

# **POWER BI**

**2024**

## **NEW MATERIAL**

**ANALYTICS BENCHMARK TRAININGS**

---

**SUNIL SIR**

***MANOJ ENTERPRISES & XEROX***

**All soft ware institute materials, spiral-binding,**

**Printouts & stationery also available ..,**

**CONTACT:9542556141**

**Add: Plot No.40, Gayatri Nagar, Behind HUDA, mithrivannam, HYD.**



<b>Power BI - A Complete Introduction.....</b>	<b>01</b>
<b>Data Transformation with Power Query / Query Editor.....</b>	<b>08</b>
<b>Inbuilt Column Transformations.....</b>	<b>26</b>
<b>Remove Columns / Remove Other Columns</b>	<b>    Name / Rename a Column</b>
<b>Reorder Columns or Sort Columns</b>	<b>    Add Column / Custom Column</b>
<b>Split Columns</b>	<b>    Merge Columns</b>
<b>Pivot, Unpivot Columns</b>	<b>    Transpose Columns</b>
<b>Power BI Modeling.....</b>	<b>55</b>
<b>Introduction to DAX.....</b>	<b>67</b>
<b>DAX Functions.....</b>	<b>74</b>
<b>DAX Logical Functions.....</b>	<b>77</b>
<b>Time Intelligence.....</b>	<b>95</b>
<b>Power BI Visual Interactions (Ad-hoc filtering and highlighting)</b>	<b>105</b>
<b>Grouping and Binning.....</b>	<b>116</b>
<b>Functions in DAX / DAX Functions Categories.....</b>	<b>74</b>
<b>Filters, Drillthrough and Visual Interactions.....</b>	<b>105</b>
<b>Grouping and Binning.....</b>	<b>116</b>
<b>Sorting Data in Visuals in Power BI Desktop.....</b>	<b>124</b>
<b>Hierarchies and Drill-Down in Power BI.....</b>	<b>130</b>
<b>Visualizing Data.....</b>	<b>137</b>
<b>Visualizing Trend Data.....</b>	<b>148</b>
<b>Power BI Service Introduction.....</b>	<b>162</b>
<b>Access on-premises data from Power BI Service - Data Gateway.....</b>	<b>173</b>
<b>App.....</b>	<b>191</b>
<b>DATA WAREHOUSING CONCEPTS.....</b>	<b>205</b>
<b>Bookmarks and Selection Pane.....</b>	<b>229</b>
<b>MAP .....</b>	<b>257</b>
<b>Tooltip Page.....</b>	<b>260</b>

## Power BI - A Complete Introduction

### Power BI

Power BI is a collection of Tools or Software's provided by Microsoft Corporation for performing Business Intelligence Activities.

Power BI is a Self Service, Cloud Based Reporting Software (OR) Data Visualization Software (OR) Data Analytics Software (OR) Business Intelligence Software.

To Perform Business Intelligence there are "N" numbers of Business Intelligence Tools/Software's both Traditional and Self serviced in the Market from Different Vendors, below are the popular among them and their service provider.

BI Tool	Service Provider
Power BI	Microsoft
Tableau	Tableau Software
QlikView	Qlik
Qlik Sence	Qlik
MSBI	Microsoft
Microstrategy	Microstrategy Corporation

We have two types of BI tools in market

1. Traditional BI
2. Self-Serviced BI

In recent years, we observe that there has been an evolutionary shift from legacy, on premise traditional Business Intelligence (BI) solutions to cloud-based, self-serviced BI.

### Traditional BI

Traditional BI has been around for years and often requires a high-level technical skill to implement, administer and maintain the solution. Another characteristic of traditional BI is that it requires the solution to be housed on premise. For growing companies, this is a costly and ineffective solution. Additionally, since traditional BI is typically a technical product, it's difficult for business users to have access to it.

### Fundamental characteristics of Traditional BI

- ✓ Traditional Tools offers a broad range of features which allow companies to cover a wide spectrum of reporting types and an array of use cases.

- ✓ Requires a high level of technical expertise, users rely significantly on IT to perform even the most basic functions like building reports. As a result, user adoption rates may suffer.
- ✓ Requires IT to have SQL query skills or learn a proprietary query language in order to implement- which drives up cost and adds to the time required to deploy the solution.

## **Self Service BI**

Self-service BI is a form of Business Intelligence in which end users are empowered to independently satisfy their own information needs. With self-service BI, non-technical professionals can generate their own reports, run their own queries, and conduct their own analyses, without the assistance of IT staff.

### **Fundamental characteristics of Self-Service BI**

- ✓ Business users are able to access the real-time data they want and quickly generate results without the need for technical expertise. Often, no coding skills are required.
- ✓ Self Service BI's upfront costs, total cost of ownership (TCO) and total cost of change (TCC) are significantly less than Traditional BI's costs.
- ✓ User adoption is typically greater with Self Service BI than with Traditional BI because it's easier for the non-technical business user to understand and leverage.

## **Cloud Based**

- ✓ Cloud-based is a term that refers to applications, services or resources made available to users on demand via the Internet from a cloud computing provider's servers.
- ✓ With the help of Power BI Service you can analyze your data anywhere from the world with the help of internet.

## **On-Premise BI**

- ✓ On-Premises BI software which is installed locally, on a company's own computers and servers.
- ✓ Power BI Report Server is a solution that customers deploy or install on their own premises for publishing, sharing and managing Reports.
- ✓ Reports shared from Power BI Report Server can be accessed within the Network only.

## **Reporting**

- ✓ Presenting the Data in a Structured Format we call it as Reporting. With the help of Power BI we can present the Data in a Structured Format hence we called Power BI Software as a Reporting Software.

## Data Visualization

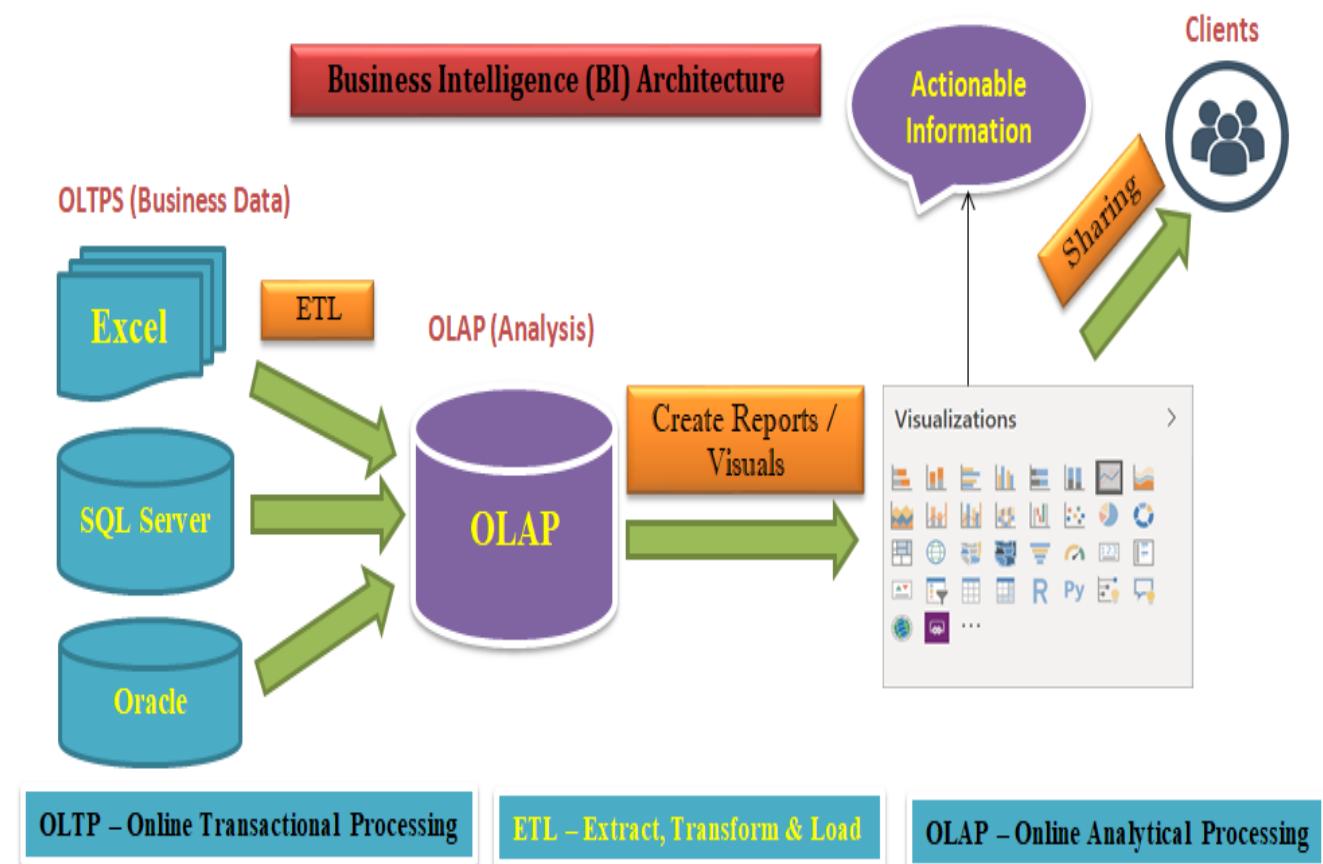
- ✓ Presenting the Data in the form Graphs, Charts, Maps etc., we call it as Data Visualization (OR) Graphical Representation of Data we call as Data Visualization. With Power BI Software we Can Present the Data in a Graphical Format hence we call Power BI Software as Data Visualization Software.

## Data Analytics

- ✓ Performing the Analysis on the Data we call it as Data Analytics. With the Help of Power BI Software we can perform Analysis on huge Volumes of Data hence we call Power BI Software as Data Analytics Software.

## Business Intelligence

- ✓ Business Intelligence is a Process which Converts Business Data into Actionable Information.
- ✓ Business Intelligence Process helps Business Managers to make more informed business decisions.



When you go as any BI Developer Clients will provide the OLTP(S) which Contains Business Data and ask you to perform the Analysis. It is not recommended to perform the Analysis Directly on top of OLTP System. As a BI Developer First we need to create an OLAP System where we need to bring data that is required for Analysis. To move the Data from OLTP Systems to OLAP System we need to use ETL Tools. Once Data is there in the OLAP we need to Start Performing the Analysis. Create the Reports and Visuals on Top of OLAP and Finally Share them with Clients.

**To Perform this Business Intelligence Activities Microsoft Provided Power BI Software.**

Power BI is the not the Name of Single Software. It's the Name for Suite of Software's which we used for performing Business Intelligence (BI) Activities.

### **Power BI Software's / Tools / Products**

- Power BI Desktop
  - Power Query
  - Power Pivot
  - Power View
- Power BI Service
- Power BI Report Server
- Power BI Mobile

### **Power BI Desktop**

- ✓ Power BI Desktop is a tool to Connect to, clean, model, and visualize your data.
- ✓ With Power BI Desktop, you can connect to different Data Sources to Extract the Data, Transform the Data if required, Model the Data (Data Modeling) and visualize the Data in different ways.
- ✓ Power BI Desktop is the combination of below software's
  - Power Query
  - Power Pivot
  - Power View

## **Power Query**

- ✓ Power Query is used for Data Extraction, Transformation and Loading. It's an ETL Software in Power BI.
- ✓ With the help of Power Query, we will connect to the different Data Sources to Extract the Data, Transform the Data and then we will load the Data into Power Pivot.
- ✓ The Power Query Software comes with a graphical tool "Power Query Editor" and a formula language M (Mashup) Language to Transform the Data.
- ✓ With the help of predefined functions in the graphical tool we can transform the data in Power Query. However, Power Query can be programmed to create custom functions. This gives you seemingly unlimited potential to transform your data in just about any way possible.
- ✓ The formula language we used in Power Query is the M (Mashup) language to create the custom functions.
- ✓ Power Query can load the result set into Power Pivot model.
- ✓ Power Query not only makes all these tasks easier, but it also records your steps.

## **Power Pivot**

- ✓ Power Pivot is an In-Memory Columnar Database where we store the Data that is required for Analysis Purpose. Power Pivot is used for preparing an OLAP / Dataset.
- ✓ Power Pivot is the place where we place the transformed data that is loaded by Power Query for Data Modeling.
- ✓ Power Pivot works on xVelocity In-Memory based tabular engine. Current Name for Power Pivot "In Memory" is xVelocity previously they used to call as Vertipaq Engine.
- ✓ The In-Memory engine gives Power Pivot super-fast response time and the modeling engine would provide you a great place to build relationships through Entities, build your Star Schema, Create New Columns, New Measures/Quick Measures and New Tables, and so on.
- ✓ To Enhance the Data Model Power Pivot uses Data Analysis eXpression language (DAX) for building New Columns, New Measures/Quick Measures & New Tables.
- ✓ DAX is a powerful functional language which contains multiple functions that are helpful to perform the Data Analysis extensively as per the Client needs by creating new information Like New Columns, New Measures/Quick Measures & New Tables.

## **Power View**

- ✓ Power View is used for Data Visualization.
- ✓ With Power View you can create interactive charts, graphs, maps, and other visuals that bring your data to life.

## Power BI Desktop Process Flow



- ✓ Power BI Desktop as mentioned above is an editor for three components → Power Query, Power Pivot, and Power View.
- ✓ Power Query connects to data sources and mash up the data with a formula language, the result set of Power Query will be loaded into a tabular model which is Power Pivot.
- ✓ Power Pivot can set the relationships and allow you to create New Columns, New Measures / Quick Measures and New Tables using DAX Language and set the Data Model as you want.
- ✓ Then Power View connects to the Data Model and Visualizes the data with different charts and visualization elements.
- ✓ Power BI Desktop has everything in one editor, and this makes it an easy to use tool.
- ✓ You can solve very complex challenges with Power BI Desktop only because of its underlying components.

## **Power BI Service**

- ✓ Power BI Service is cloud based solution which is managed by Microsoft Corporation for publishing, sharing and managing Reports and Dashboards.
- ✓ Reports or Dashboards shared from Power BI Service can be accessed anywhere from the world.
- ✓ You can view the Reports or Dashboards that are shared with you with the help of web browsers or Mobile Application (Power BI Mobile).

## **Power BI Report Server**

- ✓ Power BI Report Server is a solution that customers deploy or install on their own premises for publishing, sharing and managing Reports.
- ✓ Reports shared from Power BI Report Server can be accessed within the Network only.
- ✓ You can view the Reports that are shared with you with the help of web browsers or Mobile Application (Power BI Mobile).

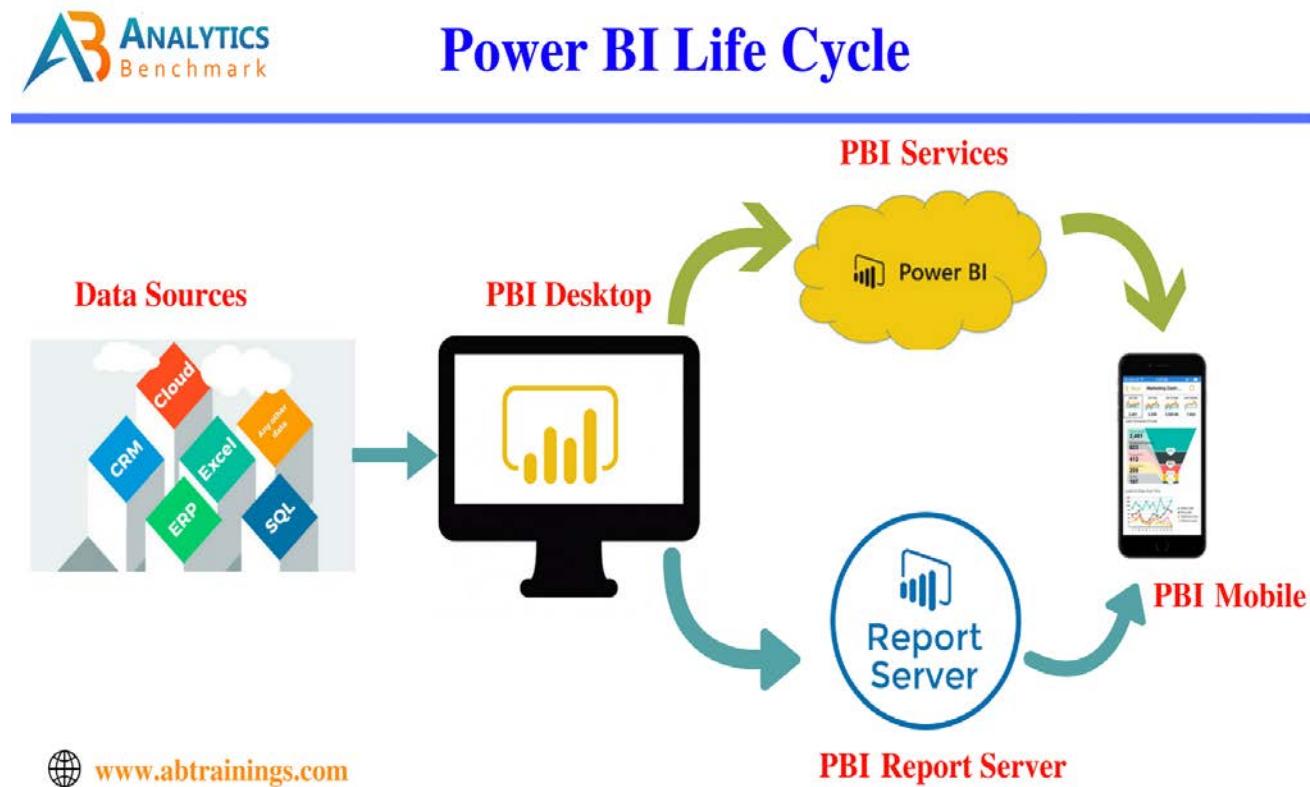
## **Power BI Mobile**

- ✓ Power BI Mobile is a mobile application which is used to see the Reports and dashboards shared with us by logging with our server credentials.
- ✓ Power BI Mobile App is available for Android, Apple, and Windows Phone, simply download it from Google Play (Android), or App Store (Apple), or Windows Store (Windows Phone) and install.
- ✓ After the installation login with your server credentials (username and password), and you'll see Reports and Dashboards shared with you.

