports by logging in to the power bi service then go to “**Browse**” option. And then click on “**Shared with me**” option. They can view the reports on their **shared with me** option.

We have shared a report only but haven't provide an access on this report.

- 5) Add favorites:** Whenever lot of reports is there in a workspace, but we need to make some reports as favorites then will use this option. That means important reports can be made as favorites for better identification. Daily routines we can put as favorites in this option by clicking on star (*) button.

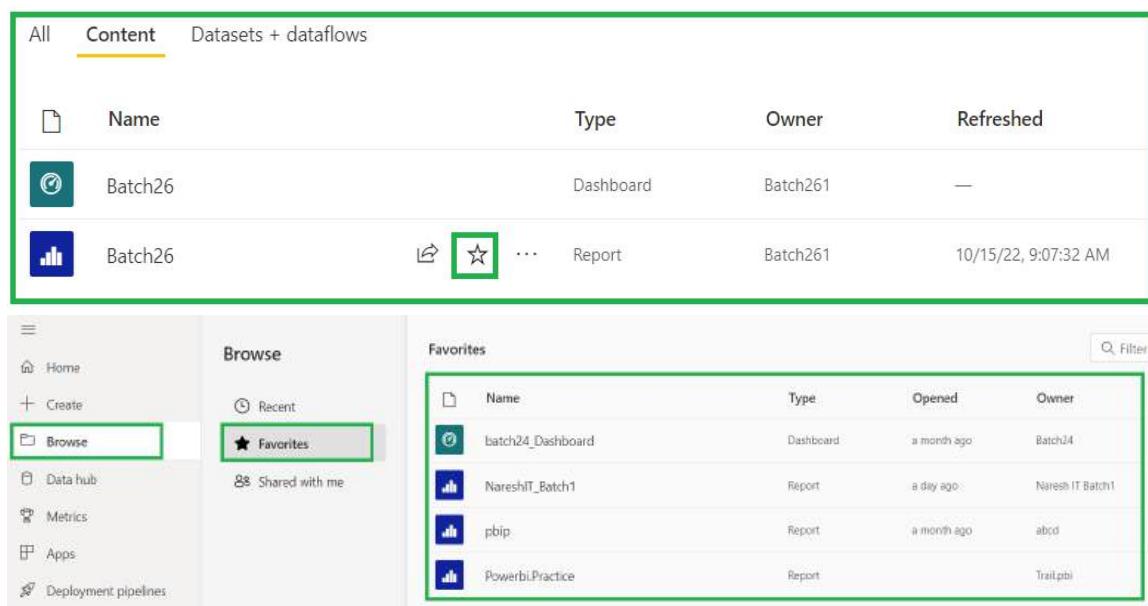
To see the favorite reports then you can go to “**Browse**” option and click on “**Favorites**” option. You can see and access the easily.



The screenshot shows the Power BI service interface. On the left, there's a sidebar with options like Home, Create, Browse (which is selected and highlighted with a green box), Data hub, Metrics, Apps, Deployment pipelines, and Learn. In the center, under the “Browse” tab, there are sections for Recent, Favorites (also highlighted with a green box), and Shared with me. To the right, a table titled “Favorites” lists reports. The table has columns for Name, Type, Opened, and Owner. One report, “Batch26”, is highlighted with a green box and has a star icon next to its name, indicating it is a favorite.

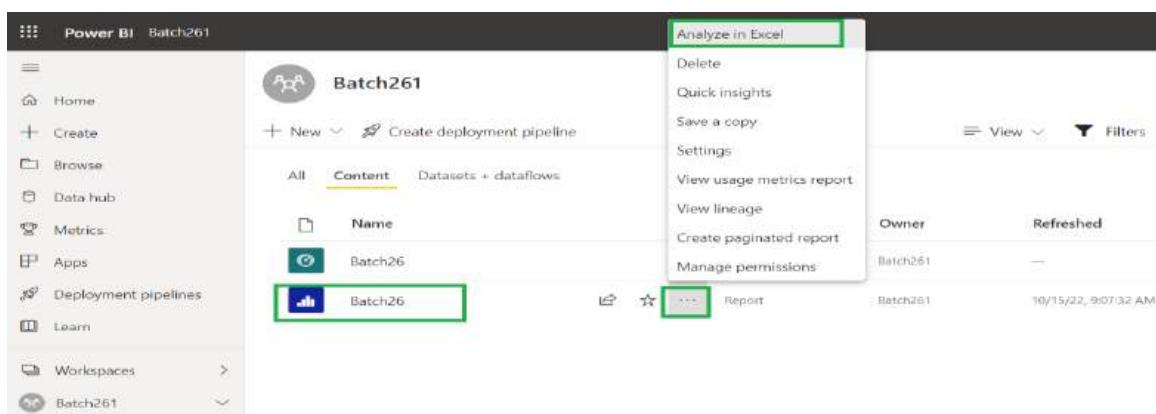
Name	Type	Opened	Owner
batch24_Dashboard	Dashboard	a month ago	Batch24
Batch26	Report	24 minutes ago	Batch261
NareshIT_Batch1	Report	a day ago	NareshIT Batch1
pbi	Report	a month ago	abcd
PowerBI.Practice	Report		Trail.pbi

If you click on star symbol again, that means if you uncheck the report, then it will not be visible in favorites.



This screenshot shows two parts of the Power BI service interface. The top part is a modal window titled “Content” showing datasets and dataflows. It has columns for Name, Type, Owner, and Refreshed. Two items are listed: “Batch26” (Dashboard) and “Batch26” (Report). The report item has a star icon next to it, which is highlighted with a green box. The bottom part is the main interface, showing the “Content” tab under the “Browse” tab. It has sections for Recent, Favorites (highlighted with a green box), and Shared with me. A table titled “Favorites” lists reports, similar to the one in the first screenshot, with “Batch26” being the highlighted favorite.

Analyze in Excel: With Analyze in Excel, you can bring Power BI datasets into Excel, view, and interact with them using PivotTables, charts, slicers, and other Excel features; If you click on this option, then report will be downloaded in to an excel format.



This screenshot shows the Power BI service interface with a report named “Batch26” selected. A context menu is open over the report, with “Analyze in Excel” highlighted with a green box. The menu also includes options like Delete, Quick insights, Save a copy, Settings, View usage metrics report, View lineage, Create paginated report, and Manage permissions. The main interface shows the “Content” tab under the “Browse” tab, with a table listing reports and their details like Type, Owner, and Refreshed.

Delete: If you want to delete a report, then click on delete option in the ellipse button available on the right side of the report. If you click on delete option, then report only will be deleted, but not deletes the data set.

The screenshot shows the Power BI Content page for a workspace named 'Batch261'. On the left, there's a navigation bar with options like Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, and Learn. The main area displays a list of items under 'Content'. A report titled 'Batch26' is selected, indicated by a green border around its row. A context menu is open over this report, with the 'Delete' option highlighted in green. Other options in the menu include Analyze in Excel, Quick insights, Save a copy, Settings, View usage metrics report, View lineage, Create paginated report, and Manage permissions.

Delete report

Are you sure you want to permanently delete
Batch26?

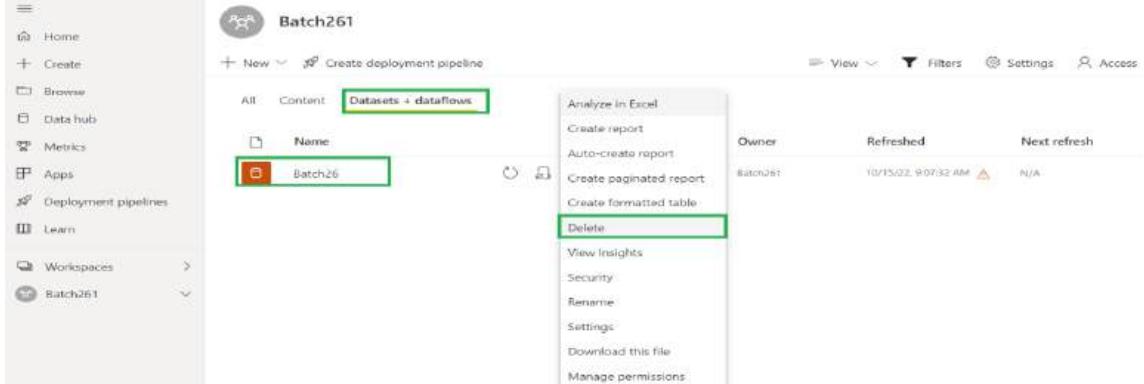
Delete **Cancel**

6) Difference between delete option in content and delete option in datasets + data flows

If you delete report in content tab, then report will be deleted but dataset for the corresponding report in datasets + dataflows option will not be deleted.

The screenshot shows the Power BI Content page for a workspace named 'Batch261'. The navigation bar on the left is identical to the previous screenshot. In the main content area, the 'Datasets + dataflows' tab is selected, indicated by a green border. A dataset named 'Batch26' is selected, indicated by a green border around its row. The dataset details show it is a Dataset type, owned by 'Batch261', last refreshed on '10/15/22, 9:07:32 AM', and has 'N/A' for the next refresh. Other tabs visible include 'Content' and 'All'.

But if you delete dataset for the report in the datasets + dataflows option, then automatically corresponding report will also be deleted.

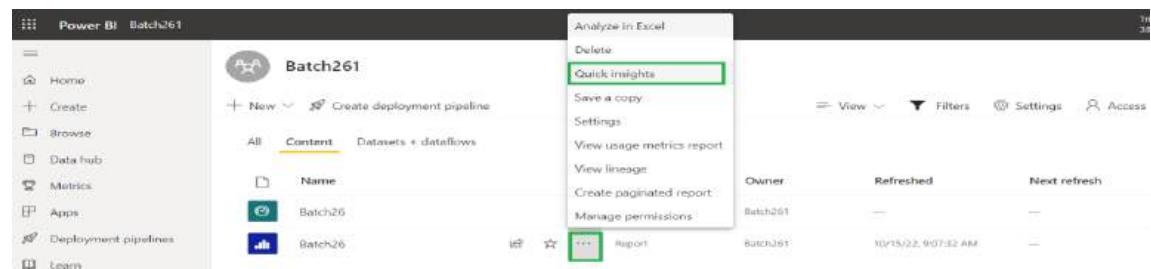
7) 

Quick Insights: By default, Microsoft will give the suggestions for your data sets. Once you publish your report and data set from desktop to service then you can use this option.