## The Flow of Work in Power BI / Power BI Architecture

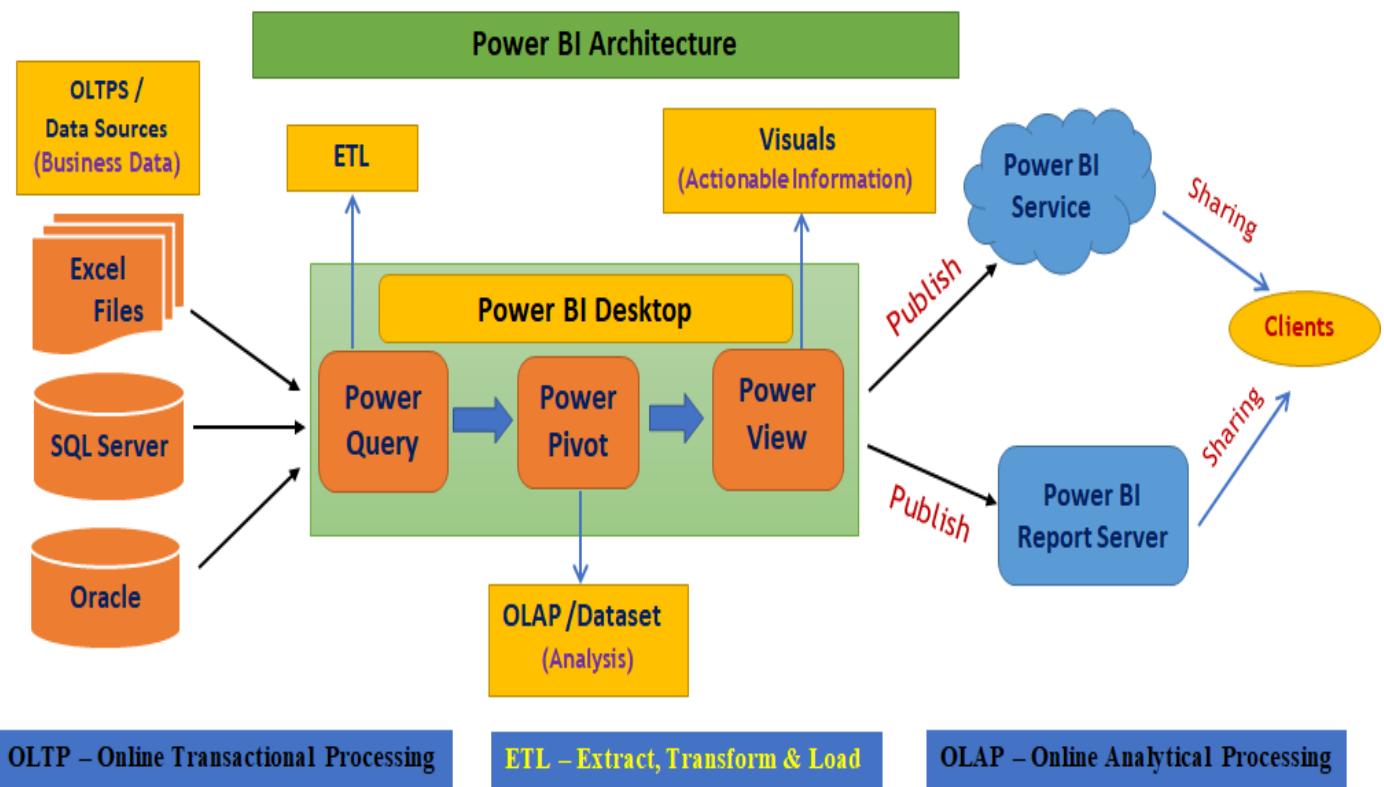
A common flow of work in Power BI is

- ✓ Bring data into Power BI Desktop, and create a report.
- ✓ Publish to the Power BI service or Power BI Report Server, where you create new visualizations or build dashboards.
- ✓ Share your dashboards with others, especially people who are on the go.
- ✓ View and interact with shared dashboards and reports in Power BI Mobile apps (Windows phones and tablets, as well as for IOS and Android devices).



## Power BI Reporting life cycle or BI Life Cycle

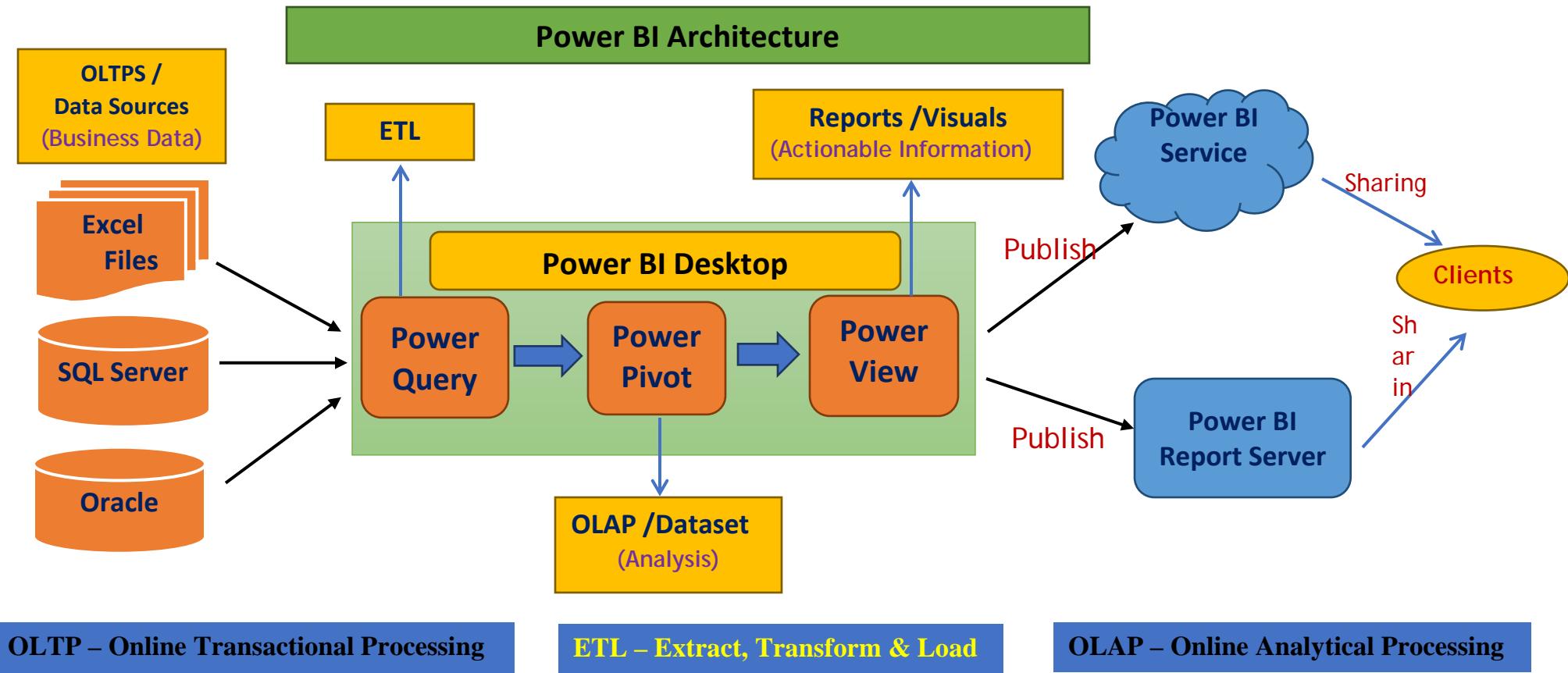
- ✓ Preparing the data and Loading into Reporting Tool → Power Query
- ✓ Data Modeling and defining Metrics → Power Pivot
- ✓ Report generation → Power View
- ✓ Creating Dashboards, Sharing & Admin Activities → Power BI Service or Power BI Report Server
- ✓ Viewing Reports & Dashboards → Mobiles and Web Browsers.



## A Brief History of Power BI

The tools in the Power BI are not new into the market. Let us see the history of Power BI.

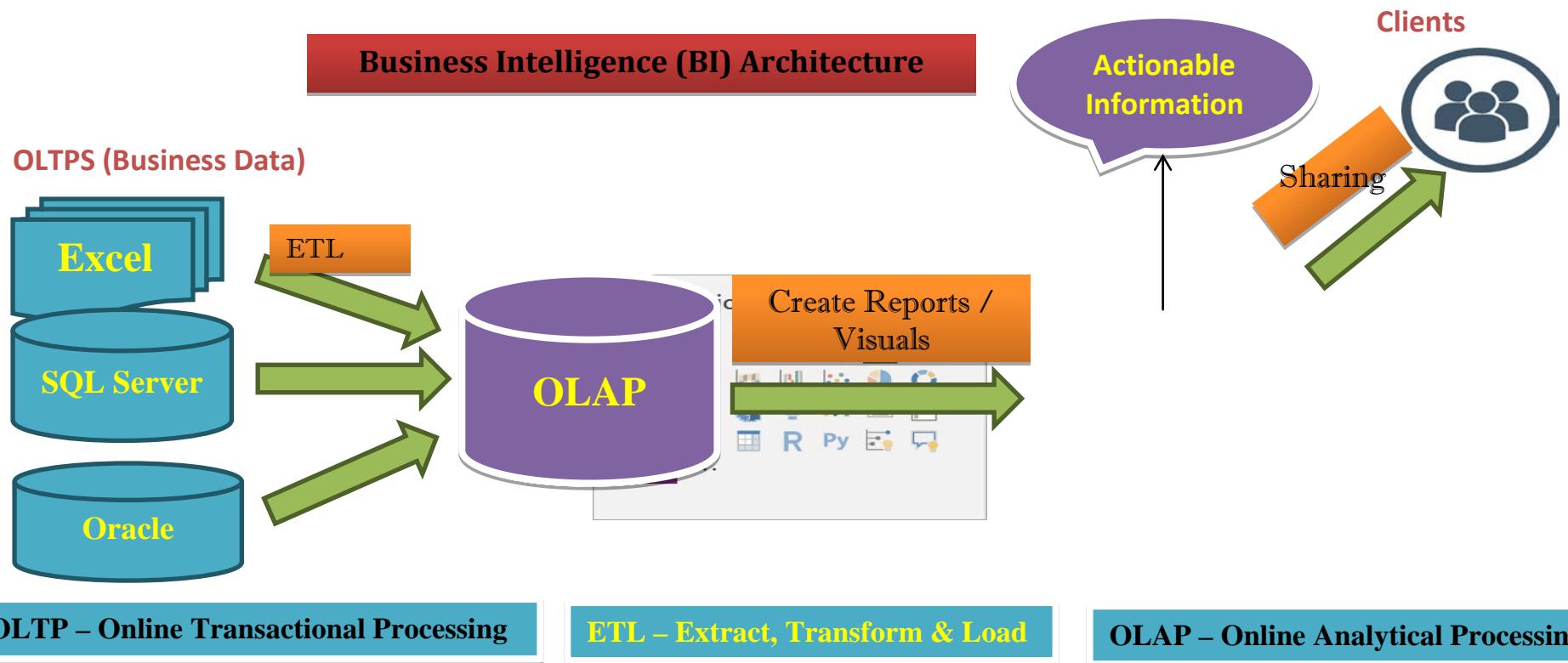
- ✓ Power Query is a free add-in in Excel 2010 and 2013 and it is inbuilt in Excel 2016.
- ✓ Power Pivot is a free add-in in Excel 2010 and 2013 and it is inbuilt in Excel 2016.
- ✓ Power View is a free add-in in Excel 2013 and it is inbuilt in Excel 2016.
- ✓ Power BI Service is released on Jan 2015.
- ✓ Microsoft combined Power Query, Power Pivot and Power View as Power BI Desktop and released on July 2015.
- ✓ Power BI Report Server was released on June 2017.



OLTP – Online Transactional Processing

ETL – Extract, Transform & Load

OLAP – Online Analytical Processing



## Data Transformation with Power Query / Query Editor

### Data Transformation

Data transformation is the process of converting data or information from one format to another, usually from the format of a source system into the required format of a new destination system.

### Data Transformation - Why?

When existing Business model is hard to understand we use power query to Shape or Transform the data, and build a model that will be easily understandable for a Report User.

If existing Business model contains too many tables and many relationships between tables makes a reporting query very slow and not efficient. Here we use Power Query to Shape and Transform the data to build a star or snow flake schema by creating dimension tables and fact table, which is more comfortable for report development.

Transactional databases are not best option for reporting purpose because

- ✓ The model is hard to understand for a Report User.
- ✓ Too many tables and many relationships between tables makes a reporting query (that might use 20 of these tables at once) very slow and not efficient.
- ✓ Also we don't need all the transactional data to be loaded into Reporting Tools we just load whatever data we need for reports into our reporting tools.

### Shape or Transform Data using Power Query

With Power BI Desktop or Query Editor or Power Query, you can connect different types of data sources, and then shape the data to meet your reporting needs.

In Power Query or Query Editor we will transform or shape the data using built-in GUI transformations in the ribbon or using M language code.

### Benefits of Data Transformation

Data transformation ensures that data that enters your enterprise is usable and manageable.

It facilitates cost-efficient storage, ease of analysis for greater business intelligence, and operational efficiency.

On the flip side, storing data that has not been transformed wastes resources and creates the possibility of compliance risk because the data cannot be managed under the organization's data governance rules.

## Overview of Power Query / Query Editor

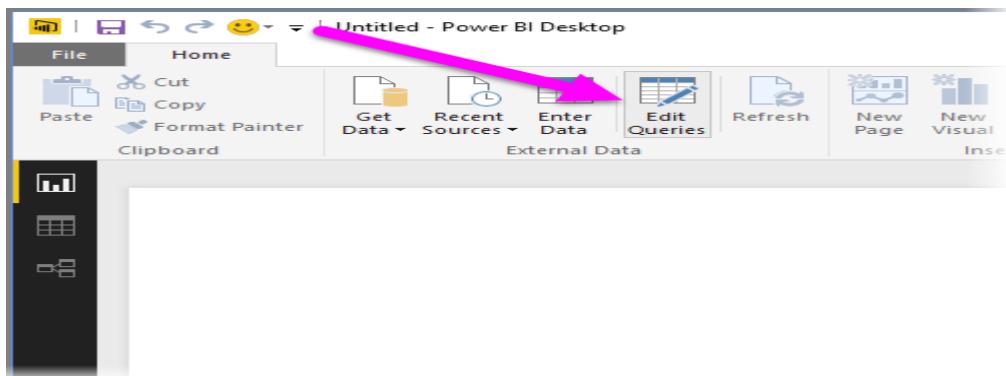
- ✓ Power Query is a Data Extraction, Transformation and Loading Engine.
- ✓ The Engine comes with a Graphical Tool and a Formula Language (M Language).
- ✓ Power Query can connect to set of data sources and read data from them for data preparation.
- ✓ Once connected to any data source, then Queries (one for each table, or entity) are listed and available for selection, viewing, and shaping.
- ✓ The Graphical Tool has list of Transformations that can be applied on a data set or Queries, and it also supports different data sources.
- ✓ Power Query graphical interface is so easy to work with that even business analyst or a power user can work with it, on the other hand Power Query M language is so powerful that can be used for complex real world challenges of data transformations.
- ✓ However, the Power Query formula language (M Language) is much more powerful than the GUI. Actually there are some features in Power Query engine that not yet has been implemented through GUI, but they are available through M Language.
- ✓ Power Query can load the result set into Power Pivot for data modeling.
- ✓ M is the formula language behind the scenes of Power Query. Everything you do in the Query Editor will be translated to an M script. M contains full list of functions that you can use. So the powerful side of Power Query is actually M. M is a functional language and it has a simple structure.
- ✓ Every data preparation steps or applied steps on Queries will be recorded and displayed in Query Editor under Applied Steps Section.

## Query Editor User Interface

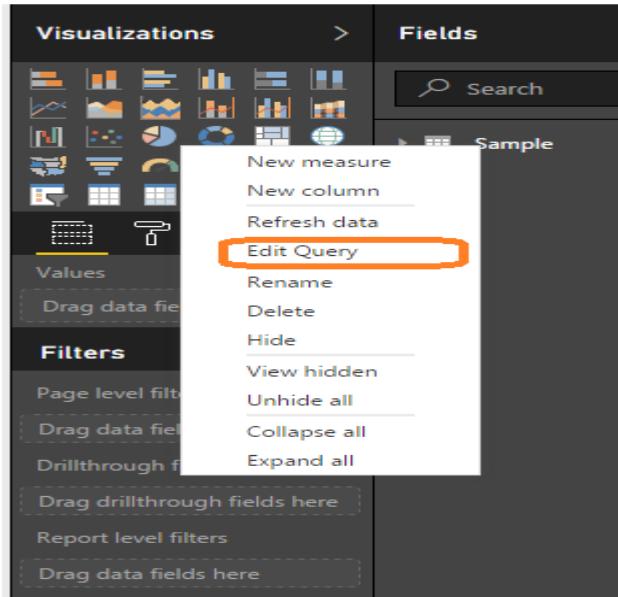
You can open Power Query Editor in three different ways

1. From Home Tab you can find Edit Queries.
2. In the Table Level Options you can find Edit Query.
3. While loading the table edit option that takes you to the Edit Queries.

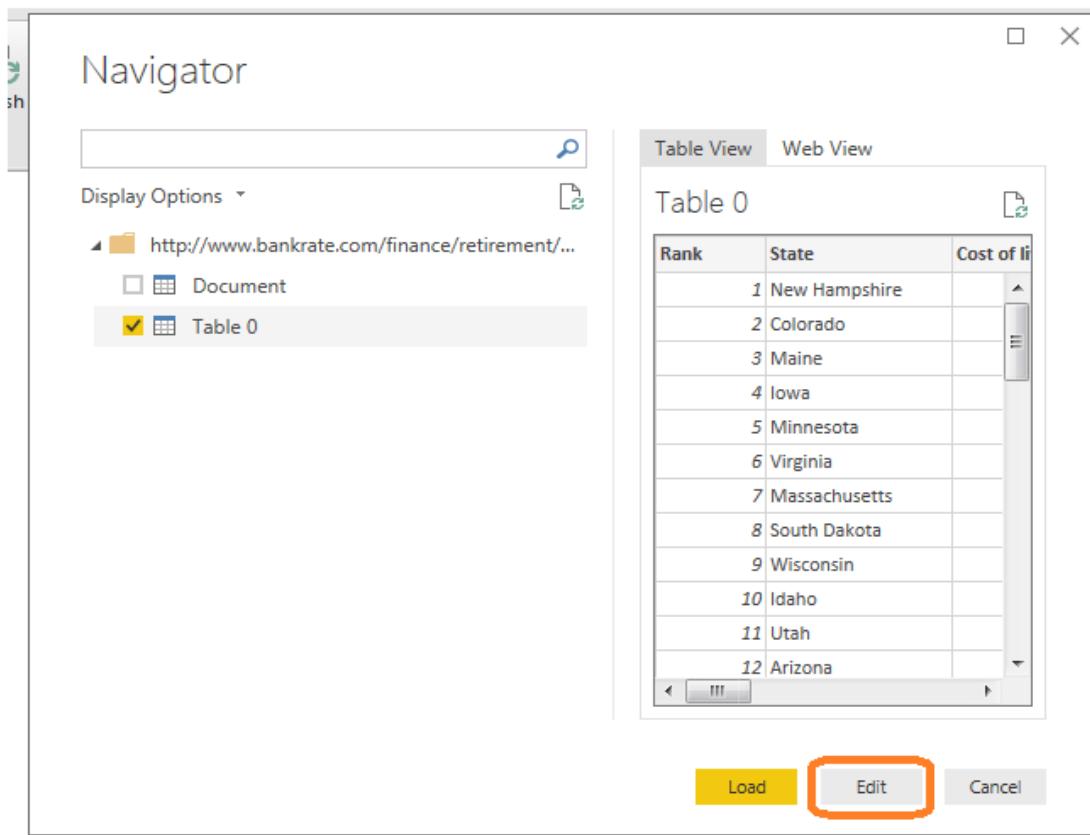
To get into Query Editor, select Edit Queries from the Home tab of Power BI Desktop.



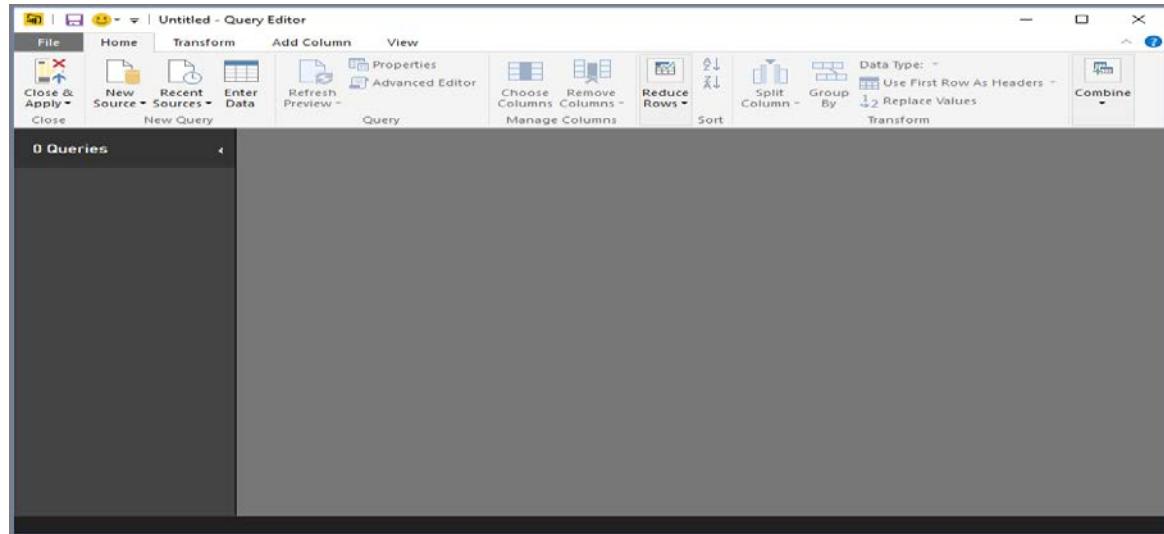
Other way to get or open Query Editor is, go to Table Level Options you can find Edit Query.



Third way is while loading the table “Edit” option that takes you to the Edit Queries.



With no data connections, Query Editor appears as a blank pane, ready for data as shown below.



### How to establish connection to the source?

Home Tab → New Source → Get Data window → Select the Source Type → Select the Source → Ok

Once Query Editor is loaded with data that's ready for you to shape, you see a handful of sections. Here's how Query Editor appears once a data connection is established.

A screenshot of the Microsoft Power Query interface with a data connection established. The title bar says "Untitled - Query Editor". The ribbon has tabs for File, Home (highlighted with a red circle 1), Transform, Add Column, and View. Under the Home tab, there are buttons for Close & Apply, New Source (highlighted with a red circle 2), Recent Sources, Enter Data, Refresh Preview, Properties, Advanced Editor, Choose Columns, Remove Columns, Reduce Rows, Sort, Split Column, Group By, and Transform. A status bar at the bottom shows "1 Query" and "Table 0".  
  
The main area displays a preview of the data in a table. The table has three columns: "Header", "Overall rank", and "State". The data consists of 18 rows, each containing a header and a state name. Row 1: "Check out how your state ranks for retirement" and "Wyoming". Row 2: "Check out how your state ranks for retirement" and "Colorado". Row 3: "Check out how your state ranks for retirement" and "Utah". Row 4: "Check out how your state ranks for retirement" and "Idaho". Row 5: "Check out how your state ranks for retirement" and "Virginia". Row 6: "Check out how your state ranks for retirement" and "Iowa". Row 7: "Check out how your state ranks for retirement" and "Montana". Row 8: "Check out how your state ranks for retirement" and "South Dakota". Row 9: "Check out how your state ranks for retirement" and "Arizona". Row 10: "Check out how your state ranks for retirement" and "Nebraska". Row 11: "Check out how your state ranks for retirement" and "Minnesota". Row 12: "Check out how your state ranks for retirement" and "Maine". Row 13: "Check out how your state ranks for retirement" and "North Dakota". Row 14: "Check out how your state ranks for retirement" and "Kansas". Row 15: "Check out how your state ranks for retirement" and "Vermont". Row 16: "Check out how your state ranks for retirement" and "New Hampshire". Row 17: "Check out how your state ranks for retirement" and "Wisconsin". Row 18: "Check out how your state ranks for retirement" and "Massachusetts".  
  
A "Query Settings" pane is open on the right side, containing sections for "PROPERTIES" (Name: Table 0, highlighted with a red circle 4) and "APPLIED STEPS" (Source, Navigation, Changed Type, highlighted with a red circle 5). The bottom of the screen shows "9 COLUMNS, 50 ROWS" and "PREVIEW DOWNLOADED AT 10:14 AM".

1. In the ribbon, many buttons are now active to interact with the data in the query for data preparation.
2. In the left pane or queries pane, queries (one for each table, or entity) are listed and available for selection, viewing, and shaping.
3. In the center pane or Results Pane, data from the selected query is displayed and available for shaping.
4. The Query Settings window appears, listing the query's properties and applied steps.
5. The Formula bar is the place where you can see and edit the M code of the current transformation step.

We'll look at each of these four areas

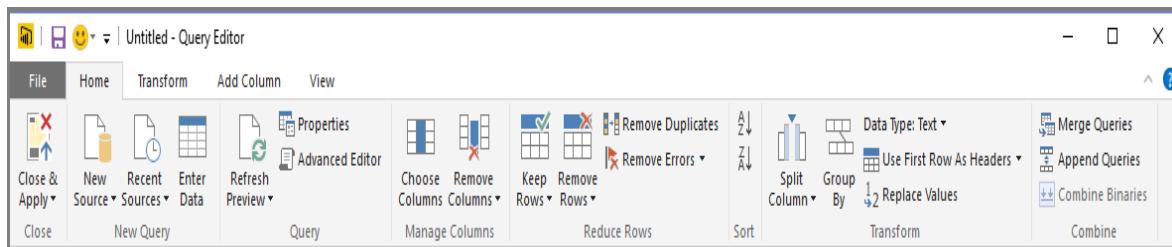
- ✓ The ribbon
- ✓ The queries pane
- ✓ The data view / Results Pane
- ✓ The Query Settings pane
- ✓ Formula Bar

## The Query Ribbon

The ribbon in Query Editor consists of four tabs - Home, Transform, Add Column, and View.

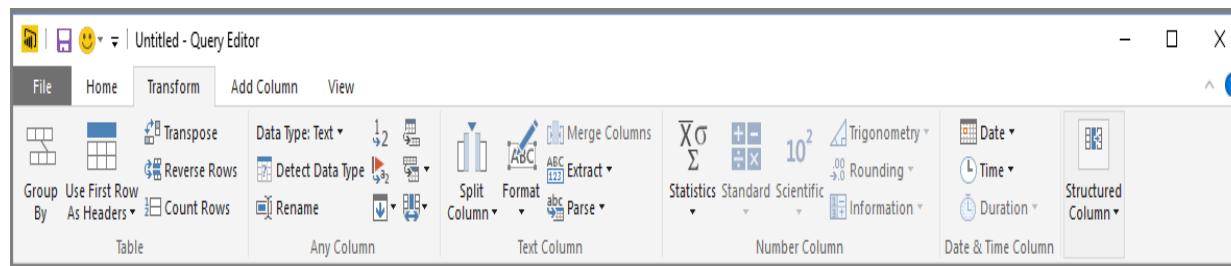
### Home Tab

The Home tab contains the **common query tasks**, including the first step in any query, which is Get Data. The following image shows the Home ribbon.



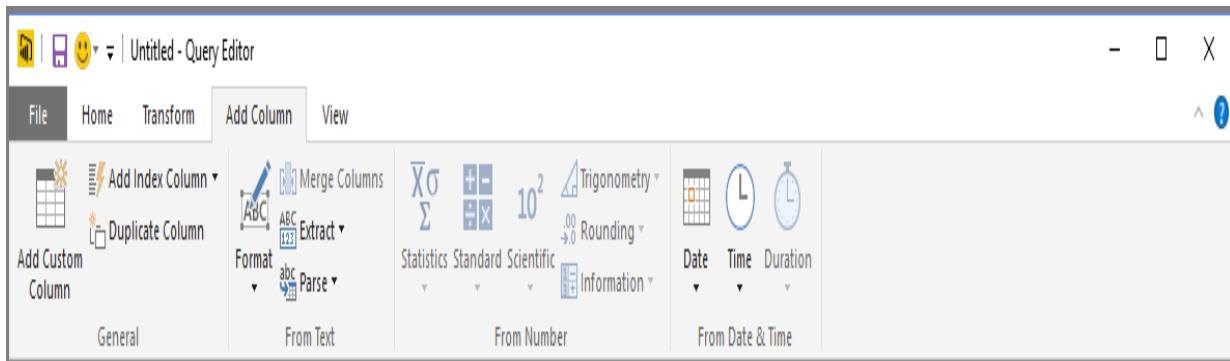
### Transform Tab

The Transform tab provides access to common **data transformation tasks**, such as adding or removing columns, changing data types, splitting columns, and other data-driven tasks. The following image shows the Transform tab.



## Add Column Tab

The Add Column tab provides additional tasks associated with adding a column, formatting column data, and adding custom columns. The following image shows the Add Column tab.



## The Difference between the Transform and Add Column Tabs

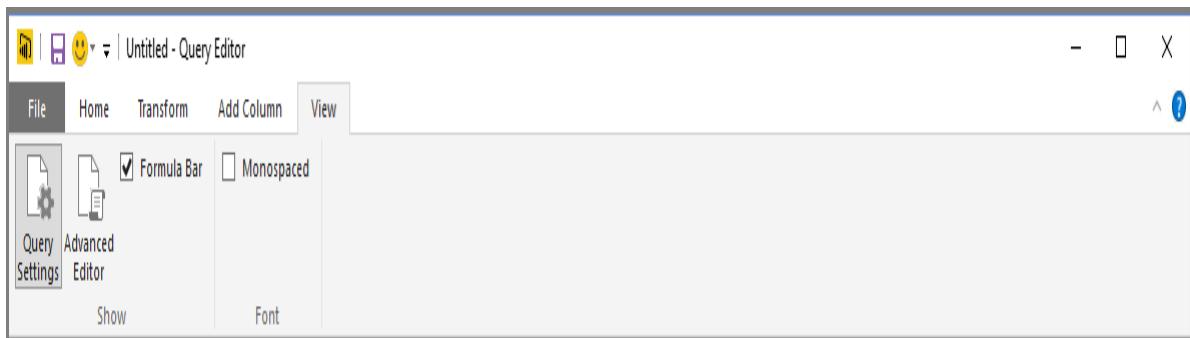
The bulk of all transformations available in power query can be accessed through either the Transform tab or the Add Column tab.

You might think there is a lot of duplication between these two tabs. For example, both tabs contain a form Text section with a lot of the same commands. It's not really the case, there is a subtle difference!

When you use a command from the Add Column tab that is found in both tabs, it will create a new column with the transformed data and the original column will stay intact. Whereas using the equivalent command from the Transform tab will change the original column and no new column is created. This is a critical point to be aware of!

## View Tab

The View tab on the ribbon is used to toggle whether certain panes or windows are displayed. It's also used to display the Advanced Editor. The following image shows the View tab.



It's useful to know that many of the tasks available from the ribbon are also available by right-clicking a column, or other data, in the center pane.

## The Left Pane / Queries Pane

The left pane displays the number of active queries, as well as the name of the query. When you select a query from the left pane, its data is displayed in the center pane, where you can shape and transform the data to meet your needs. The following image shows the left pane with multiple queries.

A screenshot of the Microsoft Power Query Editor interface. The left pane, titled '3 Queries', lists three items: 'RetirementStats', 'Products\_by\_Categories', and 'StateCodes'. A pink arrow points from the text above to this list. The right pane displays a table with columns 'Overall rank', 'State', and 'Cost of living'. The data shows the top 9 states by overall rank: Wyoming, Colorado, Utah, Idaho, Virginia, Iowa, Montana, South Dakota, and Arizona.

Overall rank	State	Cost of living
1	Wyoming	
2	Colorado	
3	Utah	
4	Idaho	
5	Virginia	
6	Iowa	
7	Montana	
8	South Dakota	
9	Arizona	

## The Center (Data) Pane / Results Pane

In the center pane, or Data pane, data from the selected query is displayed. This is where much of the work of the Query view is accomplished.

Notice that many of these right-click menu items are the same as buttons in the ribbon tabs.

When you select a right-click menu item (or a ribbon button), Query applies the step to the data, and saves it as part of the query itself. The steps are recorded in the Query Settings pane in sequential order, as described in the next section.

## The Query Settings Pane

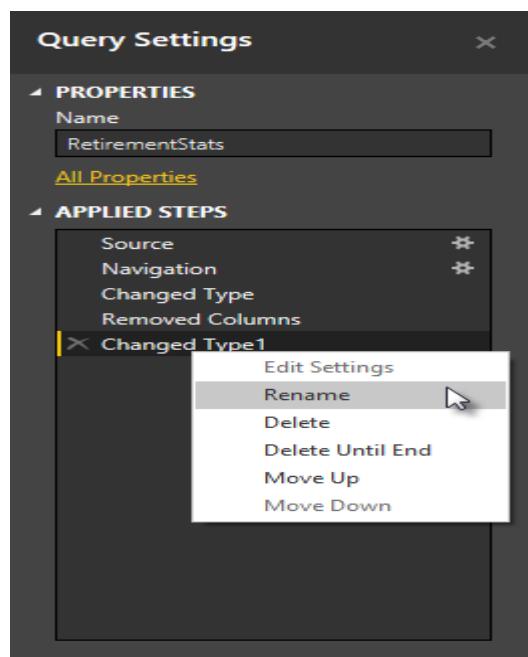
The Query Settings pane is where all steps associated with a query are displayed. For example, in the following image, the Applied Steps section of the Query Settings pane reflects the fact that we just changed the type of the Overall score column.

A screenshot of the 'QUERY SETTINGS' pane. It has two main sections: 'PROPERTIES' and 'APPLIED STEPS'. The 'PROPERTIES' section shows a 'Name' field set to 'Orders' and a link to 'All Properties'. The 'APPLIED STEPS' section lists three steps: 'Source', 'Navigation', and 'Promoted Headers'. The 'Promoted Headers' step is highlighted with a yellow selection bar.

As additional shaping steps are applied to the query, they are captured in the **Applied Steps** section.

It's important to know that the underlying data is not changed rather Query Editor adjusts and shapes its view of the data, and any interaction with the underlying data occurs based on Query Editor's shaped and modified view of that data.

In the Query Settings pane, you can rename steps, delete steps, or reorder the steps as you see fit. To do so, right-click the step in the Applied Steps section, and choose from the menu that appears. All query steps are carried out in the order they appear in the Applied Steps pane.



## Formula Bar

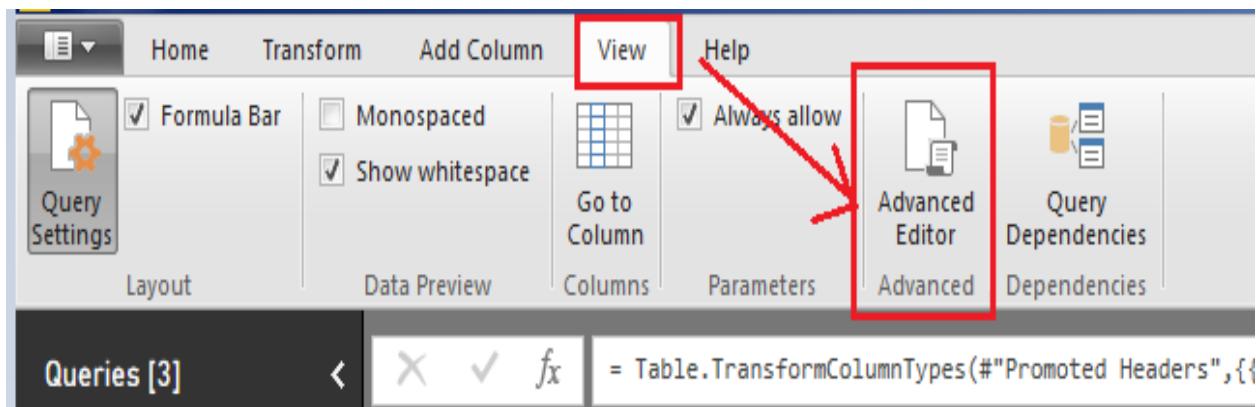
This is where you can see and edit the M code of the current transformation step. Each transformation you make on your data is recorded and appears as a step in the applied steps area.