This option will give you the clear suggestion about the reports. What type of visuals we can use for this data... like this.

Quick Insights are a feature of Power BI that analyses data sets and finds patterns, trends and outliers. Once your data is uploaded to the Power BI cloud, you can use the Quick Insights feature.

Once you click on this option, based on your data some visuals will be created. You can see them in “View Insights” option.





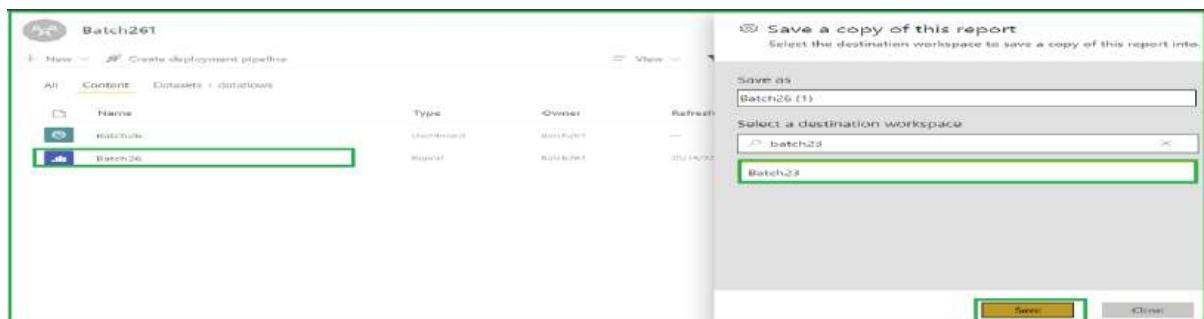
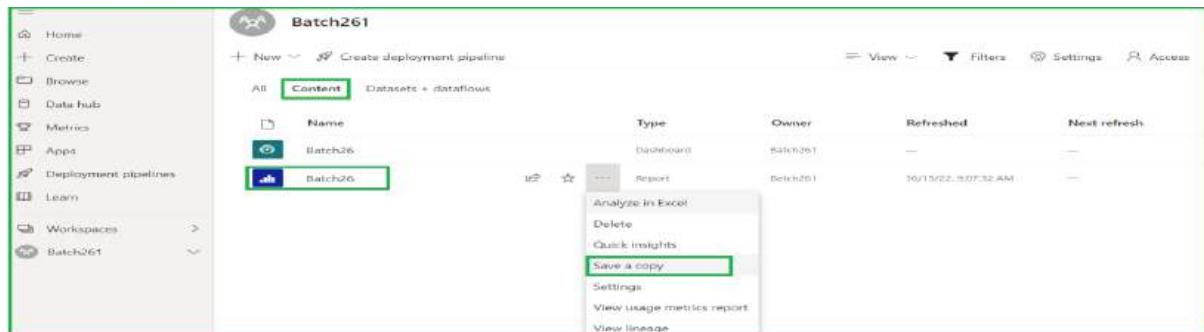
Whatever the columns we have in your data, based on that data Microsoft will give you the suggestions like what type of visuals we can generate what type of

columns we have to use. Like this, if it is ok, then you can create the same visuals in your power bi desktop.

8) Moving a report from one workspace to another workspace:

Save a copy: If you want to move a report from one workspace to another workspace then we will use this option.

If you click on this option, then it will ask the destination workspace name. If you provide a name of that workspace and click on “**Save**” button, then copy of the report will be saved in the destination workspace.



Then you can see the same batch26 report in the batch23 workspace also. See the below screen shot.



Note: You can move a report from one workspace to another workspace but you can't move the data set from one workspace to another.

In the above example, we have moved **batch26** report to **Batch23** workspace. Same report is available in **Batch23** workspace but corresponding data set will not be available in **Batch23** workspace. See the below screen shot.

The screenshot shows the Power BI service dashboard. On the left, there's a sidebar with options like Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, and Learn. The main area has a title 'Batch23'. Below it, there are tabs for All, Content, and Datasets + dataflows (which is highlighted with a green box). Under 'Content', there's a table with columns: Name, Type, Owner, Refreshed, and Next refresh. It lists two items: 'Batch23' (Dataset, Batch23, 6/25/22, 11:12:40 AM, N/A) and 'Batch25' (Dataset, Batch23, 7/11/22, 10:30:11 AM, N/A).

Settings: If you want to change the name of the report, description, and contact mail id. Then you can use settings option in the report. You can upload the snapshot of the report and you can delete the snapshot as well. Once changes done; you have to click on “Save” button to save the changes of the report.

This screenshot shows the 'Content' tab selected in the Power BI service. It lists two reports: 'Batch26' (Dashboard, Batch26, —) and 'Batch26' (Report, Batch26, 10/15/22, 9:07:32 AM). Over the second 'Batch26' entry, a context menu is open, with 'Settings' highlighted by a green box. Other options in the menu include Analyze in Excel, Delete, Quick insights, Save a copy, View usage metrics report, and View lineage.

This screenshot shows the 'Settings' dialog for the 'Batch26' report. It includes fields for Report name (Batch2B), Description, Contact (Vijay Rama Raju), Snapshot (Upload, Delete), and Endorsement (Help coworkers find your quality content by endorsing this report). At the bottom are 'Save' and 'Cancel' buttons.

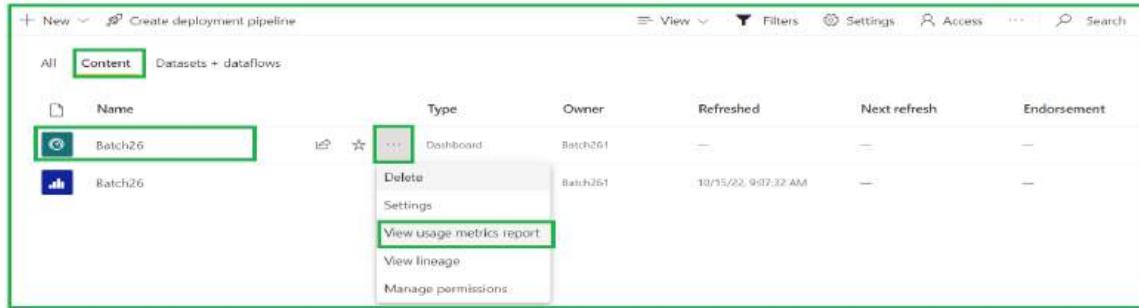
9) View Usage Metrics Report: Usage metrics is one of the features in Power BI Service. The usage metrics report will give you an analysis of how many times the content is viewed or share, through which platforms (web or mobile), and by which users.

It will store the details of the number of viewers of a report and how many users are viewing this report per day.

How many times a user has viewed the report, which users are viewing like this can be handled. You can find out the users by their names and mail ids.

View usage metrics are two types. 1) Dashboard usage metrics and 2) Report usage metrics

- 1) **Dashboard usage metrics:** In order to see Dashboard usage metrics, you can go to Workspaces and select any workspace then go to Dashboards tab and then click on View Usage metrics option in ellipse symbol.

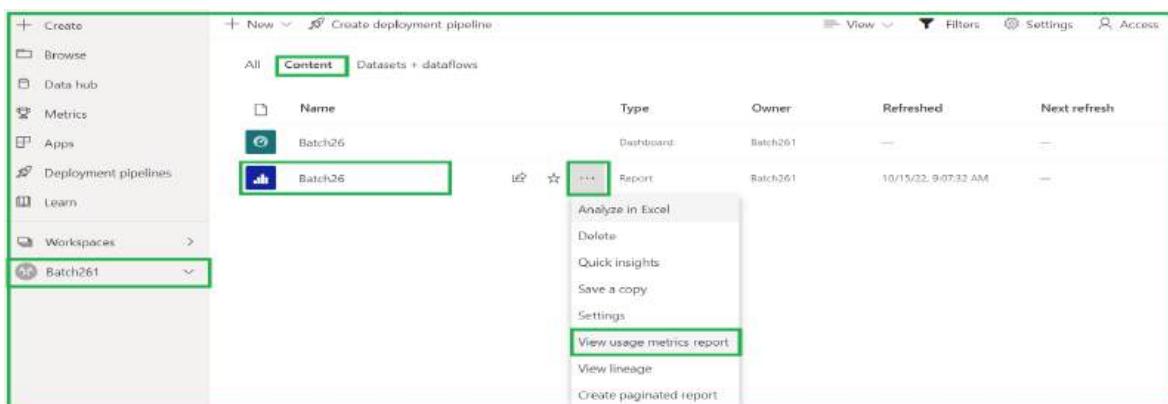


Once you clicked on Usage metrics button, you can see a report with visuals describing consumption of dashboard with users.