A screenshot of the Power BI Query Editor. At the top, there is a formula bar with several icons: a red X, a green checkmark, and a fx symbol. To the right of these icons is the M code: '= Table.TransformColumnTypes(Returns1,{{"Column1", type text}, {"Column2", type text}})'. Below the formula bar is a preview table with two columns: 'Column1' and 'Column2'. The table has four rows of data:

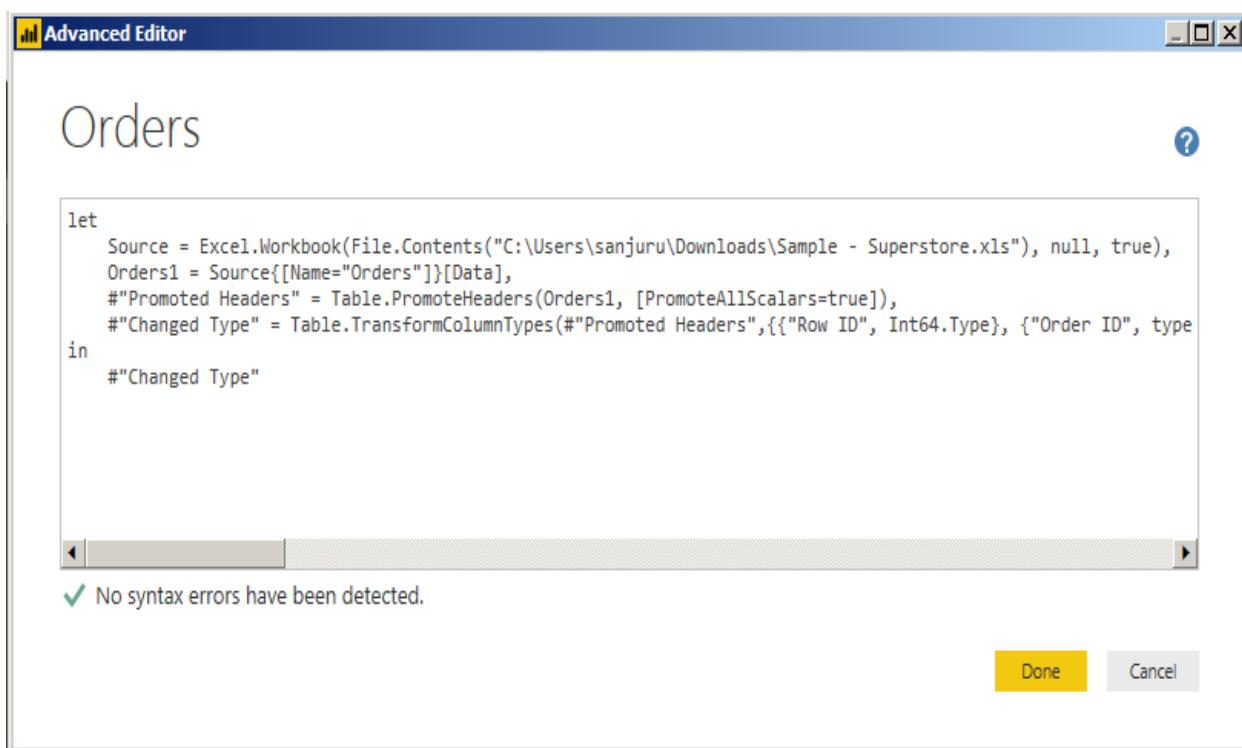
	Column1	Column2
1	Returned	Order ID
2	Yes	CA-2017-153822
3	Yes	CA-2017-129707
4	Yes	CA-2014-152345

## The Advanced Editor

If you want to see the code that Query Editor is creating with each step, or want to create your own shaping code, you can use the Advanced Editor. To launch the advanced editor, select View from the ribbon, then select Advanced Editor.

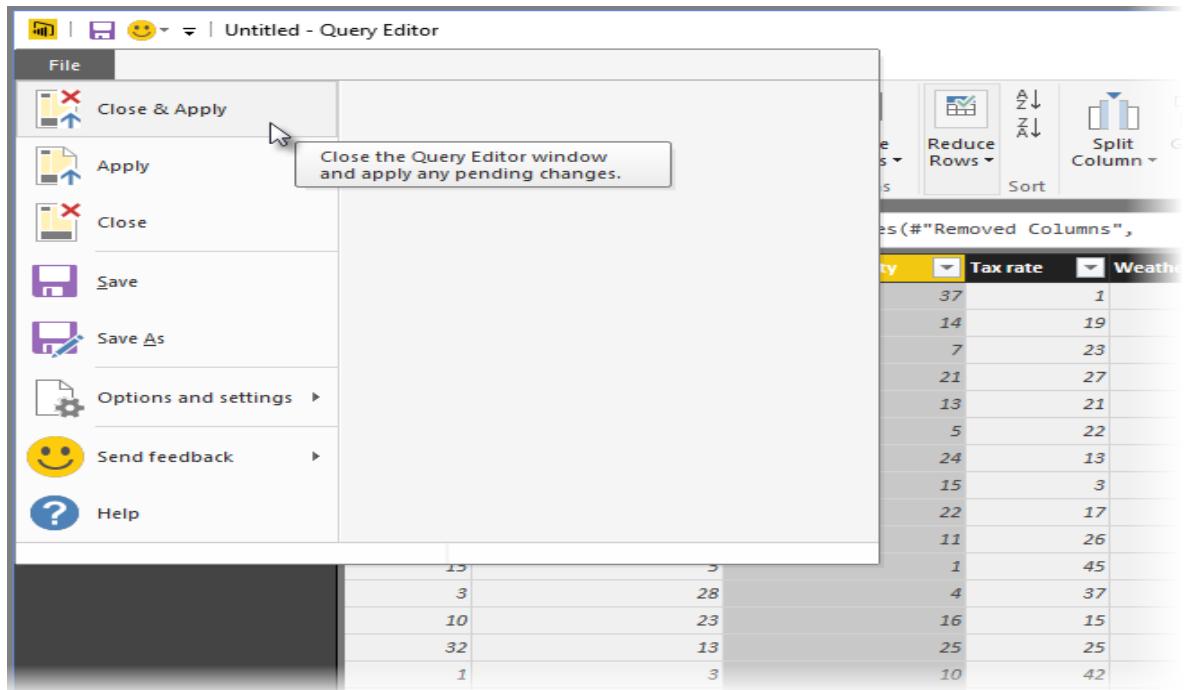


A window appears, showing the existing Query code. You can directly edit the code in the Advanced Editor window. To close the window, select the done or Cancel button.



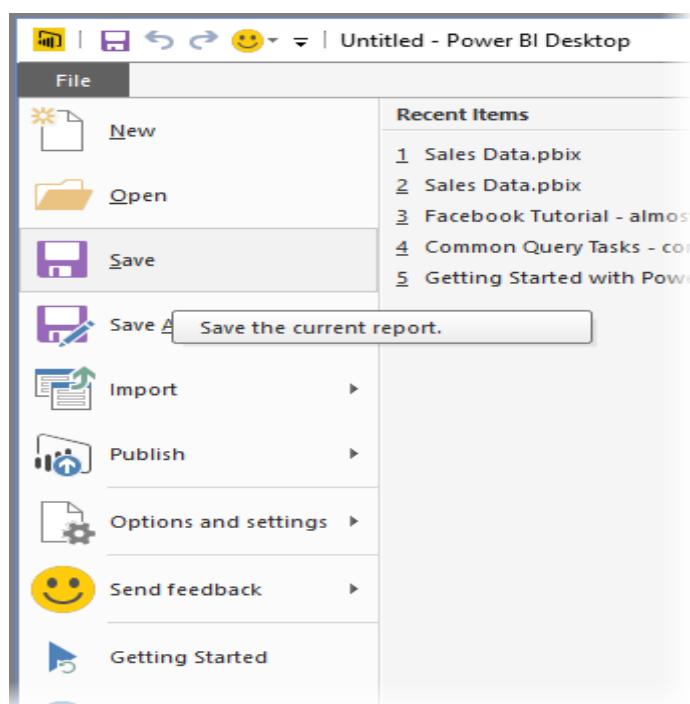
## Saving Your Work

When your query is where you want it, you can have Query Editor apply the changes to the data model into Power BI Desktop, and close Query Editor. To do that, select Close & Apply from Query Editor's File menu as shown below.

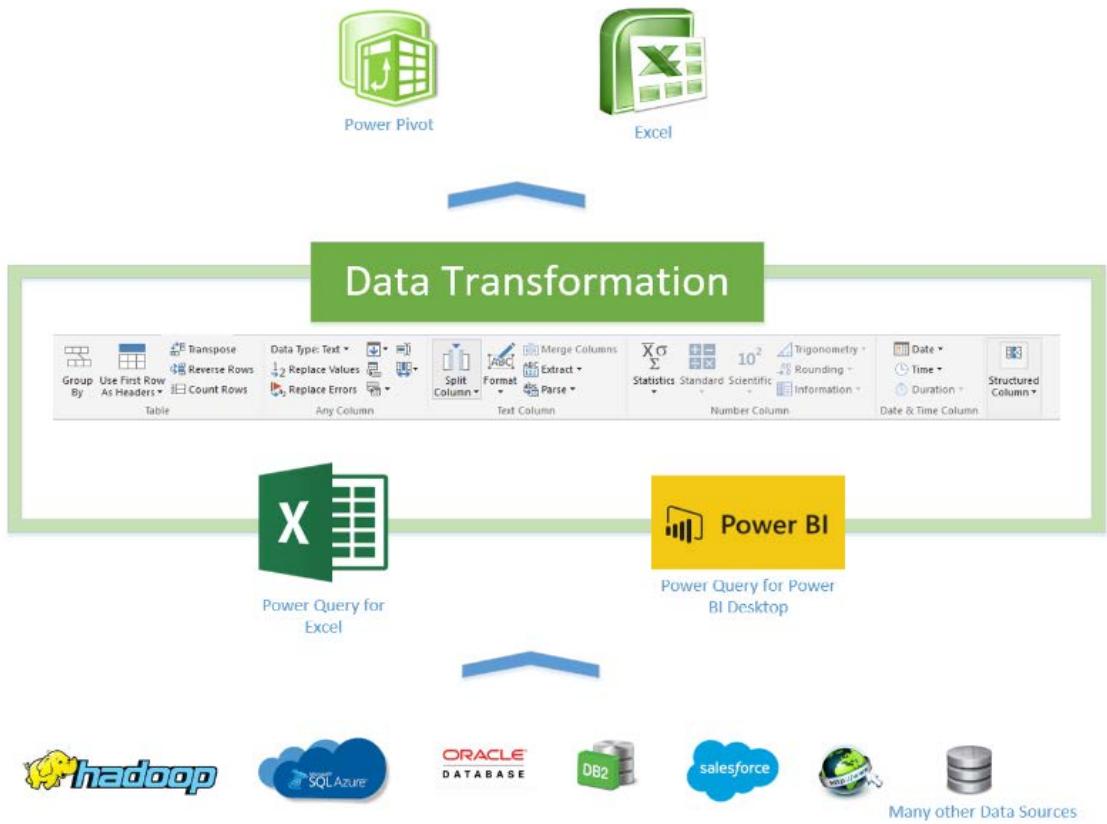


Once you have your query where you want it, or if you just want to make sure your work is saved, Power BI Desktop can save your work in the form of ". pbix" file.

To save your work, select File > Save (or File > Save As), as shown in the following image.



In below diagram you can see a high level diagram of Power Query conceptually



## Data Type

Data Type represents the type of information stored in the memory location or Column. Each column should have only one Data Type.

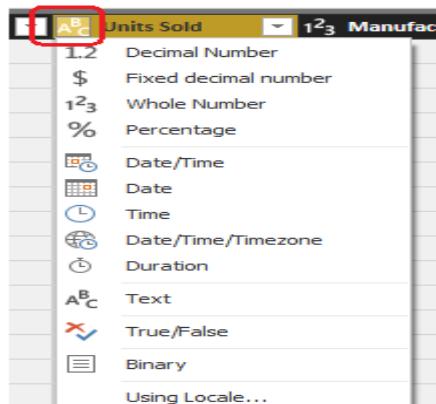
### Change Data Type of a Column in Power BI

When you import or load a table from any data source, Power BI will automatically detect the data type of a column. However, there may be some situations where Power BI might get them wrong. For example, it may consider amounts, values, or even dates as the text. Now we will see how to Change Data Types of a Column in Power BI with example. Changing data type of the column is important as DAX functions have special data type requirements and also filtering options will change based on data type of the column.

In Query Editor or Power Query you can change the Data Type of a column in different ways.

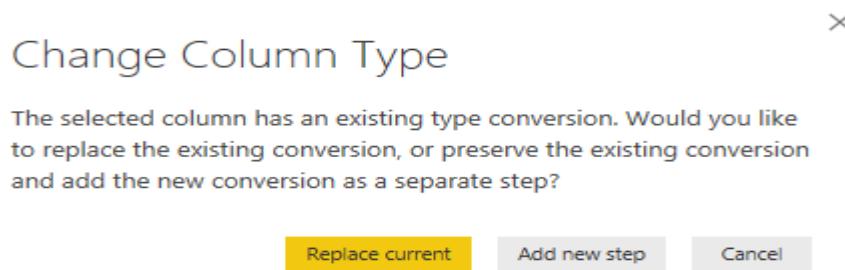
#### Approach 1

In the below image for Units Sold Column, Power BI identified it as string column. But actually we have Decimal Numbers as that column values.



So to change the data type, select the Column for which you want to change the data type. Next, click on the left corner of the column header which is marked in Red Box. Then select the data type which is appropriate, here Decimal Number.

Changing data type of a column will open the following pop up window. You can Choose "Replace Current" to update current step or also you can choose "Add new step" to add a new transformation step to the Query.



## Approach 2

Select the Column name that you want to alter the data type, and click on the Data Type button under the Home tab in Power Query Ribbon.

The screenshot shows the Power Query Editor interface. The ribbon at the top has 'Home' selected. On the far right of the ribbon, there is a 'Data Type' dropdown menu with a red box around it. The menu is currently set to 'Text'. Below the menu is a list of other data types: Decimal Number, Fixed decimal number, Whole Number, Percentage, Date/Time, Date, Time, Duration, Text, True/False, and Binary. The main area of the editor shows a table with columns: Segment, Country, Product, Discount Band, Units Sold, and Man. The 'Units Sold' column is currently set to 'Text' data type. The table contains 10 rows of data.

## Approach 3

Select the Column that you want to change the data type and right-click on it will open the context menu. Select the Change Type and then select the data type from the list. For now, we are selecting the Decimal Number.

This screenshot shows a context menu for the 'Units Sold' column in the Power Query Editor. The 'Change Type' option is highlighted with a blue selection bar. To the right of the menu, a list of data types is displayed, with 'Decimal Number' currently selected and highlighted in blue. Other options in the list include 'Fixed decimal number', 'Whole Number', 'Percentage', 'Date/Time', 'Date', 'Time', 'Duration', 'Text', 'True/False', and 'Binary'. The main table area shows the 'Units Sold' column with values 1618.5, 1321, 2178, 888, 2470, 1513, 921, 2518, 1899, and 1545.

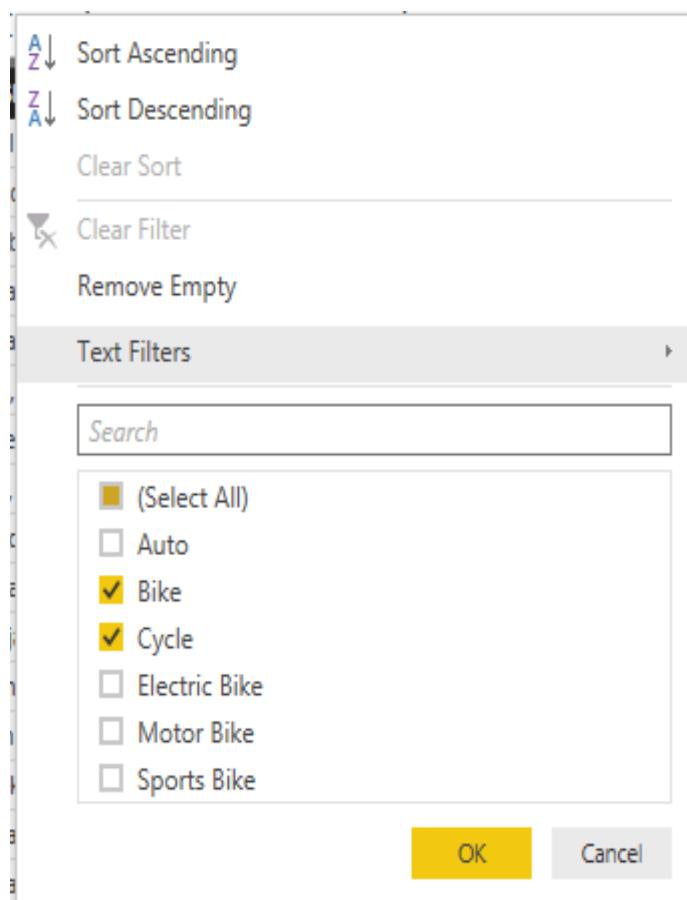
## Filtering select Rows in Power Query / Filters in Power Query

Data Type of column has impact on filtering options available. Filter options changes with respect to data types. Before going to filter rows check the data types of the columns.

"Text Filtering Options Are Case Sensitive".

## Filter a column using an Auto Filter / Basic Filtering

- ✓ Select the column that you need to filter.
- ✓ Click the down arrow (▼).
- ✓ Uncheck the Select All box to deselect all Column Values.
- ✓ Select the column values you want to include in your table.
- ✓ Click OK.



"Search Bar is Case Insensitive".

### Note:

Be careful if you are filtering the rows using Search Bar. Always look at the M code return by Power Query and cross check it is filtering as expected.

Basic Filtering is good only if you want to do equity filtering for values that exists in the current data set, however it won't work correctly if you want to check ranges, or contains or things that is not an exact equity filter. Advanced Filtering is the correct way of filtering in Power Query, and there are advanced filters for all types of data types; Numbers, Text, Date...

When you filter a column, only the top 1,000 distinct values in the column will load into the filter list. If there are 1,000 or more values in the column in Query Editor that you are filtering, a message will appear indicating that the list of values in the filter list may be incomplete, and the Load more link is shown. Click the Load more link to load another 1,000 distinct values.

If exactly 1,000 distinct values are found again, the list is displayed with a message stating that the list could still be incomplete.

If less than 1,000 distinct values are found, the full list of values is shown.

### Filter a Column using Text Filters

In addition to the “Auto Filters” or Basic Filtering, you can filter a Text values using the Text Filters context menu.

Click the down arrow (▼) of the column containing a Text values you want to filter on.

Click Text Filters and select the filter option required from Context Menu.

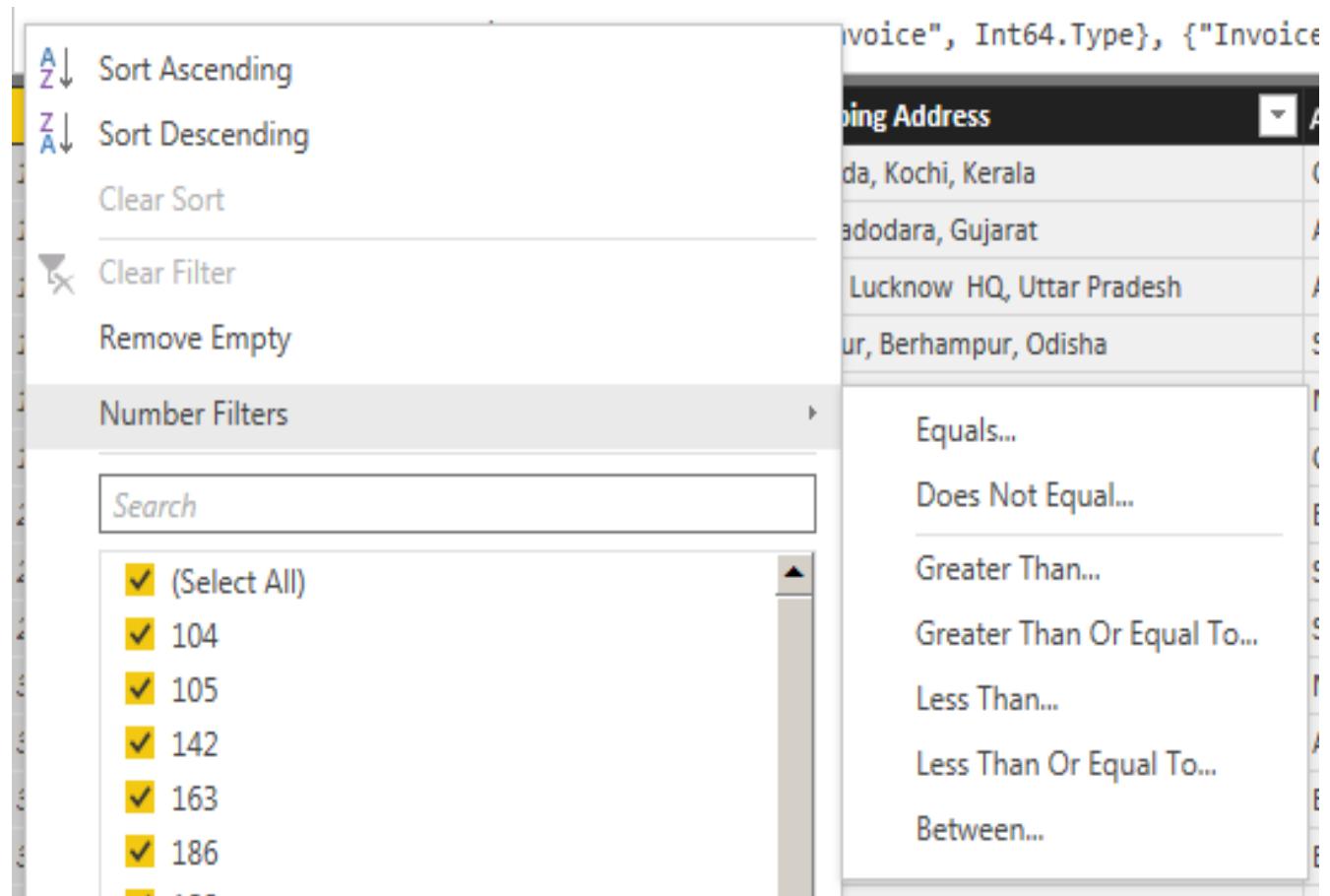
The screenshot shows the Microsoft Query Editor interface. A context menu is open over the 'Inv' column header. The menu includes options for sorting (Sort Ascending, Sort Descending), clearing sort (Clear Sort, Clear Filter, Remove Empty), and applying text filters. The 'Text Filters' submenu is expanded, showing a search bar and a list of distinct values: BENIWAL R., BHARAT S., GORAKH S., KARUNESH D., KIRAN J., NARENDRA P., and MITTAL D. To the right of the main window, a list of shipping addresses is visible: Irinjalakuda, Kochi, Kerala; Faizabad, Lucknow HQ, Uttar Pradesh; Shrirampur, Pune, Maharashtra; and Mahesana, Ahmedabad HQ, Gujarat.

## Filter a Column using Number Filters

In addition to the “Auto Filters”, you can filter Number values using the Number Filters Context Menu.

To filter a column using Number Filters, Click the down arrow (▼) of the column containing a Number values you want to filter on.

Click Number Filters, and select the filter option required from Context Menu.

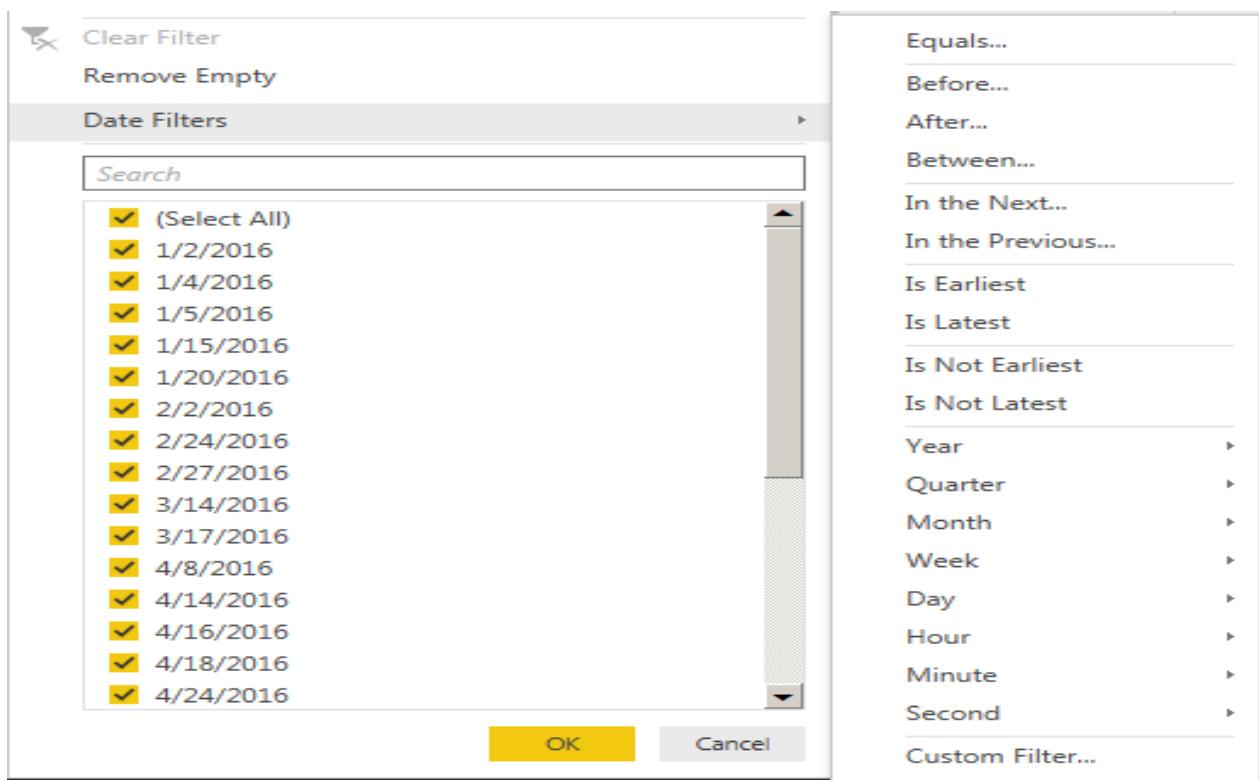


## Filter a Column using Date Filters

In addition to the “Auto Filters”, you can filter Date values using the Date Filters Context Menu.

To filter a column values using Date Filters, Click the down arrow (▼) of the column containing Date values you want to filter on.

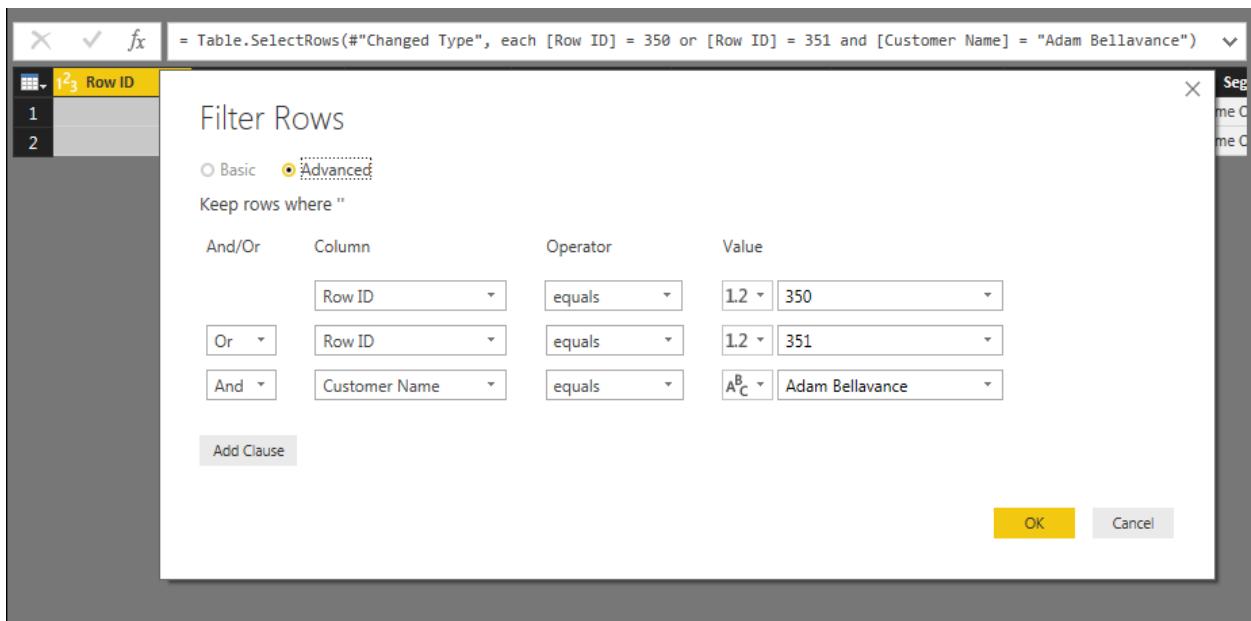
Click Date Filters, and select the filter option required from Context Menu.



## Filter Multiple Columns

To filter multiple columns, select an additional column, and repeat one of the column filter steps. **AND** Operation will be performed between the columns if you apply filters on multiple columns individually.

Other way is by Using **Advanced** option in Filter Rows you can apply filters on multiple columns at a time. Here you can select **And / Or** operation between columns.



In below image we applied filters on Row ID and Customer Name Column and you can see M Language Code.

The screenshot shows the Power BI Data Editor interface. At the top, there is a formula bar with the following M language code:

```
= Table.SelectRows(#"Changed Type", each ([Customer Name] = "Adam Bellavance") and ([Row ID] = 350 or [Row ID] = 351))|
```

Below the formula bar is a table preview showing two rows of data. The columns are: Row ID, Order ID, Order Date, Ship Date, Ship Mode, Customer ID, Customer Name, and Seg. The data is as follows:

	Row ID	Order ID	Order Date	Ship Date	Ship Mode	Customer ID	Customer Name	Seg
1	350	CA-2016-129714	9/1/2016	9/3/2016	First Class	AB-10060	Adam Bellavance	Home C
2	351	CA-2016-129714	9/1/2016	9/3/2016	First Class	AB-10060	Adam Bellavance	Home C

## Inbuilt Column Transformations

- ✓ Remove Columns / Remove Other Columns
- ✓ Name / Rename a Column
- ✓ Reorder Columns or Sort Columns
- ✓ Add Column / Custom Column
- ✓ Split Columns
- ✓ Merge Columns
- ✓ Pivot, Unpivot Columns
- ✓ Transpose Columns

### Remove Columns / Remove Other Columns

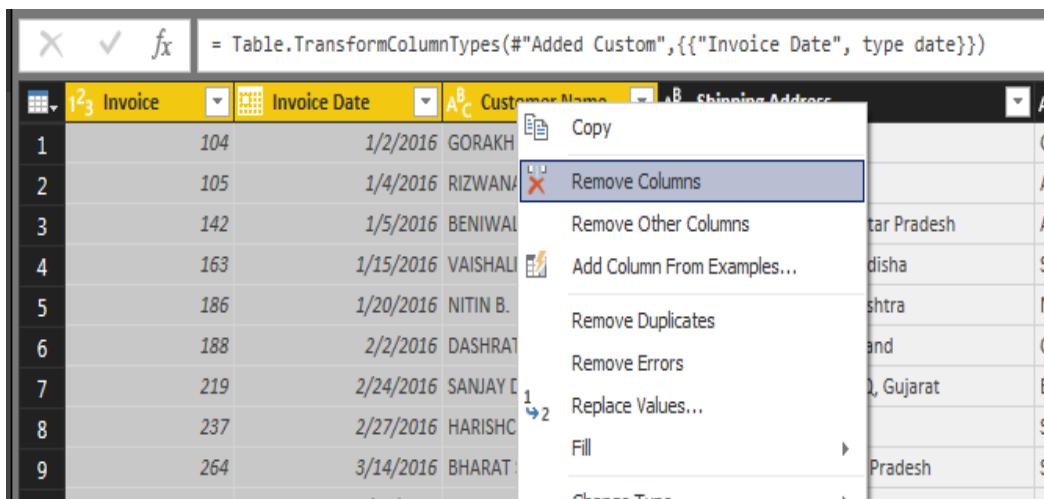
If you want to remove unwanted columns which are not necessary in your data model for the data source in your query, you can use Remove Columns / Remove Other Columns option as shown in the below image.

If you want remove selected columns

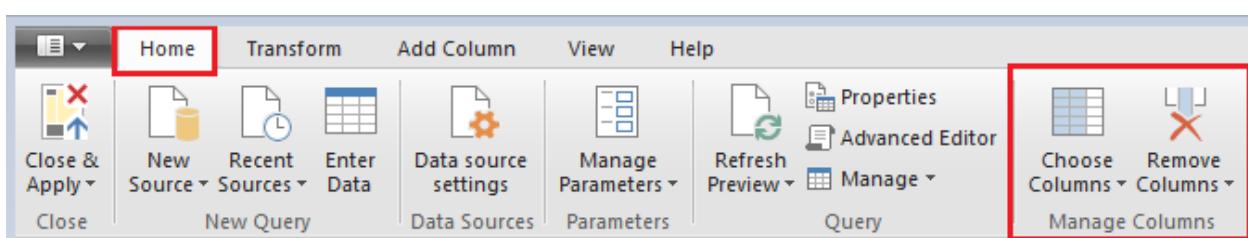
Select the columns you want to remove → Right Click → Remove Columns

If you want to remove all other columns other than selected

Select the columns you want to keep → Right Click → Remove Other Columns



You can also remove columns from Manage Columns Section in Home Tab in Query Editor Ribbon.



## Name or Rename a Column

When you want give a meaningful name for a column that needs in Report, you can do it by renaming a column. To rename a column

Right click a column → Rename → Enter a meaningful Name

(OR)

Double click the column → that will allow you to edit Name → Enter a meaningful Name

## Reorder Columns or Sort Columns in Power BI

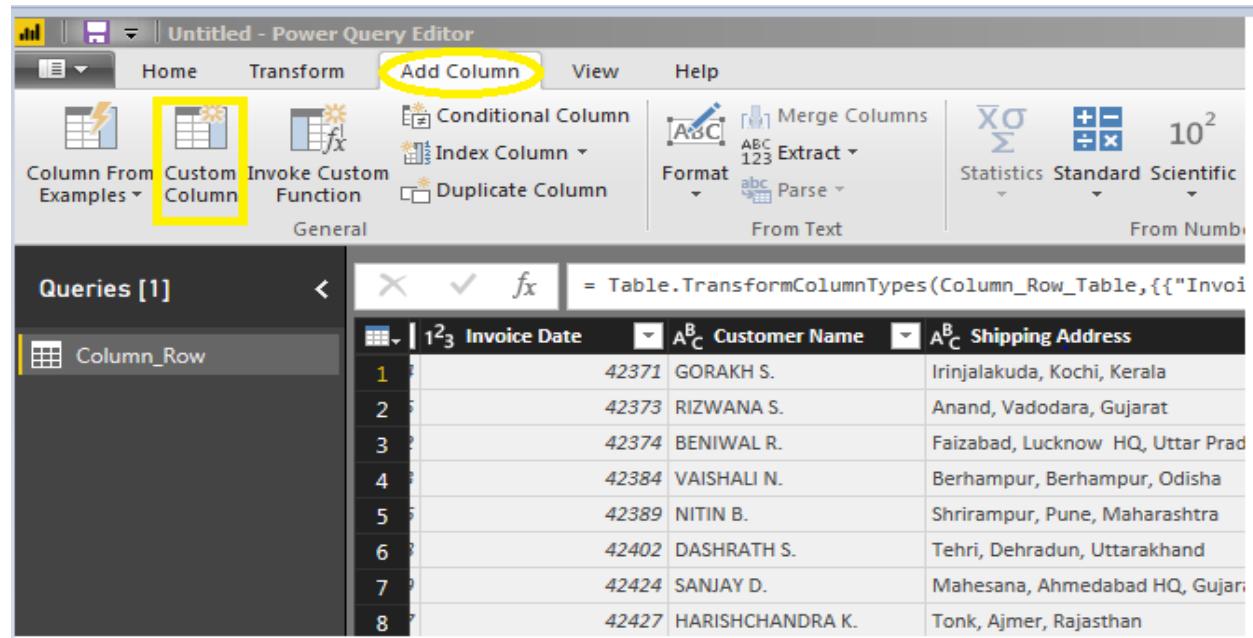
One way is drag the required column and Drop at the position you want to place.

Second way, right-click on the column name that you want to move will open the context menu. Select the Move and then select Right, Left, To End, or To Beginning options.

## Add Column / Add Custom Column

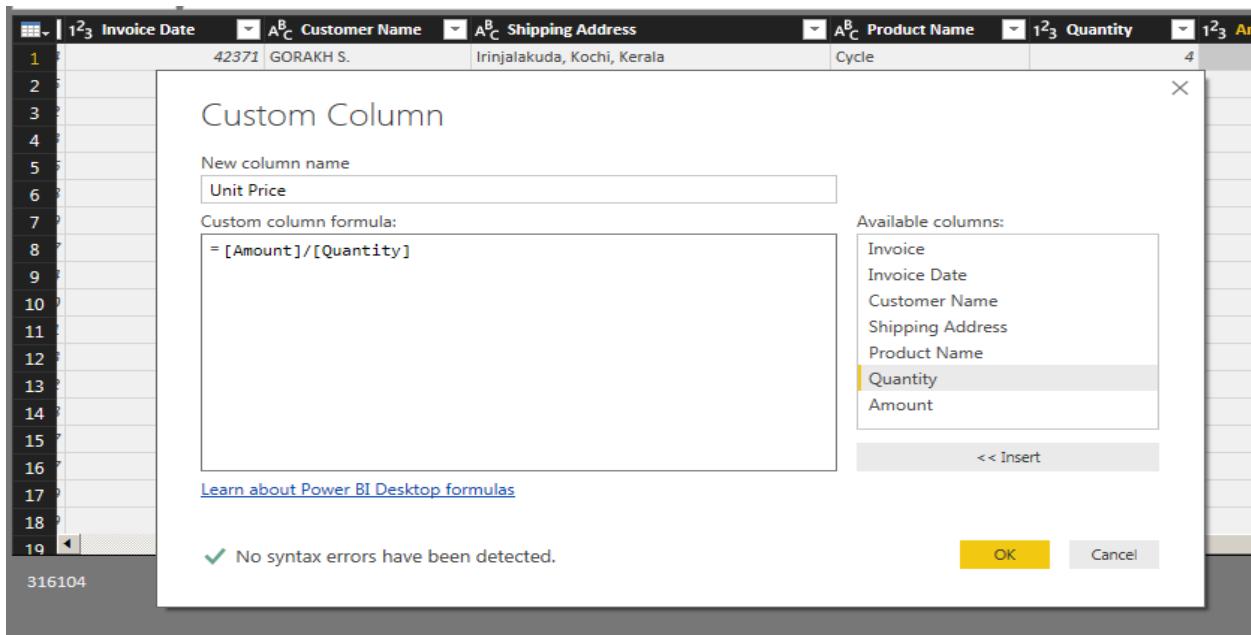
In Query Editor you can create custom formulas that operate on multiple columns in your table, and then place the results of such formulas into a new (custom) column. Query Editor makes it easy to create custom columns.