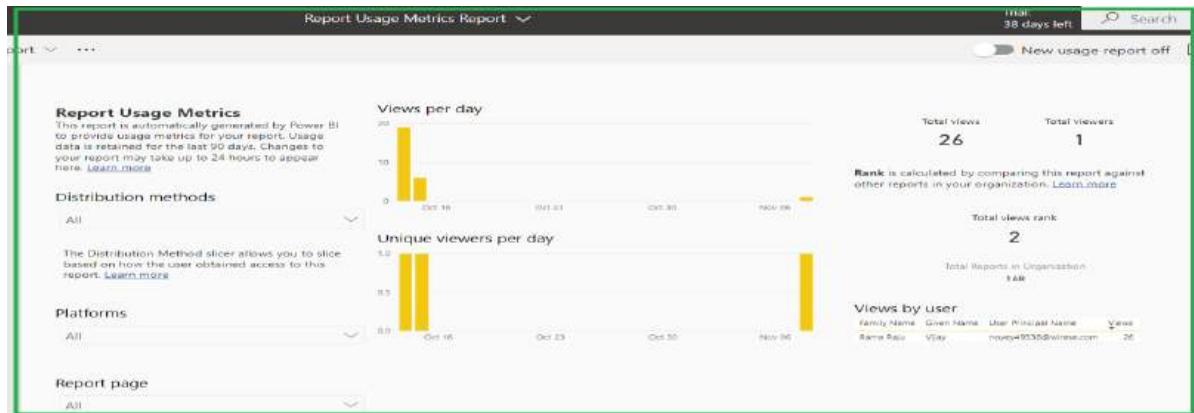
Note: You will not able to see usage metrics for shared reports



- 2) **Report usage metrics:** In order to see Report usage metrics, you can go to Workspaces and select any workspace then go to Content tab and mouse-hover on any report and then click on ellipse symbol. Now select “View usage metrics” option.



Once you clicked on usage metrics button, you can see a report with visuals describing consumption of reports with users.



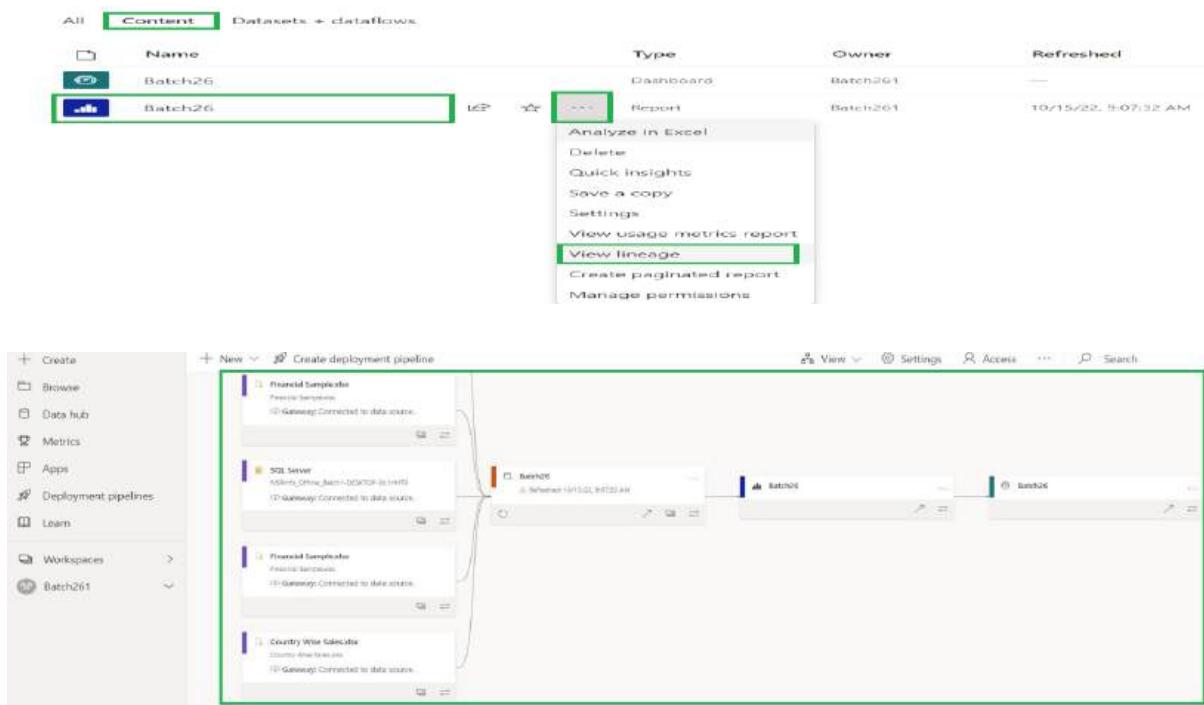
Note: It will show the usage metrics in the form of views per day, unique viewers per day, and views by user.

10) View Lineage: Lineage view is a feature within the Power BI Service which gives a single relational view of all the data and visual building blocks of the Power BI workspace.

This enables better management, troubleshooting, and impact analysis in the flow of data from the initial data source to the end user dashboard.

That means what data sources we have used and which reports we have created, what are the data flows for this report we have used can be analyzed here.

If you click on “View Lineage” option, it will show you the data sets, dashboards, reports, data flows in a tree structure. You can search for the particular building block to access quickly.



Once you click on the report, you can see all the visuals, pages of that report in order. See below screen shot.

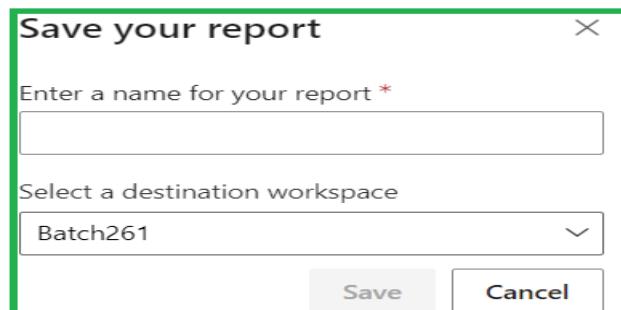
The screenshot shows the Power BI workspace interface. On the left, there's a sidebar with navigation options like Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, Learn, Workspaces, and the current workspace, Batch261. The main area displays the 'Content' tab under 'All'. A list of items is shown, with 'Batch26' and 'Batch26' highlighted in green boxes. Below this, a detailed report page for 'Batch26' is displayed. The page includes a navigation bar with 'File', 'Export', 'Share', 'Chat in Teams', 'Get insights', 'Subscribe', 'Edit', and '...'. The report content features several visualizations: a donut chart titled 'Discounts by Product' with values 804.47K (45.26%), 259.64K (13.57%), 232.41K (12.48%), 144.18K (8.11%), and 111.25K (6.23%); a bar chart titled 'COGS by Country' showing Mexico at 18.04M and Germany at 118.73M; and a table titled 'COGS' listing products and their COGS values. The table data is as follows:

Product	Mexico	Total
Amarilla	2,378,944.00	2,378,944.00
Chanel Partners	2,045,540.00	2,045,540.00
Enterprise	2,652,000.00	2,652,000.00
Small Business	1,101,930.00	1,101,930.00
Montañesa	802,700.00	802,700.00
Small Business	1,105,500.00	1,105,500.00
Carretera	2,495,933.00	2,495,933.00
Montaña	1,603,640.00	1,603,640.00
Paseo	6,692,980.00	6,692,980.00
Velo	2,877,434.00	2,877,434.00
VTT	2,596,798.00	2,596,798.00
Total	18,041,829.00	18,041,829.00

File Tab: In this option, you can see the options like save a copy, download, manage permissions, print this page, Generate QR code and settings.

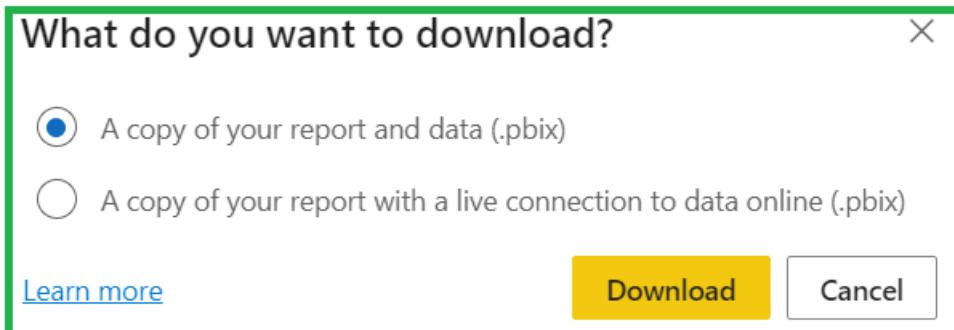
Save a copy: If you click on this option, it will ask you to enter the name of the report and destination workspace name. Once provided, report will be saved in to the destination workspace.

This screenshot shows the same report page from above, but with the 'File' menu open. The 'Save a copy' option is highlighted with a green box. Other options visible in the menu include 'Download this file', 'Manage permissions', 'Print this page', 'Embed report', 'Generate a QR code', and 'Settings'. The report content itself remains the same, displaying the donut chart, bar chart, and COGS table.



Download this file: Whenever you want to modify the report from the particular workspace, then you have to first download the report from that workspace and do the modifications in power bi desktop and publish it again it to the service.

Once you click on “**Download this file**” option it will pop up another window stating that which type of download you required. Once specified, that type of file will be downloaded.



Manage Permissions: If you want to give permission to access this report, then you can approve it or else you can deny it if not required.

Pages

- Home
- Create
- Browse
- Data hub
- Metrics
- Apps
- Deployment pipelines
- Learn

Product

Discounts by Product

Product	Paseo	Velo	Carretera	Amarilla
804.47K (45.26%)	259.64K	232.41K	225.64K	
	144.18K	111.25K		

COGS by Country

118.73M | 118,726,350.26 | 118,726,350.26

18.04M

Batch26

Related content

- Dashboards
- Workbooks
- Datasets

Link

People

VR

Grant people access

Vijay Rama Raju

Enter a name or email address

Allow recipients to share this report

Allow recipients to build content with the data associated with this report

Send an email notification

Add a message (optional)

Grant access Cancel

Print this page: If you want to take a printout of the currently viewing page then you can click on this option. You can print either in color or black and white, you can print page either in portrait or landscape. You can print all pages or specific page if you want.