In Query Editor, select Custom Column from the Add Column tab on the ribbon.



The screenshot shows the Power Query Editor interface. The ribbon at the top has tabs for Home, Transform, and Add Column (which is highlighted with a yellow circle). Below the ribbon is a toolbar with various icons for operations like Conditional Column, Index Column, Duplicate Column, Merge Columns, Extract, Format, Parse, and Statistics. The main area is divided into two panes: 'Queries [1]' on the left and a data grid on the right. The data grid contains 8 rows of data with columns for Invoice Date, Customer Name, and Shipping Address. The first row shows '42371 GORAKH S.' and 'Irinjalakuda, Kochi, Kerala'. The second row shows '42373 RIZWANA S.' and 'Anand, Vadodara, Gujarat'. The third row shows '42374 BENIWAL R.' and 'Faizabad, Lucknow HQ, Uttar Pradesh'. The fourth row shows '42384 VAISHALI N.' and 'Berhampur, Berhampur, Odisha'. The fifth row shows '42389 NITIN B.' and 'Shrirampur, Pune, Maharashtra'. The sixth row shows '42402 DASHRATH S.' and 'Tehri, Dehradun, Uttarakhand'. The seventh row shows '42424 SANJAY D.' and 'Mahesana, Ahmedabad HQ, Gujarat'. The eighth row shows '42427 HARISHCHANDRA K.' and 'Tonk, Ajmer, Rajasthan'.

Once we select custom column a Custom Column window opens as below where we can provide New column name and Custom column formula.

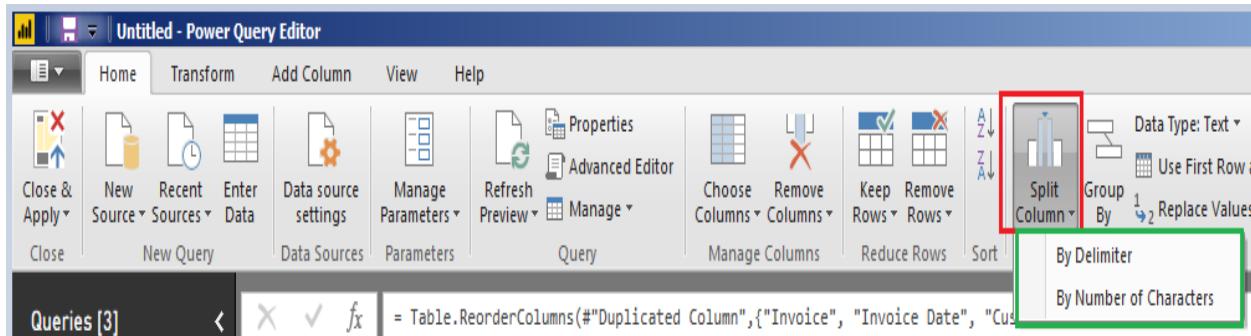


## Split Columns

Sometimes you will get merged columns (one column with too much information). In that situation, you can use Power BI Split Columns option to split that column into multiple columns.

We have two options here to split columns as shown in below image.

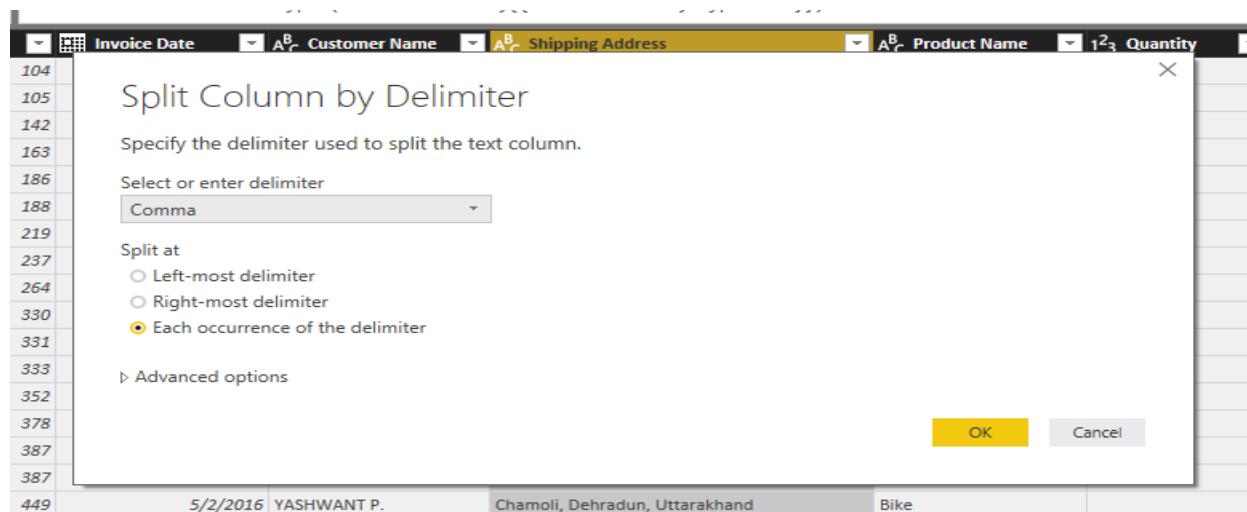
- By Delimiter
- By Number of Characters



In order to split the columns in a table, right-click on the column that you want to split will open the context menu. Select the Split Columns and then select “By Delimiter” option.

Customer Name	Shipping Address	Product Name	Quantity
GORAKH S.	Irinjalakuda,		4
RIZWANA S.	Anand, Vadodara		3
BENIWAL R.	Faizabad, Lucknow		1
VAISHALI N.	Berhampur, Odisha		1
NITIN B.	Shrirampur, Maharashtra		3
DASHRATH S.	Tehri, Dehradun		1
SANJAY D.	Mahesana, Gujarat		5
HARISHCHANDRA K.	Tonk, Ajmer		1
BHARAT S.	Khandwa, Madhya Pradesh		1
TANAJI V.	Anakapalle, Andhra Pradesh		5
SUPRIYA S.	Shahjahanpur, Uttar Pradesh		1
VIDYA K.	Madhubani, Bihar		3
NARENDRA P.	Sitamarhi, Bihar		4
NARESH D.	Gokak, North Karnataka		1
RAVIKUMAR S.	Cachar, Guwahati		1

Selecting the “By Delimiter” option will open the following window.



### Select or enter delimiter

Please select the delimiter that you want to use as the split character from the drop down list. If it is not there in the list, then select Custom option in the drop down and specify that custom character.

#### Left most delimiter

This option will split the left most string before first delimiter.

#### Right most delimiter

This option will split right most string after the last delimiter.

## Each Occurrence of the delimiter

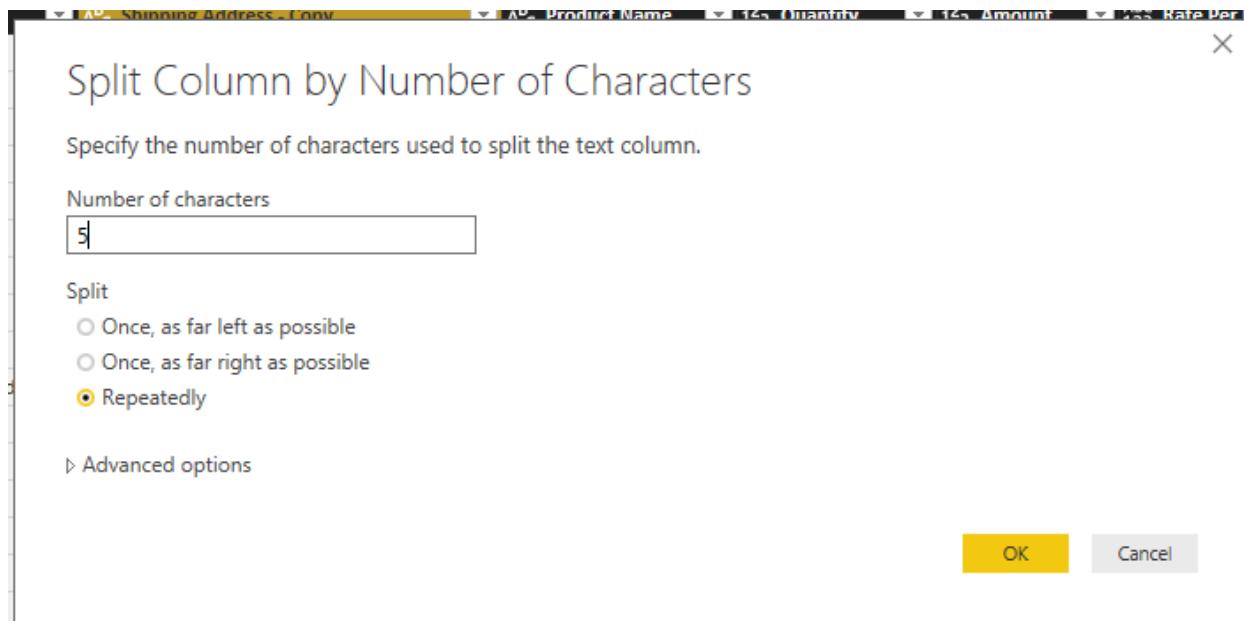
Text will split at each occurrence of a delimiter.

## Split Columns by Number of Characters

Right-click on the column that you want to split will open the context menu. Please select the Split Columns and then select “By Number of Characters” option.

The screenshot shows the Power BI desktop interface with three queries listed in the 'Queries [3]' pane: 'Column Row', 'Transpose2', and 'Unpivot1'. In the main workspace, there is a table with columns: 'Address', 'Shipping Address - Copy', 'Product Name', 'Quantity', and 'Amount'. The 'Address' column contains state names like 'Kerala', 'Gujarat', etc. The 'Shipping Address - Copy' column contains city names like 'Irinjalakuda, Kochi, Kerala', 'Anand, Vadodara, Gujarat', etc. A context menu is open over the 'Shipping Address - Copy' column, with the 'Split Column' option highlighted. A sub-menu for 'Split Column' is open, showing 'By Delimiter...' and 'By Number of Characters...'. Other options in the context menu include Copy, Remove, Remove Other Columns, Duplicate Column, Add Column From Examples..., Remove Duplicates, Remove Errors, Change Type, Transform, Replace Values..., Replace Errors..., Group By..., and others.

Selecting the “By Number of Characters” option will open the Split Column by Number of Characters window.



### Number of Characters

Please specify the number of characters used to split the column. Let us give as 5.

#### Once, as far left as possible

This option will split the given string into two strings first string with first 5 Characters and second string with remaining characters.

## Once, as far right as possible

This option will split the given string into two strings first string with all the characters except last 5 Characters and second string will be last 5 characters.

## Repeatedly

Text will split for every 5 characters.

## Merge Columns

With Power Query, you can merge two or more columns in your query. You can merge columns to replace them with a merged column, or create a new merged column alongside the columns that are merged.

### Merge columns to replace existing columns

Select two or more columns that you need to merge. Press the CTRL key, and then click on the column headers to select each of the columns that you'll include in the merge.

**NOTE:** The order in which you select the columns sets the order of the values in the merged column.

Right-click the columns and click Merge Columns.

In the Merge Columns popup window, specify the separator that is in use between each of the column values. You can select from predefined separator values, or specify a custom separator value. Give a meaningful Name in “New column name” section.



Click OK to create a merge column that replaces the columns selected for the merge operation.

## Merge Columns to create a new column

Perform all the above steps in Add Column Tab in Query Ribbon to create a new column for all the merged columns.

## PIVOT and UNPIVOT with Power BI

Turning columns to rows, or rows to columns is easy with Power Query and Power BI.

UNPIVOT → Converting Columns to Rows

PIVOT → Converting Rows to Columns

### Transpose

This is used to reverse the rows and column of a table. Once you select the column and click on transpose option the rows becomes columns and column becomes rows. For using it click on Transpose option of Transform tab as below.

The screenshot shows the Power Query ribbon with the 'Transform' tab selected. The 'Transpose' button is highlighted with a green box. The 'Queries [4]' pane shows four queries: 'Column\_Row', 'Transpose2', 'Unpivot1', and 'Transpose2 (2)'. The main table view shows a table with Region (East, West, North, South) in rows and Year 2009 through Year 2016 in columns.

Region values are in Rows and Year Values are in columns in the below Query and we will use this for Transpose.

	Region	Year 2009	Year 2010	Year 2011	Year 2012	Year 2013	Year 2014	Year 2015	Year 2016
1	East	49145	15805	63544	14588	27252	20240	28094	38449
2	West	33950	43735	66850	23125	10444	41878	41825	27325
3	North	49924	76718	96314	99194	94416	31575	75907	20659
4	South	53491	20462	99796	57194	29032	34539	62883	32993

Once after Transpose the result will looks like below. Year values in rows and Region values in columns.

	ABC 123 Year	ABC 123 East	ABC 123 West	ABC 123 North	ABC 123 South
1	Year 2009	49145	33950	49924	53491
2	Year 2010	15805	43735	76718	20462
3	Year 2011	63544	66850	96314	99796
4	Year 2012	14588	23125	99194	57194
5	Year 2013	27252	10444	94416	29032
6	Year 2014	20240	41878	31575	34539
7	Year 2015	28094	41825	75907	62883
8	Year 2016	38449	27325	20659	32993

## In built Row Transformations

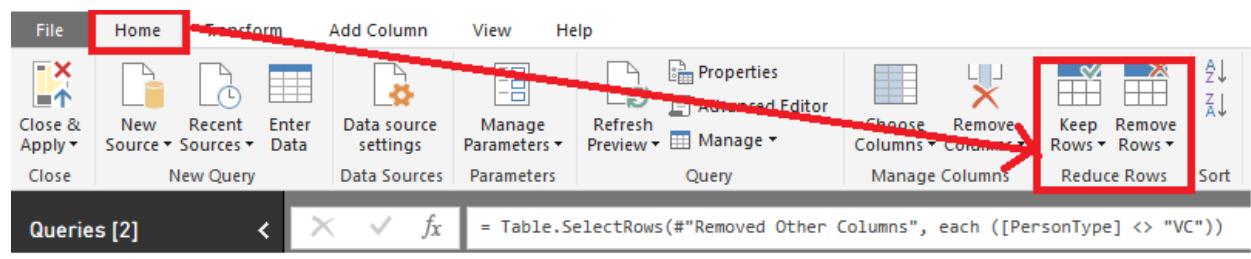
- ✓ Header Row or Use First Row as Headers
- ✓ Keep Top Rows
- ✓ Keep Bottom Rows
- ✓ Keep Range of Rows
- ✓ Keep Duplicates
- ✓ Keep Errors
- ✓ Remove Top Rows
- ✓ Remove Bottom Rows
- ✓ Remove Alternative Rows
- ✓ Remove Duplicates
- ✓ Remove Blank Rows
- ✓ Remove Errors
- ✓ Group Rows

## Use First Row as Header

When power bi is not able to identify headers automatically, you can manually do that using “Use First Row as Header”.

### Reduce Rows

In case you want to filter the data you are importing, you have two options: either by keeping the specific rows or removing rows. Both options can be found by clicking Home → Reduce Rows.



Under **Keep Rows**, you have the following options

**Keep Top Rows** → where you specify the number of top rows to keep.

**Keep Bottom Rows** → for which you pick the number of bottom rows to keep.

**Keep Range of Rows** → which skips a specified number of top rows and then keeps the chosen number of rows.

In addition to the first three options, which work on whole tables, you have **Keep Duplicates** and **Keep Errors**, both of which can work on either the whole table or the selected columns only. For example, if you select the whole table and choose Keep Duplicates, you will only see the rows that are complete duplicates of each other. However, if you choose only one column and click Keep Duplicates, you will get the rows where the values in the selected column are duplicates, regardless of other columns' values.

Under **Remove Rows**, you have six options

**Remove Top Rows** → Removes a specified number of top rows. Works on the whole table only.

**Remove Bottom Rows** → Removes a specified number of bottom rows. Works on the whole table only.

**Remove Alternate Rows** → Removes rows following a user-supplied pattern: it starts with a specified row, then alternates between removing the selected number of rows and keeping the chosen number of rows. Works on the whole table only.

**Remove Duplicates** → Removes rows that are duplicates of other rows. Works on either the whole table or the selected columns only.

**Remove Blank Rows** → Removes rows that completely consist of either empty strings or nulls; if you need to remove blank values from one column, you can click on the arrow to the right of a column's name and click **Remove Empty**. Remove Blank Rows Works on the whole table only.

**Remove Errors** → Removes rows that contain errors. Works on either the whole table or the selected columns only.

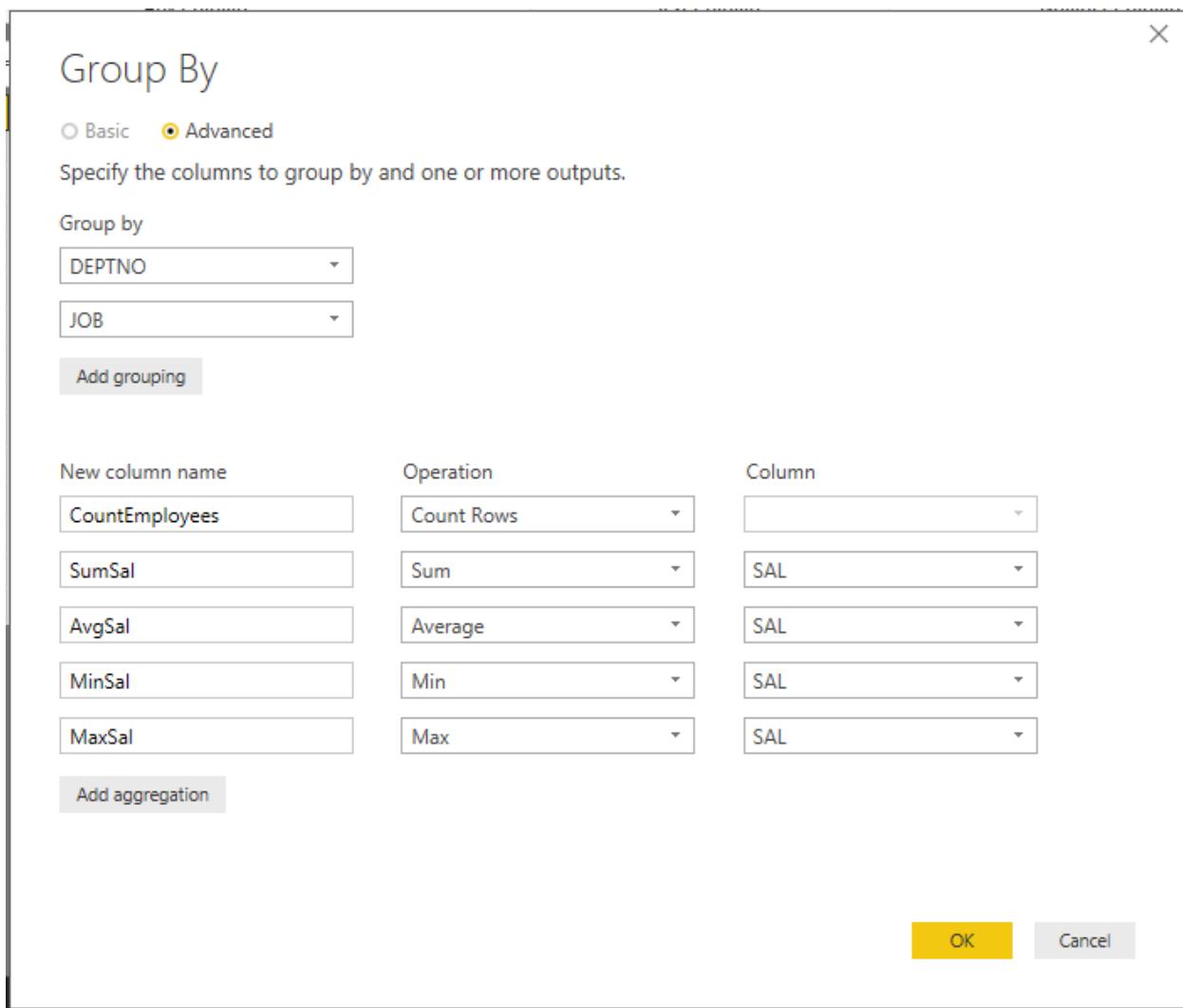
In case of Remove Duplicates and Remove Errors, there is a difference between applying these options to all selected columns or the whole table. In the first case, if you have new columns added to your query, the functions will not work on the new columns, because selecting all columns keeps their names in the code. To remove duplicates or errors from the whole table, select the table icon above row numbers and choose either Remove Duplicates or Remove Errors.