Pages

- Page 1
- Test
- Page 2
- Page 3
- Page 4
- Page 5
- Page 6
- Page 7

Product

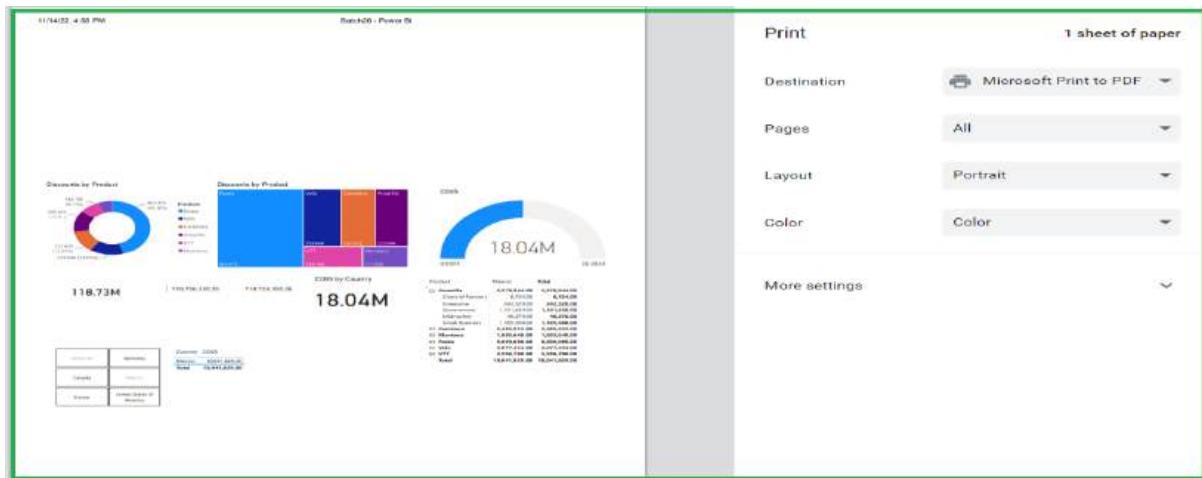
Discounts by Product

Product	Paseo	Velo	Carretera	Amarilla
804.47K (45.26%)	259.64K	232.41K	225.64K	
	144.18K	111.25K		

COGS by Country

118.73M | 118,726,350.26 | 118,726,350.26

18.04M



Export: You can export the report in three ways. That means power bi files can be exported in 3 ways.

- ⇒ Analyze in Excel -> You can download power bi file in excel format
- ⇒ Power Point -> You can download power point file as an image or live data
- ⇒ PDF-> You can download the report as PDF

Chat in Teams: You can open a Teams chat about Power BI dashboards, reports, visuals, and datasets directly from the Power BI service.

Use the **Chat in Teams** feature to quickly start conversations when you view reports, dashboards, and datasets in the Power BI service.

That means you call the report from the Microsoft teams chat. For example Video call, chat etc we can do in this option.

To use the **Chat in Teams** functionality in Power BI, make sure your Power BI administrator hasn't disabled the Share to Teams tenant setting in the Power BI admin portal. This setting allows organizations to hide the Chat in Teams buttons.

You can use this as conversations tool for Microsoft teams. You can put the reports and all by using this chat in team's option.

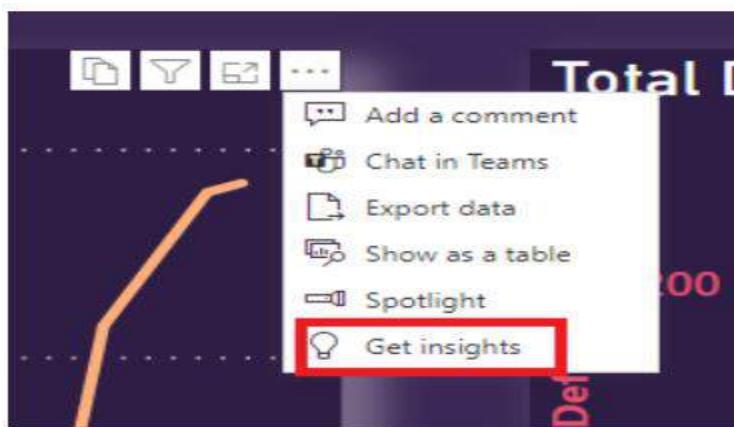
Get insights: This Insights feature helps you easily explore and find insights such as anomalies and trends in your data as you interact and consume your reports. It notifies you if there are interesting insights and provides explanations for them.

Select Get insights in the action bar to open the Insights pane.



The pane only shows insights about the current report page and it updates when you select a different page on the report.

Select More options (...) in the upper-right corner of a visual and then Get insights to see insights about just that visual.



Insights: The Insights pane currently shows you three types of insights – Anomalies, Trends, and KPI analysis. The Top tab shows you Top insights. All tab shows you both Top insights and other insights.

The screenshot shows the 'Insights' section of a Power BI report. At the top, there are navigation links for 'Top' (which is highlighted in yellow) and 'All'. Below this, a section titled 'Anomalies' is expanded, showing a blue line graph icon and the text 'Recent anomaly in Defect Qty'. It states that the most recent anomaly was on Thursday, April 24, 2014, when 'Defect Qty' had a high value of 1721128. Another section titled 'Trends' is also visible, showing a blue line graph icon and the text 'Recent trend in Units Produced'. It states that 'Units Produced' started trending up on Tuesday, July 1, 2014, rising by 38.92% (2793968) in 6 months.

Insights are computed every time you open a report or interact with a report such as changing pages, changing filters, or cross-filtering your data.

Anomalies: An anomaly is an abnormality in time-series data, such as unexpected spikes and dips in the data.

The algorithm computes a boundary around what's considered a normal or expected value. Any value found outside this boundary is marked as an anomaly.

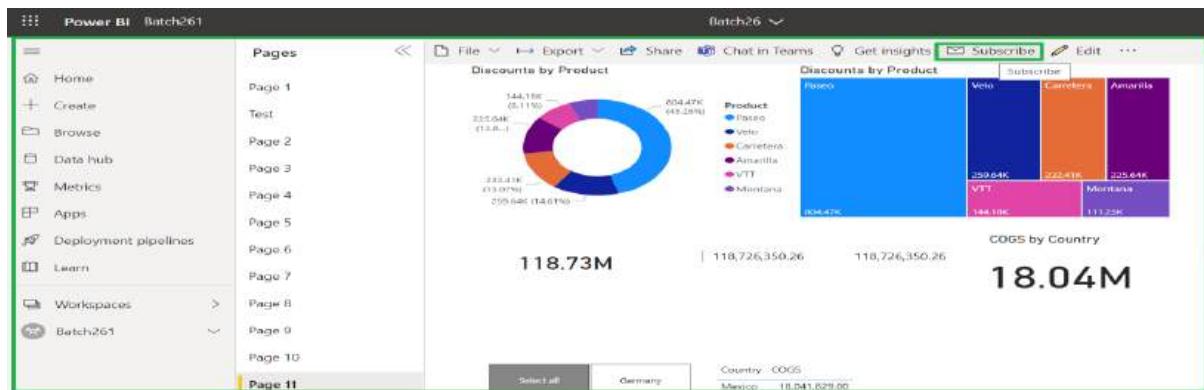
Trends: A trend occurs there's a prolonged increase or decrease in time-series data.

11) Subscribe: Power BI subscription is a great way to receive an up-to-date copy of the report on a scheduled basis in the email inbox of the Power BI users.

Power BI reports and dashboards are not static often. As the data changes, the report and dataset would also change (depending on the frequency of the refresh of the dataset).

Creating and managing a Subscription is very simple. When you open a report in the Power BI service (or website), then you will see a **Subscribe** option at the top of it.

Subscribe is nothing but sending a report through mail on scheduled basis.



This screenshot shows the "Subscribe to emails" dialog for "Page 11". It includes fields for "Enter email addresses" (Vijay Rama Raju), "Subject" (Subject), "Include an optional message...", "Report page" (Page 11), and "Frequency" (Daily). A "Run now" button is set to "On".

We have to provide mail ids of the users to whom you want to send the report. Subject you need to mention before sending a report. If you want to type an optional message, you can.

By using subscribe option, we can send single page of the report; we can't send the entire report. Selected page only can be sent.

This screenshot shows the "Subscribe to emails" dialog with more detailed settings. It includes "Frequency" (Daily), "Scheduled time" (9:00 AM, PM, UTC+05:30 Chennai, Kolkata, Mu), "Start date" (11/14/2022) and "End date" (M/d/yyyy), and checkboxes for "Permission to access the report in Power BI", "Link to report in Power BI", "Preview image", and "Full report attachment". A note states "Emails will be sent daily at 09:00 PM India Standard Time starting 11/14/2022, 12:00:00". Buttons for "Save and close" and "Cancel" are at the bottom.

Frequency: On which basis, the report has to send. That means hourly, daily, Weekly, Monthly.

Frequency

Hourly

Sun Mon Tue Wed Thu Fri

Scheduled time

9 00 PM (UTC+05:30) Chennai, Kolkata, Mu

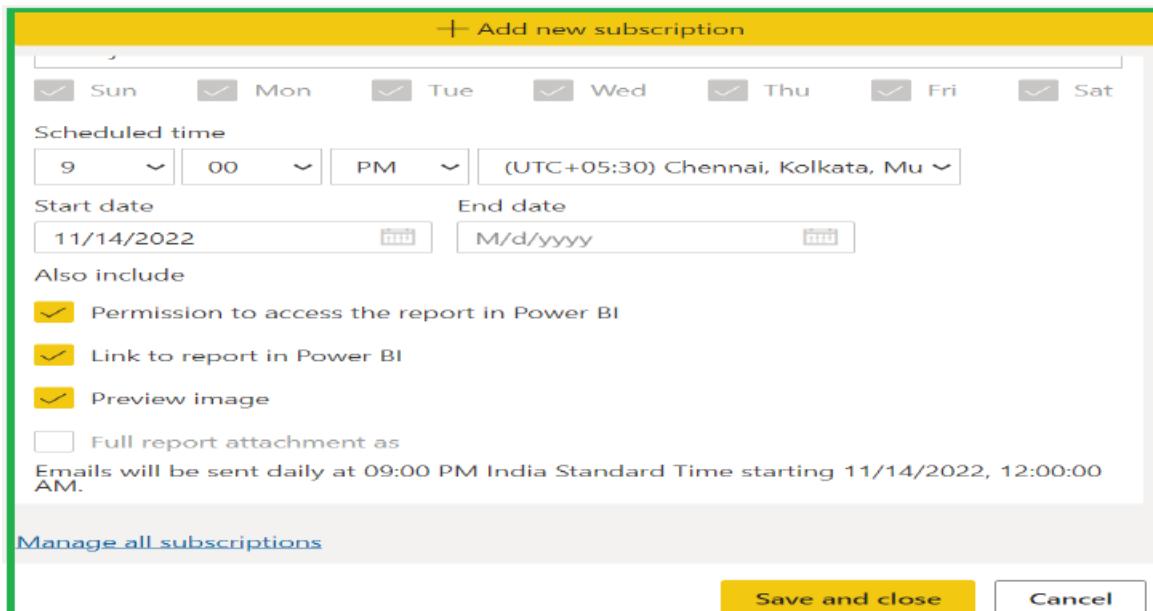
If you select, **hourly** as frequency, then for each and every hour it will send the report for the scheduled dates. And it will start at scheduled date.

If you select, **Daily** as frequency, then you can schedule the date. If you give start date and end date, report will be sent between these days. If you forget to give end date, the report will be sent until you stop the process.

If you select, **weekly** as frequency, then you can schedule to send a report on weekly basis.

If you select, **monthly** as frequency, then you can schedule to send a report on monthly basis.

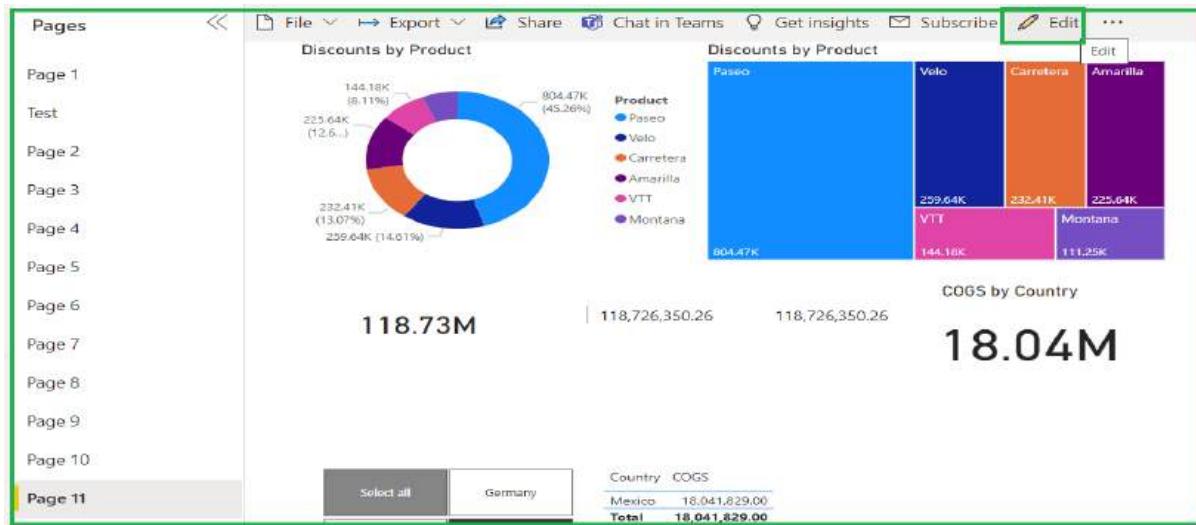
The mail will be sent to the users' inbox in outlook if you click on **save and close** option.



12)Edit: We can edit the reports but we don't edit in power bi service, because we can't change the **dax function** logic. But edit option will be useful for visual modifications.

If you want to change or modify the visual then we can use.

Power BI desktop options will not be available in Power BI service. For example, Transform data, DAX functions etc. That's why we can't edit the visual in logic level. We can change the columns, colors, fonts etc.



13) Dashboard: Multiple visuals we can put in single page is called as a dashboard.

A Power BI dashboard is a single page, often called a canvas that tells a story through visualizations. Because it's limited to one page, a well-designed dashboard contains only the highlights of that story. Readers can view related reports for the details.

Note: Dashboards are a feature of the Power BI service only. They're not available in Power BI Desktop. Although you can't create dashboards on mobile devices, you can **view and share** them there.

The visualizations you see on the dashboard are called tiles. You pin tiles to a dashboard from reports.

The visualizations on a dashboard originate from reports and each report is based on a dataset. One way to think of a dashboard is as an entryway to the underlying reports and datasets.

Selecting visualization takes you to the report (and dataset) that it's based on.

Dashboards are a wonderful way to monitor your business and see all of your most important metrics at a glance.

Note: Business users can't see the data by going page by page, instead of that they will put some important visuals in dashboard and see the dashboard. If there is any deviation in the visual of a dashboard, then they will directly click on that see the complete visual data.

If you want to put the visuals in a dashboard, you have to select the visual and click on the **pin** symbol on that visual.



You can select the visuals as many as you want and click on pin button. Then you have to give the existing dashboard name or new dashboard name. Then click on “Pin” button. Automatically that visual will be pinned to the dashboard.

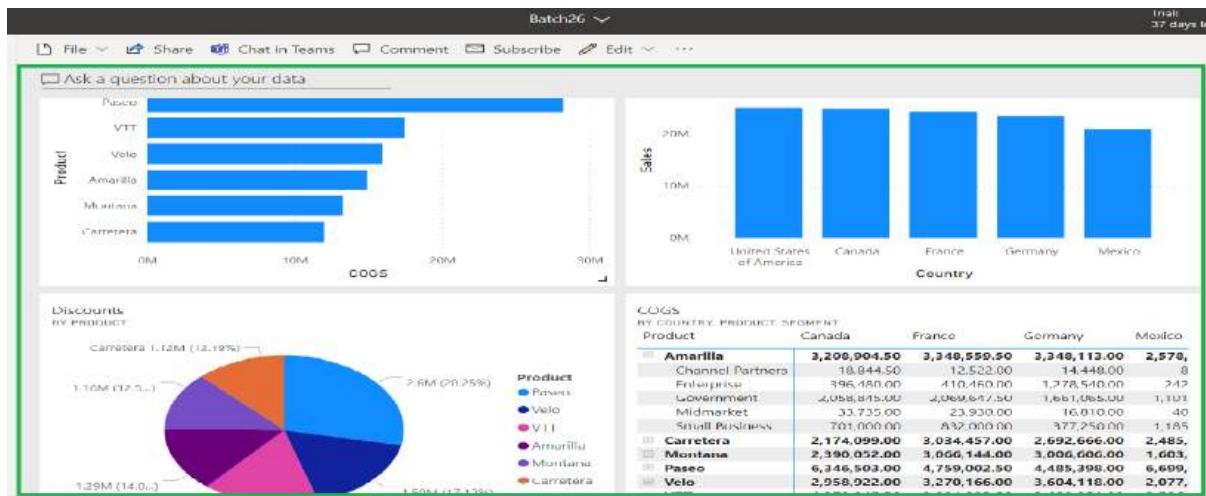
Whatever the visuals you want to pin from any page, those you can select individually and pin them separately to the dashboard.

Note: We can't put the slicer visual into the dashboard, because it will not have the pin option.

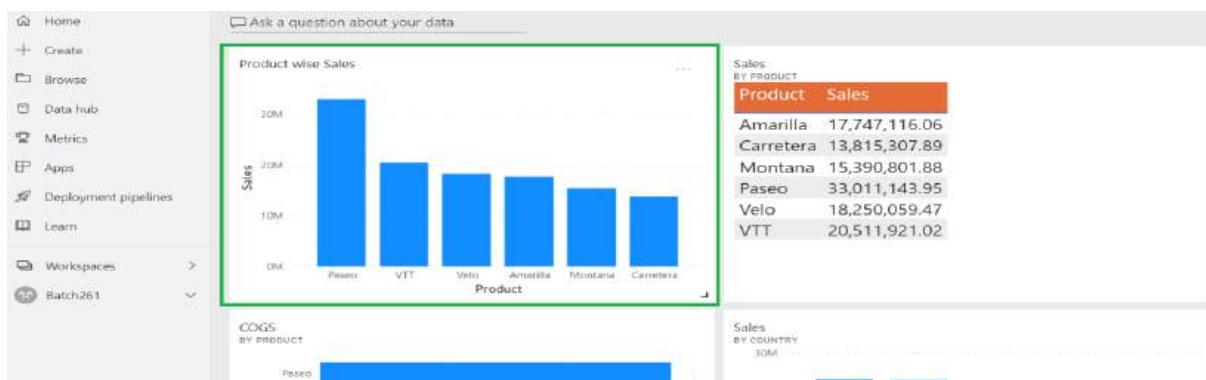
Pin: Pin is a single visual or tile.

If you want to see the dashboard, go to the particular workspace, and check in the **content** tab or **all** tabs. Click on that dashboard to see the pinned visuals.