## Group rows / Group By

In Query Editor, you can group the values in multiple rows into a single value. This can be useful when summarizing the number of products offered, the total sales, or the count of students or total salary paid for each department.

Let's summarize our EMP table by Deptno and Job wise → TotSal, MinSal, MaxSal, AvgSal, CountOfEmp.

To do that, you can select Deptno and Job first, then click Home → Transform → Group By (Or) Transform → Table → Group By. The Group By window then opens; you'll see a radio button to switch between Basic and Advanced settings. Specify one or more columns to group by and how to aggregate data. To group by more than one column, switch to Advanced settings, or you could have pre-selected multiple columns before clicking Group By.



## Combine Queries

Combining more than one queries either horizontal or vertical we call it as Combine Queries. Result of a combine operation on one or more queries will be only one query. You can find Append Queries or Merge Queries in the Combine Queries section of the Query Editor in Power BI. Append Queries will combine the queries data vertically. Merge Queries will combine the queries data horizontally.

The screenshot shows the Power BI Query Editor interface. The top navigation bar has 'File', 'Home' (which is highlighted with a red box), 'Transform', 'Add Column', 'View', and 'Help'. Below the navigation bar are various icons for managing data sources and queries. On the right side, there's a 'QUERY SETTINGS' pane showing a query named 'Unpivot1'. A red box highlights the 'Combine' section in the ribbon, which includes options for 'Merge Queries', 'Append Queries', and 'Combine Files'.

## Append Queries

Append Queries means results of two (or more) queries will be combined vertically into one query in the below way

Rows will be appended after each other. (For example appending a query with 50 rows with another query with 100 rows, will return a result set of 150 rows)

Each Query should contain same number of columns and same datatype for columns for better outputs. (For example col1, col2... col10 in first query, after appending with same columns in the second query will result into one query with single set of col1, col2... col10)

### Append 2 files individually with different queries

Consider two sample data sets, one for each month sales,

#### Sales for April Month

The screenshot shows a table with the following data:

	TransDate	Account	Customer	Department	Amount
1	4/1/2016	76892	Darrin Van Huff		100
2	4/18/2016	53807	Lindsay Shagiari		105
3	4/26/2016	69735	Maria Bertelson		104

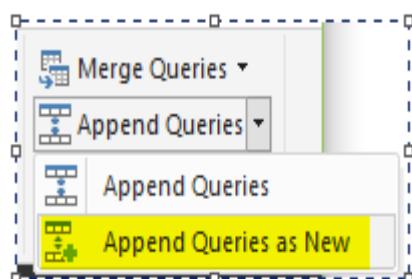
## Sales for May Month

	TransDate	Account	Customer	Department	Amount
1	5/1/2016	80596	Craig Carreira	107	3484
2	5/16/2016	72039	Ross Baird	115	3794
3	5/17/2016	55021	Troy Blackwell	101	901
4	5/29/2016	72216	Linda Cazamias	107	9437

To append these queries, click on one of them and select Append Queries from the Combine section of Home tab in Query Editor.

The screenshot shows the Power BI Query Editor interface. The ribbon at the top has tabs for Home, Transform, Add Column, View, and Help. Under the Home tab, there are several icons for managing queries, including Close & Apply, New, Recent, Enter, Data source settings, Manage Parameters, Refresh, Advanced Editor, Properties, and a dropdown for Data Type. The 'Combine' section contains options like Merge Queries, Append Queries (which is highlighted with a red box), and Combine Files. Below the ribbon, the 'Queries [6]' pane lists six queries: April 2016, May 2016, June 2016, July 2016, and two unnamed ones. The unnamed queries show sales data for April 2016. The right side of the screen displays the properties for the selected query, 'April 2016', including its name and other settings.

If you want to keep the existing query result as it is, and create a new query with the appended result choose Append Queries as New, otherwise just select Append Queries. In this example I'll do Append Queries as New, because I want to keep existing queries intact.



You can choose what is the primary table (normally this is the query that you have selected before clicking on Append Queries), and the table to append. In the append query result you will see primary table values first and others next.



For this example, I have only two tables, so I'll continue with above configuration. Append Queries simply append rows after each other, and because column names are exactly similar in both queries, the result set will have same columns.

The screenshot shows the Power Query Editor interface. On the left, the 'Queries [3]' pane lists three queries: 'April 2016', 'May 2016', and 'Append1'. The 'Append1' query is currently selected. The formula bar at the top displays the formula: `= Table.Combine({#"April 2016", #"May 2016"})`. The preview pane on the right shows the combined data from both tables. The columns are labeled: TransDate, Account, Customer, Department, and Amount. The data consists of 7 rows, with the first 6 rows coming from the 'April 2016' table and the last row coming from the 'May 2016' table. The preview pane has a dark background with light-colored text and borders.

	TransDate	Account	Customer	Department	Amount
1	4/1/2016	76892	Darrin Van Huff		100
2	4/18/2016	53807	Lindsay Shagiari		105
3	4/26/2016	69735	Maria Bertelson		104
4	5/1/2016	80596	Craig Carreira		3484
5	5/16/2016	72039	Ross Baird		115
6	5/17/2016	55021	Troy Blackwell		901
7	5/29/2016	72216	Linda Cazamias		9437

Result of Append as simple as below. Append is similar to UNION ALL in T-SQL.



#### What if files contain duplicates?

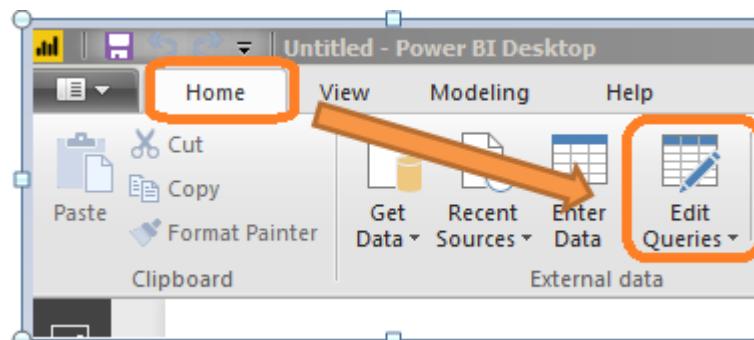
Append Queries will NOT remove duplicates. You have to use Group By or Remove Duplicate Rows to get rid of duplicates.

#### What if the Columns in the query are not matched exactly?

Append requires columns to be exactly similar to work in best condition. If columns in source queries are different, append still works, but will create one column in the output per each new column, if one of the sources doesn't have that column the cell value of that column for those rows will be null.

#### Append 3 or more files individually with different queries

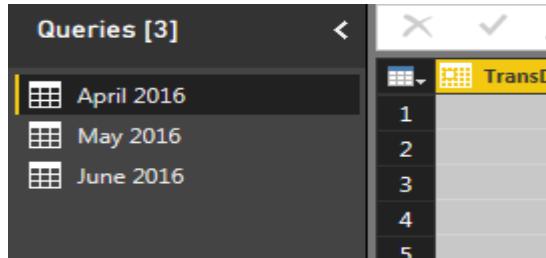
Open Edit Queries as shown below.



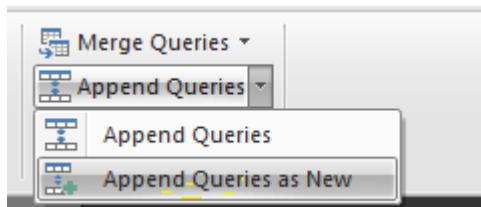
Query Editor Will opens → From Home Tab → selects New Source → Select the type of source for example Text/CSV as shown below → Browse for the file location → Select the file → Open.

Apply all steps above for each file to get multiple files into Query Editor for Appending.

Now all the three files are there in Query Editor as shown below. Now we will see how to append them.



Select Append Queries → Under select Append Queries as New



Select three or more tables → Add tables from Available tables section to Tables to append section



Change the order using up and down button as shown in above image, the way you are expecting the data in append query result → Ok. Now you will see an Append Query with the data from all the three files we selected in the same order as we placed in Tables to append section as shown below.

Queries [4]

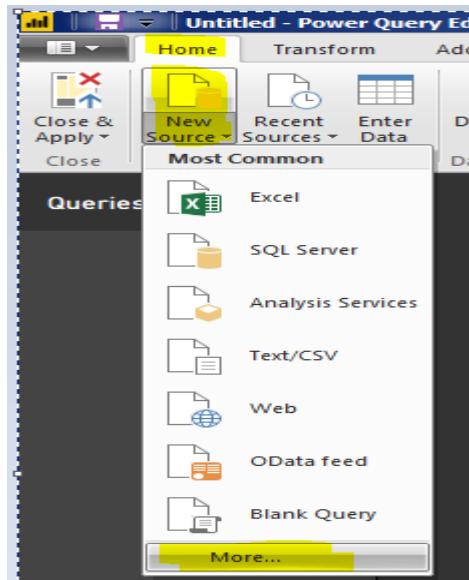
= Table.Combine({#"April 2016", #"May 2016", #"June 2016"})

	TransDate	Account	Customer	Department	Amount
1	4/1/2016	52148	Claire Gute	143	8770
2	4/1/2016	76892	Darrin Van Huff	100	6043
3	4/1/2016	62350	Sean O'Donnell	410	5597
4	4/2/2016	87606	Brosina Hoffman	355	8351
5	4/2/2016	48615	Andrew Allen	116	6347
6	4/3/2016	50615	Irene Maddox	357	5954
7	4/3/2016	57772	Harold Pawlan	179	2428
8	4/3/2016	76887	Pete Kriz	115	2404
9	4/4/2016	75877	Alejandro Grove	139	8089

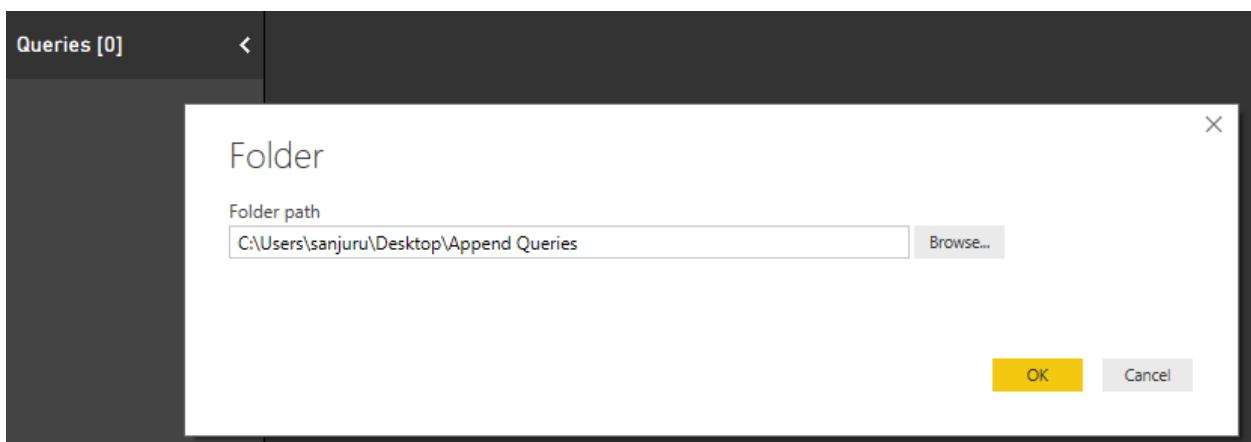
### Appending multiple files of same type from a folder using single query

In the above example we append April, May and June files. Tomorrow let us assume we get a file for July Month. In the above process you need to import it manually and need to append all the four files. It's a manual and lengthy process. Let's see how we can automate it by a single refresh and how to import and append with single query.

Open Query Editor → Home → New Source → More...



Select Folder → Connect → Enter the folder Path where files are located



Select Ok → Edit → Perform Transformation Steps as shown below.

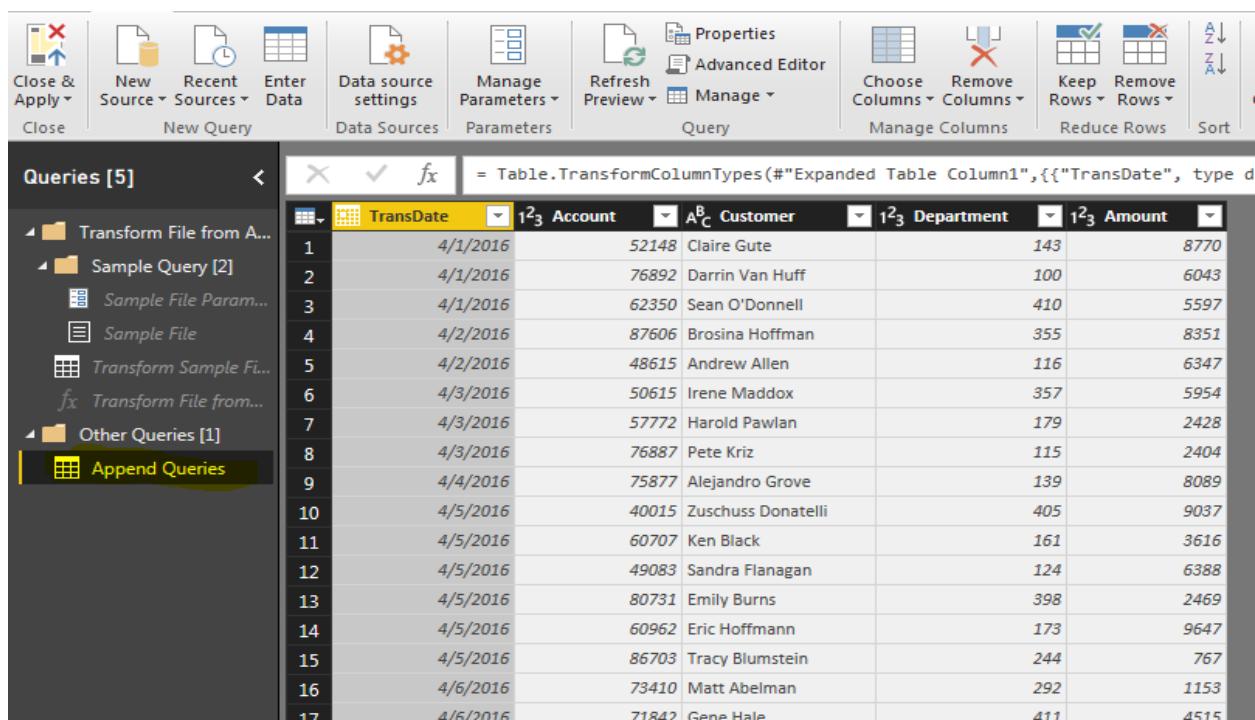
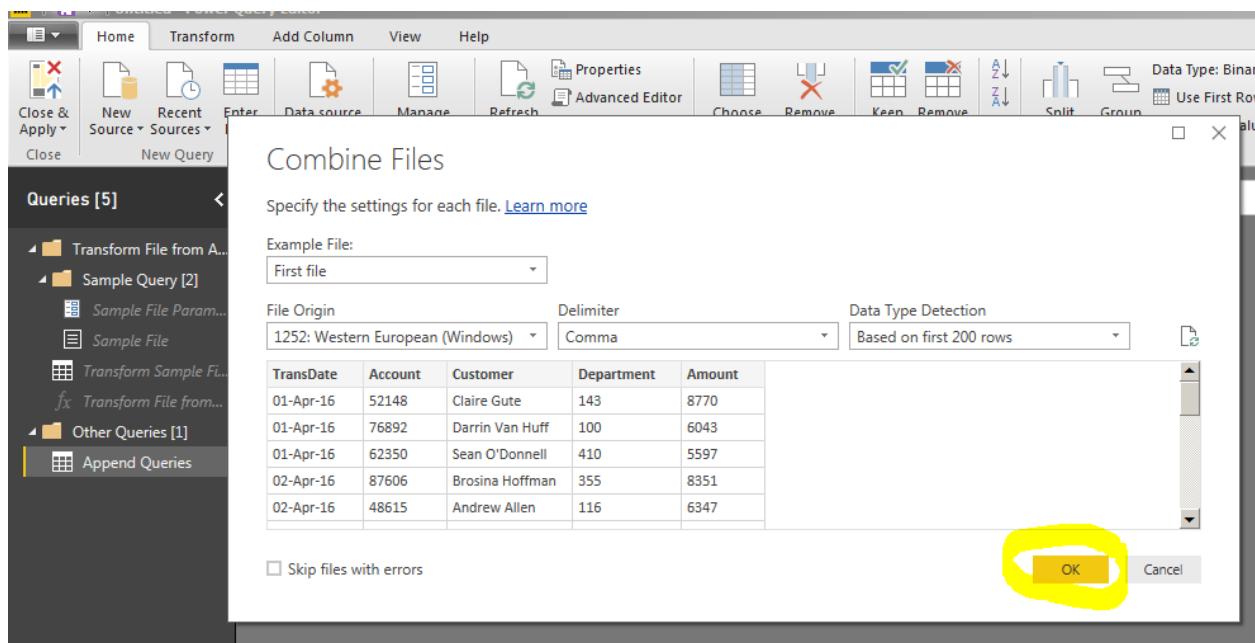
The screenshot shows the Power Query Editor interface with the title bar 'Untitled - Power Query Editor'. The ribbon includes Home, Transform, Add Column, View, and Help tabs. The 'Content' table contains four rows of binary data with dates. A context menu is open over the 'Name' column, with the option 'Remove Other Columns' highlighted.