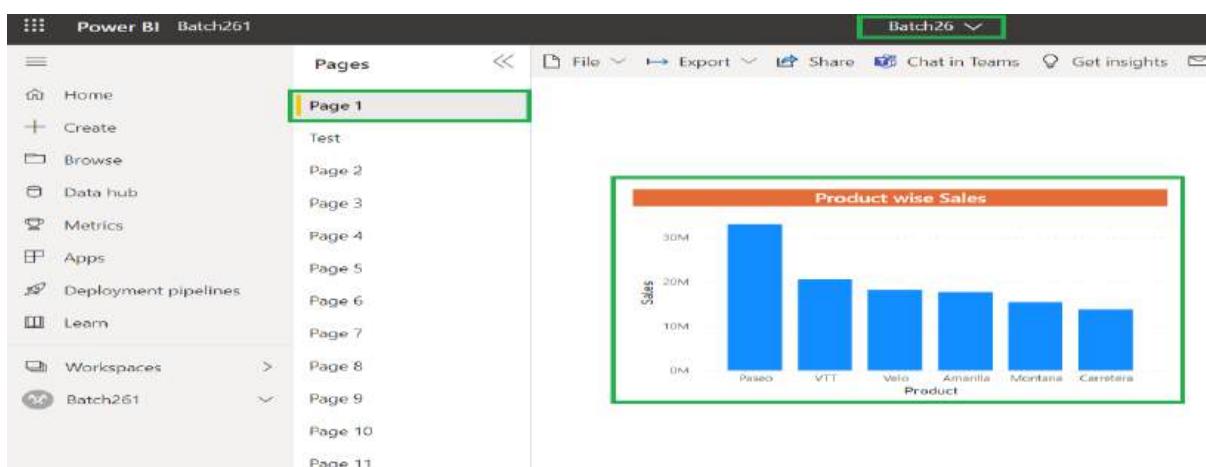
Name	Type	Owner	Refreshed	Next refresh	Endorsement
Batch261	Dashboard	Batch261	—	—	—
Batch26	Report	Batch261	10/12/22, 9:07:32 AM	—	—



If you want to see any visual data, then you just simply click on that visual. It will automatically takes you to the exact page of report corresponds to.



Suppose, I want to go to the product wise sales “**Stacked bar chart**”, then I just simply click on that visual. Then it will take me to the corresponding page of that visual in a report. Basically that visual is present in the page1. Then it shows the page1 in the report.



Datasets + Data flows: You can see some options in this section

Refresh: You can refresh the data source not only in power bi desktop; you can refresh it in power bi service as well. When you mouse-hover on any data source then you can see the refresh button.

The screenshot shows the Power BI service interface. At the top, there's a navigation bar with a user icon, the title 'Batch261', and options like '+ New', 'Create deployment pipeline', 'View', 'Filters', and a gear icon. Below the navigation is a filter bar with 'All' and 'Content' tabs, and a selected 'Datasets + dataflows' tab. A table below lists datasets, with one row highlighted. The columns are 'Name', 'Type', 'Owner', and 'Refreshed'. The highlighted row shows 'Batch26' as a Dataset owned by 'Batch261' last refreshed on '10/15/22, 9:07:32 AM'. The 'Refreshed' column contains a small warning icon.

If you want to refresh in power bi service, then it should require gateway connection.

Gateway: It is a bridge connection between source and power bi service. It is nothing but a link between source and service. Gateway not required for all the data sources. There are two types of gateways.

- ⇒ Personal Gateway
- ⇒ On premises Gateway

Personal Gateway: This is only for personal purpose only, we can't use personal gateway in organization. It will be useful for single user access only.

On premises Gateway: This will be used in organization level. With this, multiple users can access. That means multiple users can access a single gateway. Cost of gateway is high, that's why multiple users will use single gateway.

Cloud based related data sources doesn't require a gateway connection. For example **Azure, Sales force** etc.

Gateway requires for data sources like **SQL Server, Oracle, SharePoint**. Gateways require for these to refresh data sources in power bi service.

If you want to refresh data in power bi desktop, it will take time because it is completely local system. But power bi service is cloud based environment. So refreshing the data is very fast. If you want to do the refresh, gateway requires.

Suppose, different users will connect to different data sources with gateway, but organization will not provide different gateways for different data source. They will install **on premises** gateway in separate system and they will give permission to different users to access this gateway.

Once gateway is established, then we have to choose the "**Scheduled Refresh**" option. We have to check whether the gateway is enabled or not.

Go to "**Datasets + Dataflows**" option, then mouse-hover on any data set, then "**Scheduled Refresh**" option will be visible. Click on that.

The screenshot shows the Power BI dataset list. At the top, there's a search bar with 'Batch26' and navigation links like '+ New', 'Create deployment pipeline', 'View', 'Filters', 'Settings', and 'Access'. Below the search bar, there are tabs: 'All', 'Content', and 'Datasets + dataflows' (which is selected). A table lists datasets, with one row for 'Batch26'. The table columns are 'Name', 'Type', 'Owner', 'Refreshed', and 'Next refresh'. The 'Batch26' row has a 'Schedule refresh' button highlighted with a green box.

You will see the below screen after clicking on the “Scheduled Refresh” option.

The screenshot shows the 'Datasets' tab in the Power BI settings. It displays the 'Batch26' dataset settings. A yellow box highlights the 'Last refresh failed: 10/15/2022, 9:07:32 AM. Scheduled refresh has been disabled.' message. Below it, the 'Gateway connection' section is expanded, showing two entries: 'Personal Gateway' and 'satisf'. Both entries have red crossed-out icons next to them, indicating they are not reachable or correctly configured. Buttons for 'Apply' and 'Discard' are at the bottom.

Expand the **Gateway connection** option, you will see the below screen.

The screenshot shows the expanded 'Gateway connection' section. It includes instructions for using an On-premises or VNet data gateway. A toggle switch is set to 'On'. Below it is a table listing gateway connections:

Gateway	Department	Contact information	Status	Actions
Personal Gateway	DESKTOP-465NGMB		(✗) Not reachable	
satisf		noyey49538@wirese.c...	(✗) Not configured correctly	

Buttons for 'Apply' and 'Discard' are at the bottom.

Gateway connection:

If the gateway has an access given by the organization, then it will be enabled with tick mark. That particular gateway connection we have to select it and click on “Apply” button.

▲ Gateway connection

To use a data gateway, make sure the computer is online and the data source is added in [Manage Gateways](#). If you're using an On-premises data gateway (standard mode), please select the corresponding data sources and then click **Apply**.

Use an On-premises or VNet data gateway

On

Gateway	Department	Contact information	Status	Actions
Personal Gateway	DESKTOP-465NGMB		✗ Not reachable	
satish		noyey49538@wirese.c...	✗ Not configured correctly	

Apply **Discard**

Data source credentials: We have to provide the credentials for data sources to connect with that data source.

Suppose if you want to connect with SQL Server data base, then you have to give the credentials like Server Name, User name, Password and Authentication mode etc.

▲ Data source credentials

✗ Failed to test the connection to your data source. Please retry your credentials. [Learn more](#)

Combine Files	✗ Edit credentials Show in lineage view
Country Wise Sales.xlsx	✗ Edit credentials Show in lineage view
Financial Sample.xlsx	✗ Edit credentials Show in lineage view
Financial Sample.xlsx	✗ Edit credentials Show in lineage view
NSRInfo_Offline_Batch1-DESKTOP-DL1HHT0	✗ Edit credentials Show in lineage view

Parameters: If you want to create a parameter for the environment to the SQL Server or anything, then you can create and pass it here.

▲ Parameters

Country

Canada

Environment

DESKTOP-DL1HHT0

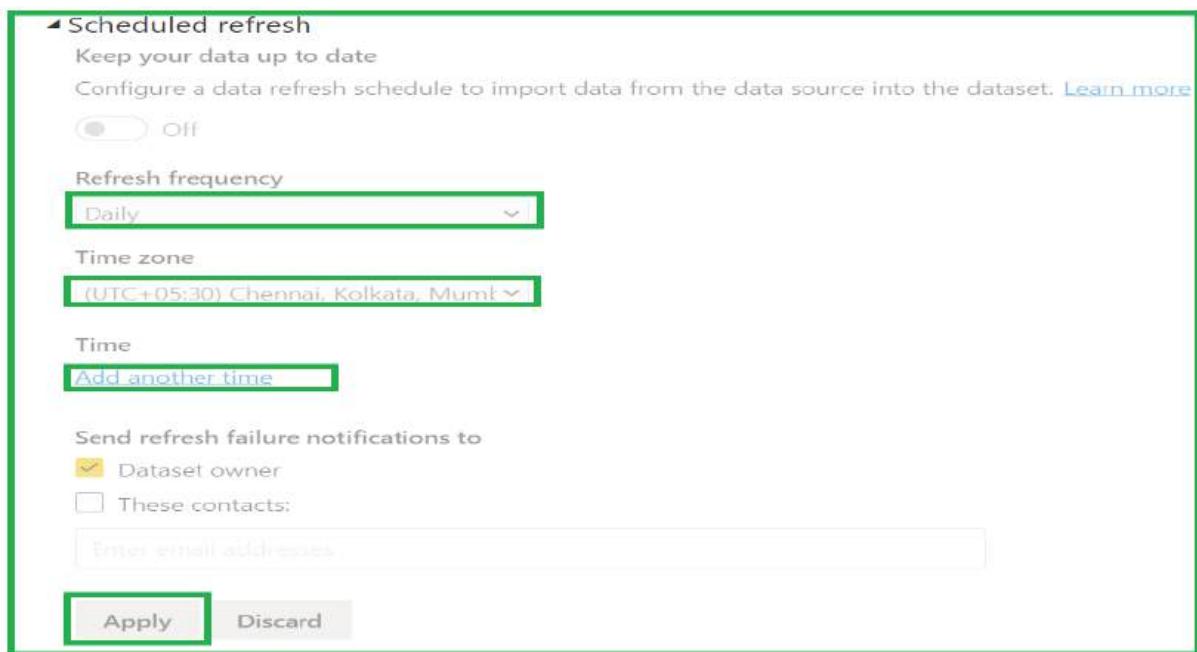
Scheduled Refresh: You can schedule the refresh in this section, we already know that we can refresh the data source 8 times a day in **Pro license**, but in **premium license** we can refresh 48 times a day.

Refresh frequency you have to choose like daily, weekly or monthly and you have to select the required time zone. You can use the “**Add another time**” option.

Add another time option will be enabled 8 times for the pro user, 48 times for premium user. You can select this option less than or equal to 8 or 48 based on the user.

Note: Whatever the time you have scheduled in the scheduled refresh option, on that time only refresh will be happened.

Suppose if you are a premium user you can schedule the refresh for every 30 minutes.



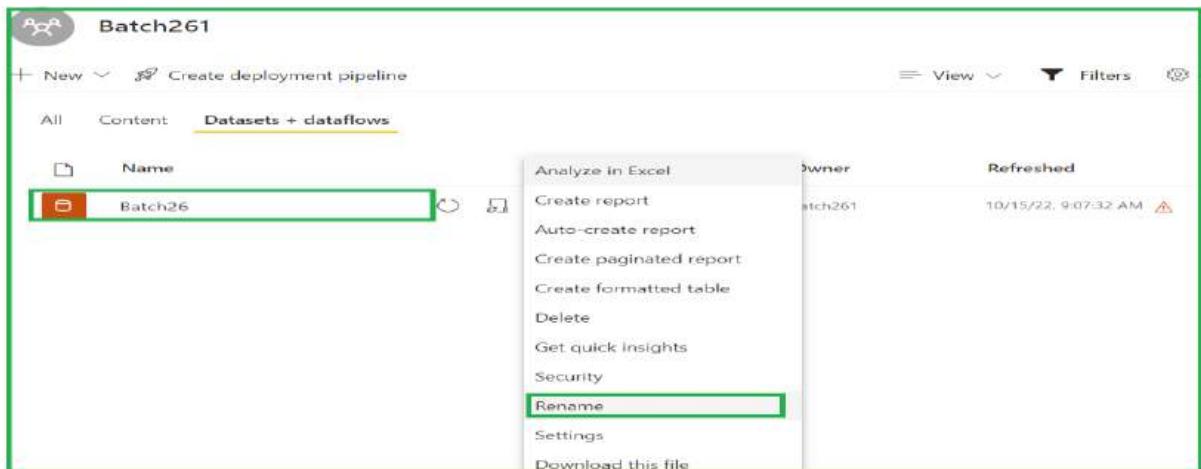
If you don't want to wait for the scheduled time to refresh; then you can click on the “**Refresh now**” option. This will be worked if you set the all options like gateway, time, timezone in scheduled refresh.

You can use “**Refresh now**” N number of times if you don't require waiting for scheduled refresh.

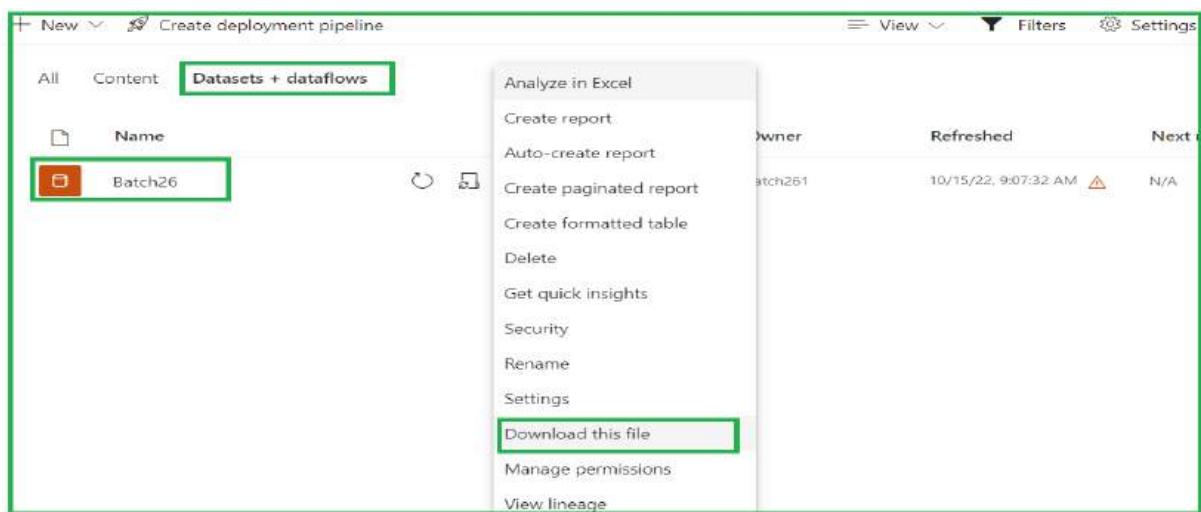
The screenshot shows a list of datasets. One dataset, 'Batch26', is selected. The 'Datasets + dataflows' tab is active. Below the table, there is a button labeled 'Refresh now'.

All	Name	Type	Owner	Refreshed
	Batch26		Dataset	Batch261 10/15/22, 9:07:32 AM

Rename: If you want to rename a data set, you can use this option.



Download this file: If you want to download a Power BI file, then can also download in “Datasets + Dataflows” option. Click on ellipse symbol and click on “Download this file” option.



Data flows: If you want to create a data set in power bi service then you can use “Data flows” option.

To create a data set, first you need to click on “New” option. Click on “Dataflow” option. Then click on “Add new tables” option.

 Batch261

+ New  Create deployment pipeline

All Content **Datasets + dataflows**

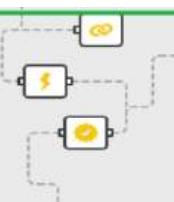
	Name	Type	Owner
	Batch26	Dataset	Batch261

 Batch261

+ New  Create deployment pipeline   

-  Report Visualize your data
-  Paginated report Build a paginated report
-  Scorecard Track related metrics together
-  Dashboard Build a single-page data story.
-  Dataset Create a dataset to use in a report
-  Dataflow Prep, clean, and transform data **Selected**
-  Streaming dataset Build visuals from real-time data
-  Upload a file Open a .pbix, .rdl, .xlsx, or .csv in Power BI

Type	Owner	Refreshed
Dataset	Batch261	10/15/22, 9:07:32 AM 



Start creating your dataflow

Define new tables
Choose a data source to define the tables for your dataflow. You can map your data to standard Common Data Model tables, or define custom tables instead.
[Learn more](#)

Add new tables

Link tables from other dataflows
Linking to tables from other dataflows reduces duplication and helps maintain consistency across your organization.
[Learn more](#)

Add linked tables

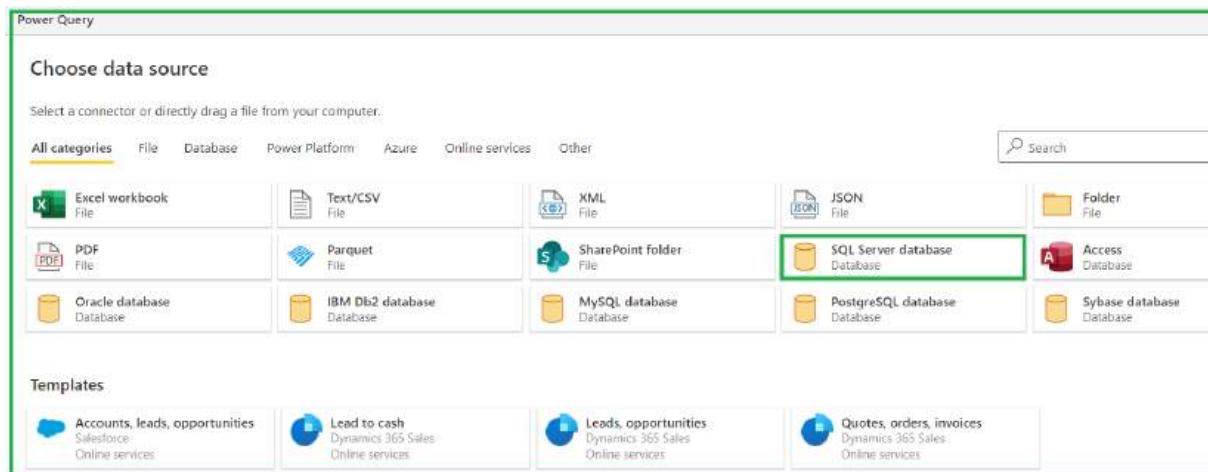
Import Model
Choose a dataflow model to import into your workspace.
[Learn more](#)

Import model

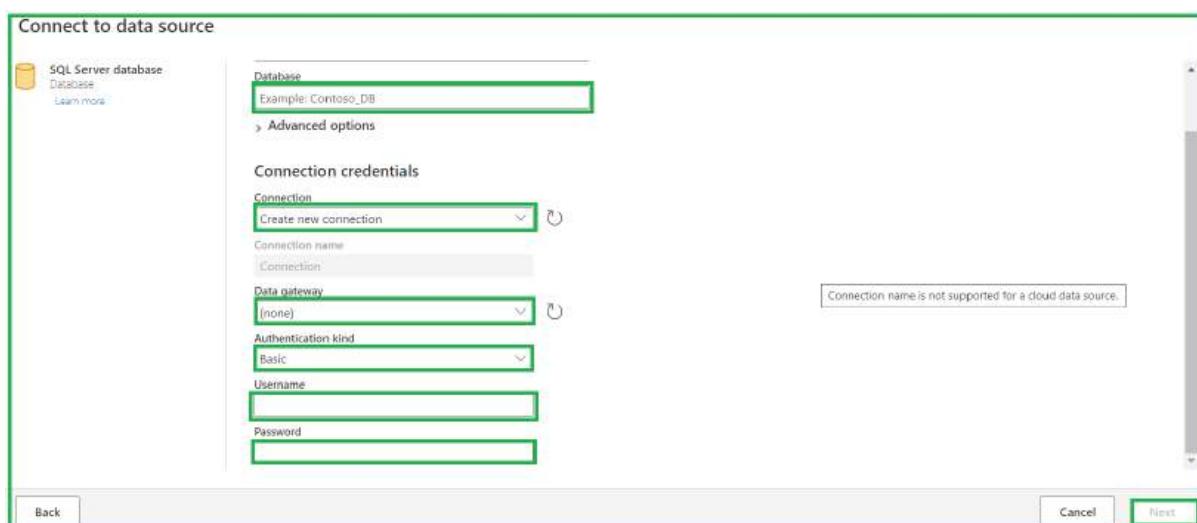
Attach a Common Data Model folder (preview)
Attach a Common Data Model folder from your Azure Data Lake Storage Gen2 account to a new dataflow, so you can use it in Power BI.
[Learn more](#)

Create and attach

How you use “**Get Data**” option in power bi desktop to create data set, in the same way you can select the data source and create a data set. Suppose I am selecting “**SQL Server**” data base to create a data set.



You have to give the credentials like data base name, connection, gateway, authentication kind, user name and password.



Note: Whatever the data sets you have created in “**Dataflows**” option, those are called as live connections.

Note: Data sets loading in power bi desktop will take much time where as in power bi service data set loading is very fast because it is cloud based environment. Millions of records can be loaded in less time in power bi service.

Difference between data set in power bi desktop and power bi service.

Data set we can create in power bi service, data flow we can create in power bi desktop.

When you create a data set in power bi desktop, it will take lot of time but if you want to create a same data set in power bi service it will take less time because of cloud based environment.

Data sets created by the **data flows** options are called as live connections.

Note: If you want to create a report using the data set created by the data flows, then you have to call that data set from the power bi desktop.

Go to “Power BI Desktop” -> Click on **Home** ribbon -> Click on **Get Data** option -> Select “**Power BI dataflows**” option.

You can see all the data flows created in power bi service. Select the “**Data set**” from them. Automatically data set will be loaded into the power bi desktop under field’s pane.

The screenshot shows the Power BI Desktop interface. The top navigation bar has tabs: File, Home, Insert, Modeling, View, Help. The Home tab is selected. Below the ribbon, there's a 'Common data sources' section with a tree view. Under 'Power BI datasets', the 'Power BI dataflows' option is highlighted with a green box. To the right of the tree view is a data grid showing sales and profit data for Canada and France across various products like Channel Partners, Government, Manufacturer, Small Business, and Commer. The data grid has columns for Country, Product, Sales, Profit, and Group. The bottom right corner of the data grid also has a green box. On the far right, there's a 'Visualizations' pane with various chart and report icons.

App: To share the report to the users, you can use “**App**” option. You can share the reports via “**Share**” option as well.

Difference between Share and App

Using “Share” option you can share a single report to the multiple users at a time. But using “App” option you can share the multiple reports to the multiple users at a time.

If you want to create an app, you have to be present in Workspace only. If you are there in workspace then you can see “**Create App**” button on the top right corner of the screen.

Navigation: Go to **Workspaces** -> Select “**Batch261**” workspace -> Click on “**Create App**” button.

The screenshot shows the Microsoft Power BI workspace interface. The left sidebar has navigation items: Home, Create, Browse, Data hub, Metrics, Apps, Deployment pipelines, Learn, and Workspaces. The 'Batch261' workspace is selected. In the center, there's a table listing items: Name, Type, Owner, Refreshed, Next refresh, Endorsement, Sensitivity, and Include in app. Two items are listed: 'Batch26' (Dashboard, Owner: Batch261) and 'Batch26' (Report, Owner: Batch261). At the top right, there's a 'Create app' button highlighted with a yellow box. The top right also has 'View', 'Filters', 'Settings', 'Access', and 'Search' buttons.

After that you have to provide “**App Name**”, “**Description**” and you can **upload the image** for the app and **color** also in “**Setup**” tab.

① Setup ② Content ③ Audience

App name*

Batch261

Description*

Batch261

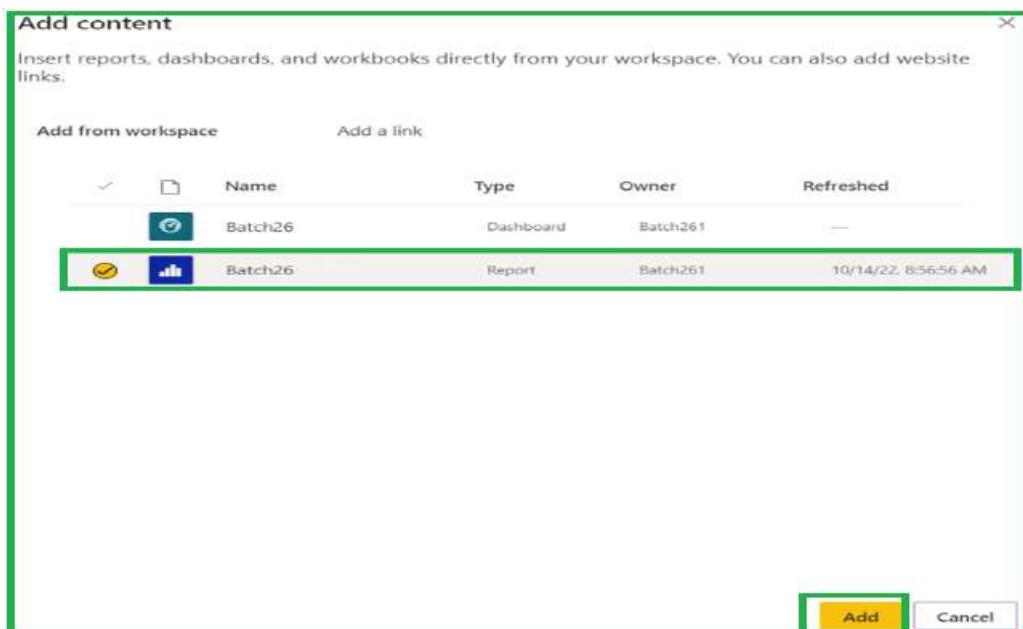
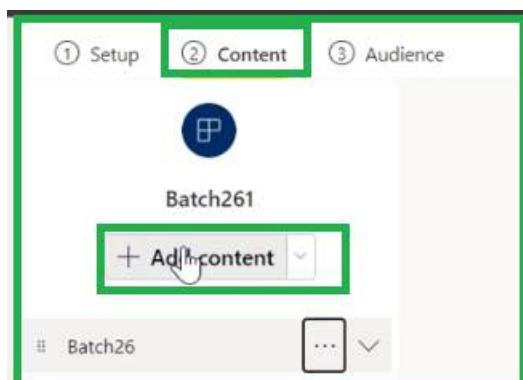
App logo

App theme color

Contact Information

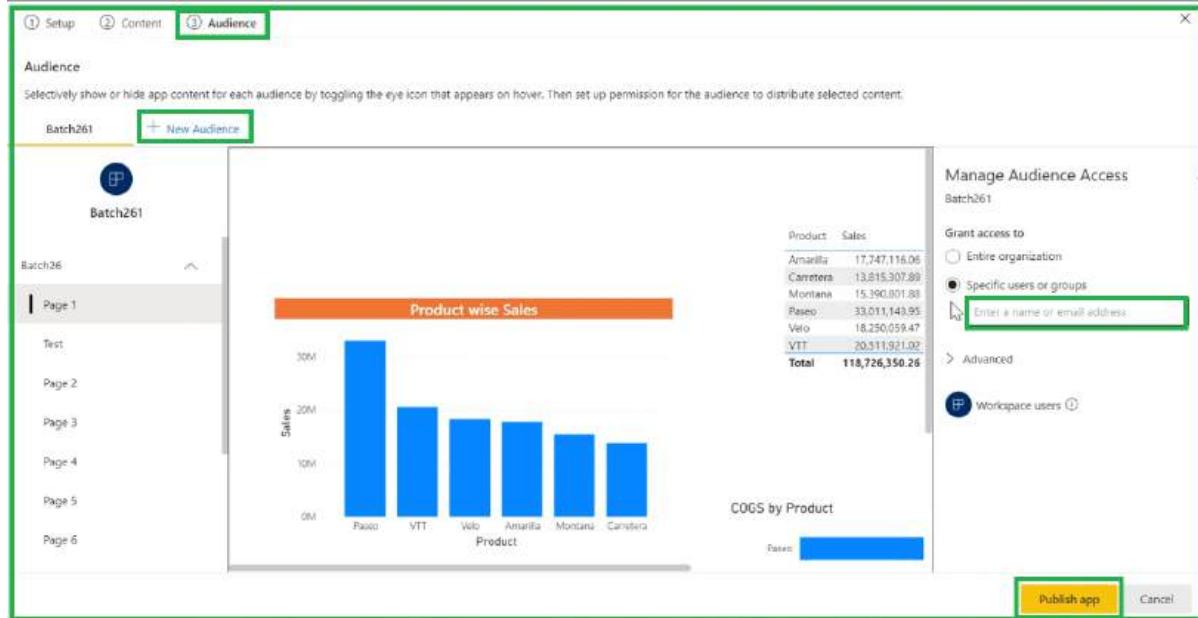
Show app publisher

If you want to select the reports that you want to share to the multiple users; then you have to go to the **Content** tab and click on “**Add Content**” in drop down menu. Then select the required reports and click on “**Add**” button.



Once you added the reports, then you have to add the audience to share the report.

Navigation: Go to “Audience” option. Mention the mail ids of the audience to whom you want to share the report in the “Specific users or groups” option and click on “Publish app” option.



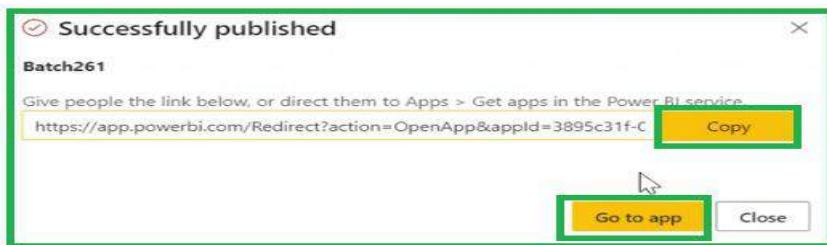
Note: “Specific users or groups” option will be available for “Member” role and “Admin” role users, but contributor role this option will not be available.



Once you click on “Publish app” option again it will ask to click on “Publish”. Now click on that.



Once you click on “Publish” option, a new link will be created. You can copy that link and paste it on new browser window. Then you can see the published report.



Now the final window looks like below in the new browser window. Like this you can share multiple reports through the “App” option. All the reports will be available in the same link.