	Content	Name
1	Binary	April 2016
2	Binary	July 2016
3	Binary	June 2016
4	Binary	May 2016

Select two down arrows to combine files as shown in below image

The screenshot shows the Power Query Editor interface with the title bar 'Untitled - Power Query Editor'. The ribbon includes Home, Transform, Add Column, View, and Help tabs. The 'Content' table contains six rows of binary data with file names. The 'Name' column has a dropdown arrow icon, indicating it is selected for transformation.

	Content	Name
1	Binary	April 2016.csv
2	Binary	Combine Files
3	Binary	April to June 2016.xlsx
4	Binary	July 2016.csv
5	Binary	July to Sep 2016.xlsx
6	Binary	June 2016.csv
		May 2016.csv



Tomorrow when you get a new file, place it in the folder and simply refresh in Query Editor. You will find updated data in Append Queries section.

## Task

Append the data from multiple excel sheets in an excel workbook and multiple workbooks or files in folder with a single query.

Add a Custom Column with the below formulae to read the content from excel workbook.

Excel.Workbook([Content])

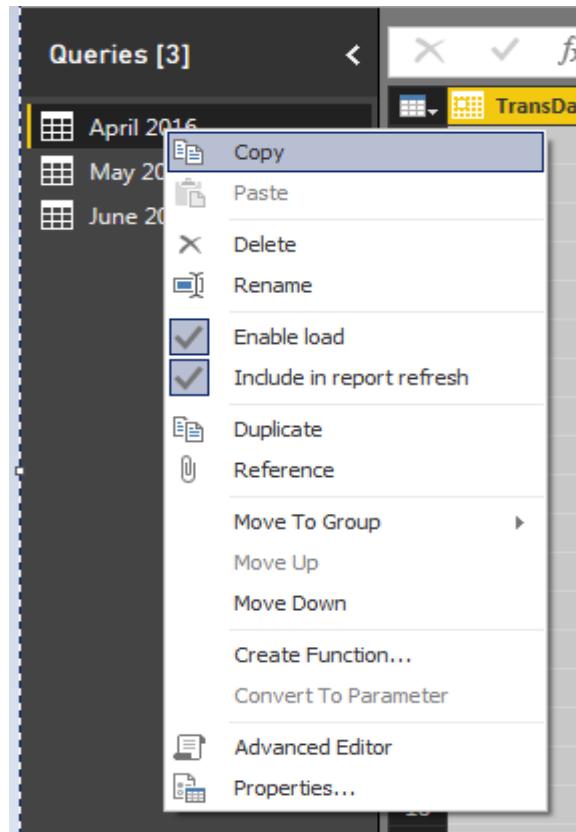
The screenshot shows the 'Custom Column' dialog box in Power BI Desktop. In the background, a table view shows two rows of binary files: 'April to June 2016.xlsx' and 'July to Sep 2016.xlsx'. The 'Content' column is highlighted. The dialog box itself has a title 'Custom Column'. It contains a 'New column name' field with 'Custom' typed in, a 'Custom column formula:' field containing '= Excel.Workbook([Content])', and a 'Available columns:' list on the right side showing 'Content' and 'Name'. At the bottom left is a link 'Learn about Power BI Desktop formulas', and at the bottom right are 'OK' and 'Cancel' buttons. A green checkmark icon and the text 'No syntax errors have been detected.' are displayed near the bottom left of the dialog.

	Content	Name	Custom
1	Binary	April to June 2016.xlsx	Table
2	Binary	July to Sep 2016.xlsx	Table

	Name	Custom
1	April to June 2016.xlsx	Table
2	July to Sep 2016.xlsx	Table

## Right-click options of a Query

When you right-click on the name of a query (in the left-most pane of Power Query's window) you will see list of options a show below. We will discuss each of them now.



### Copy

Copy option will copy the query and you can paste the query where ever you needed.

### Paste

Paste is used to paste the copied query under the queries section.

### Delete

Delete is used to delete a particular Query.

### Rename

Rename will help you to rename a query.

### Enable Load

Always remember that Query Editor is your ETL (Extract, Transform, and Load) engine. It will apply all transformations before loading the data into the model, but once you finished the transformation all queries will be loaded into the model and they take memory. By

default, all queries are enabled to Load into the model. Simply change that for queries that are not required in the model by disabling the option "Enable Load".

"Enable Load" means query results are available for report builder. Otherwise you may use it in your other queries (for example to merge data), but it is not shown in the report builder.

### Include in report refresh

"Include in Report Refresh" means query is automatically refreshed when you press "Refresh" button on the ribbon. Disable this option for the static tables to improve the performance of report refresh.

### Duplicate Query

Duplicate will duplicate the code of the query. Duplicate is generally used when you would like to create a similar query and you do not want to type the same code. Once you created a duplicate query from source query then duplicate query is independent of source query. You can make changes to this duplicate query.

### Reference Query

Reference Query means you would like to use that Query results in some other queries where your original query remains as a base Query. Once you created a reference query from source query then reference query is dependent of source query. When you make changes to source query those will reflect in reference query. You can make additional changes to this reference query.

The screenshot shows the Power BI Query Editor interface. On the left, there's a list of queries: 'Apr\_16', 'Jun\_1', and 'May\_1'. The 'Apr\_16' query is selected. A context menu is open over this query, listing several options: 'Copy', 'Paste', 'Delete', 'Rename', 'Enable load' (with a checked checkbox), 'Include in report refresh' (with a checked checkbox), 'Duplicate' (which is highlighted with a red box), and 'Reference'. Below these are 'Move To Group', 'Move Up', and 'Move Down' options. To the right of the menu, there's a preview of the data with columns 'L2 Account', 'Cust', 'Department', and 'Amount', and a list of 15 rows of data.

L2 Account	Cust	Department	Amount
ABC 123	ABC 123	ABC 123	ABC 123
52148	Claire Gute	143	8770
76892	Darrin Van Huff	100	6043
62350	Sean O'Donnell	410	5597
87606	Brosina Hoffman	355	8351
48615	Andrew Allen	116	6347
50615	Irene Maddox	357	5954
57772	Harold Pawlan	179	2428
76887	Pete Kriz	115	2404
75877	Alejandro Grove	139	8089
40015	Zuschuss Donatelli	405	9037
60707	Ken Black	161	3616
49083	Sandra Flanagan	124	6388
80731	Emily Burns	398	2469

## Merge Queries

Merge Queries are used when we need to add one or more columns to a Query from another Query. Merge is similar to JOINS in T-SQL or Other Databases.

Now we will see how to merge queries with an example. Take the EMP and DEPT Queries as shown below.

EMP

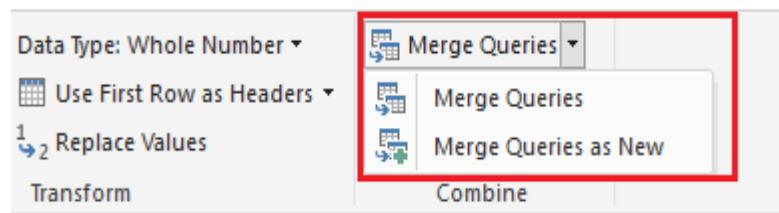
EMPNO	ENAME	JOB	MGR	SAL	DEPTNO
7369	SMITH	CLERK	1001	800	20
7499	ALLEN	SALESMAN	7566	1600	30
7521	WARD	SALESMAN	7566	1250	30
7566	JONES	MANAGER	7839	2975	20
7788	SCOTT	ANALYST	7839	3000	20
7839	KING	PRESIDENT		5000	10
7934	MILLER	CLERK	1001	1300	10
1001	SUNIL	TRAINER	7839	10000	50

DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

If you want to see employee details along with the department details in the same query then you need to merge EMP, DEPT Queries. Now we will see how to merge both of them.

Get both the Queries (EMP, DEPT) into Power Query. Select EMP Query first, and then select Merge Queries (Merge Queries as New) from the Home tab on the ribbon as shown below.



Next the Merge window appears prompting us to select the tables, matching columns and Join Kind to create a merged table as shown below. We will discuss more about Join Kind later.

Merge

Select tables and matching columns to create a merged table.

EMP					
EMPNO	ENAME	JOB	MGR	SAL	DEPTNO
7369	SMITH	CLERK	1001	800	20
7499	ALLEN	SALESMAN	7566	1600	30
7521	WARD	SALESMAN	7566	1250	30
7566	JONES	MANAGER	7839	2975	20
7788	SCOTT	ANALYST	7839	3000	20

DEPT		
DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Join Kind

Inner (only matching rows)

The selection has matched 7 out of the first 8 rows.

OK Cancel

Once you select tables, matching columns and Join Kind Then Ok button enabled. Once you press OK, a NewColumn is created at the end of the query as shown below. The NewColumn is the contents of the table (query) that was merged with the existing query. All columns from the merged query are condensed into the NewColumn.

	EMPNO	ENAME	JOB	MGR	SAL	DEPTNO	DEPT
1	7369	SMITH	CLERK	1001	800	20	Table
2	7566	JONES	MANAGER	7839	2975	20	Table
3	7788	SCOTT	ANALYST	7839	3000	20	Table
4	7499	ALLEN	SALESMAN	7566	1600	30	Table
5	7521	WARD	SALESMAN	7566	1250	30	Table
6	7839	KING	PRESIDENT	null	5000	10	Table
7	7934	MILLER	CLERK	1001	1300	10	Table

To Expand the merged table, and select which columns to include, select the expand icon (⊕). The Expand window appears as shown below.

The screenshot shows the 'Merge Columns' dialog box in Oracle SQL Developer. The dialog lists columns from the EMP and DEPT tables. The 'Select All Columns' checkbox is checked. The 'Use original column name as prefix' checkbox is unchecked. Buttons for OK and Cancel are at the bottom.

In this case, we only want the DNAME and LOC columns, so we select only that columns and then select OK. We clear the checkbox from Use original column name as prefix because we don't need or want that, if we leave that selected, the merged column would be named DEPT.DNAME, DEPT.LOC (the original column name, or NewColumn, then a dot, then the name of the column being brought into the query).

The screenshot shows the 'Merge Columns' dialog box in Oracle SQL Developer. The dialog lists columns from the EMP and DEPT tables. The 'Select All Columns' checkbox is checked. The 'Use original column name as prefix' checkbox is unchecked. Buttons for OK and Cancel are at the bottom.

Select OK to see the merged query output as shown below.

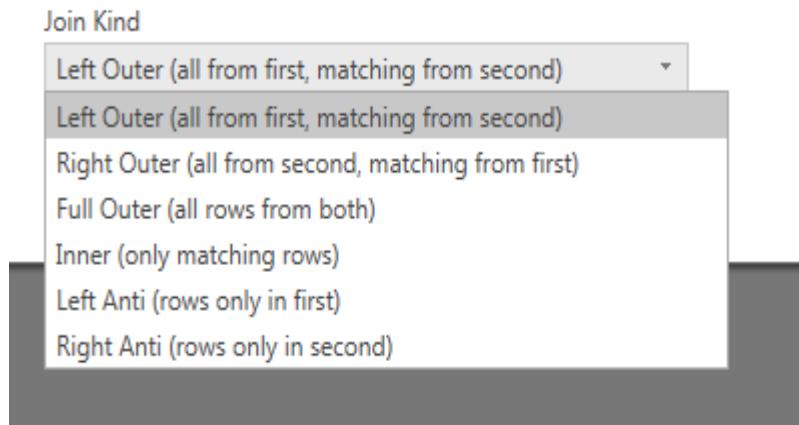
	$1^2_3$ EMPNO	$A^B_C$ ENAME	$A^B_C$ JOB	$1^2_3$ MGR	$1^2_3$ SAL	$1^2_3$ DEPTNO	$A^B_C$ DNAME	$A^B_C$ LOC
1	7369	SMITH	CLERK	1001	800	20	RESEARCH	DALLAS
2	7566	JONES	MANAGER	7839	2975	20	RESEARCH	DALLAS
3	7788	SCOTT	ANALYST	7839	3000	20	RESEARCH	DALLAS
4	7499	ALLEN	SALESMAN	7566	1600	30	SALES	CHICAGO
5	7521	WARD	SALESMAN	7566	1250	30	SALES	CHICAGO
6	7839	KING	PRESIDENT	null	5000	10	ACCOUNTING	NEW YORK
7	7934	MILLER	CLERK	1001	1300	10	ACCOUNTING	NEW YORK

You can easily use multiple columns in join condition for merging two data sets. Just select them in an order with holding Ctrl key of the keyboard.

### Types of Joins / Join Kinds / Merge Type

There are 6 types of joins you can perform for merging the queries by default as shown below. Each of these joins gives you different results in merge query output. Default Join Kind is Left Outer. Let's see what their difference is.

1. Left Outer (all from first, matching from second)
2. Right Outer (all from second, matching from first)
3. Full Outer (all rows from both)
4. Inner (only matching rows)
5. Left Anti (rows only in first)
6. Right Anti (rows only in second)



We use below EMP and DEPT Tables to illustrate these Join Types

## EMP

EMPNO	ENAME	JOB	MGR	SAL	DEPTNO
7369	SMITH	CLERK	1001	800	20
7499	ALLEN	SALESMAN	7566	1600	30
7521	WARD	SALESMAN	7566	1250	30
7566	JONES	MANAGER	7839	2975	20
7788	SCOTT	ANALYST	7839	3000	20
7839	KING	PRESIDENT		5000	10
7934	MILLER	CLERK	1001	1300	10
1001	SUNIL	TRAINER	7839	10000	50

## DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

## Left and Right

To start, you need to know the concept of Left and Right tables (or queries). When you merge two data sets with each other, the first query is considered as LEFT and the second as RIGHT.

## Merge

Select tables and matching columns to create a merged table.

EMP **Left**

EMPNO	ENAME	JOB	MGR	SAL	DEPTNO
7369	SMITH	CLERK	1001	800	20
7499	ALLEN	SALESMAN	7566	1600	30
7521	WARD	SALESMAN	7566	1250	30
7566	JONES	MANAGER	7839	2975	20
7788	SCOTT	ANALYST	7839	3000	20

DEPT **Right**

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Join Kind

Inner (only matching rows)

OK Cancel

In the above example EMP is Left and First and DEPT is Right and Second. Understanding this is important, because most of Join Kinds works with the concept of left or right or both.

### Left Outer (All from first, matching from second)

The first type of Join/Merge is Left Outer. All records from this query (LEFT or FIRST) will be showed in the result set plus their matching rows in the right (or second Query). This type of join is the default type. If you don't specify the Join Kind, it will be always Left Outer. The result set of EMP and DEPT with Join Kind Left Outer is shown below. We can see that all the Rows from EMP table are shown in the result. We don't have the DEPTNO=50 in DEPT table still it is showing in the merged query result as we select Left Outer Join Kind.

**Matching Rows from Both the Queries**

**Non Matching Rows from Left or First Query**

	EMPNO	ENAME	JOB	MGR	SAL	DEPTNO	DNAME	LOC
1	7369	SMITH	CLERK	1001	800	20	RESEARCH	DALLAS
2	7566	JONES	MANAGER	7839	2975	20	RESEARCH	DALLAS
3	7788	SCOTT	ANALYST	7839	3000	20	RESEARCH	DALLAS
4	7499	ALLEN	SALESMAN	7566	1600	30	SALES	CHICAGO
5	7521	WARD	SALESMAN	7566	1250	30	SALES	CHICAGO
6	7839	KING	PRESIDENT	null	5000	10	ACCOUNTING	NEW YORK
7	7934	MILLER	CLERK	1001	1300	10	ACCOUNTING	NEW YORK
8	1001	SUNIL	TRAINER	7839	10000	50	null	null

### Right Outer (all rows from second, matching from first)

With this type of Join, you get all rows from the RIGHT (or second) Query, with their matching rows from left (or first Query). We can see that all the Rows from DEPT Query are shown in the result along with matching rows from Both the Queries.

	EMPNO	ENAME	JOB	MGR	SAL	DEPTNO	DNAME	LOC
1	7369	SMITH	CLERK	1001	800	20	RESEARCH	DALLAS
2	7566	JONES	MANAGER	7839	2975	20	RESEARCH	DALLAS
3	7788	SCOTT	ANALYST	7839	3000	20	RESEARCH	DALLAS
4	7499	ALLEN	SALESMAN	7566	1600	30	SALES	CHICAGO
5	7521	WARD	SALESMAN	7566	1250	30	SALES	CHICAGO
6	7839	KING	PRESIDENT	null	5000	10	ACCOUNTING	NEW YORK
7	7934	MILLER	CLERK	1001	1300	10	ACCOUNTING	NEW YORK
8	null	null	null	null	null	null	OPERATIONS	BOSTON

### Full Outer (all rows from both)

This Join Kind will return all rows from both Queries (matching and non-matching). You will have all non-matching rows from first Query, and all non-matching rows from the second Query, and all matching rows from both the Queries. In the below image you can see Matching Rows from both the Queries in Green Box, Non matching rows from First or Left Query in Red Box and Non matching rows from Second or Right Query in Blue Box.

	EMPNO	ENAME	JOB	MGR	SAL	DEPTNO	DNAME	LOC
1	7369	SMITH	CLERK	1001	800	20	RESEARCH	DALLAS
2	7566	JONES	MANAGER	7839	2975	20	RESEARCH	DALLAS
3	7788	SCOTT	ANALYST	7839	3000	20	RESEARCH	DALLAS
4	7499	ALLEN	SALESMAN	7566	1600	30	SALES	CHICAGO
5	7521	WARD	SALESMAN	7566	1250	30	SALES	CHICAGO
6	7839	KING	PRESIDENT	null	5000	10	ACCOUNTING	NEW YORK
7	7934	MILLER	CLERK	1001	1300	10	ACCOUNTING	NEW YORK
8	1001	SUNIL	TRAINER	7839	10000	50	null	null
9	null	null	null	null	null	null	OPERATIONS	BOSTON

## Inner (only matching rows)

This Join Kind will only return matching rows as shown below. You will not have any record with null values (because these records generate as a result of not matching).

	EMPNO	ENAME	JOB	MGR	SAL	DEPTNO	DNAME	LOC
1	7369	SMITH	CLERK	1001	800	20	RESEARCH	DALLAS
2	7566	JONES	MANAGER	7839	2975	20	RESEARCH	DALLAS
3	7788	SCOTT	ANALYST	7839	3000	20	RESEARCH	DALLAS
4	7499	ALLEN	SALESMAN	7566	1600	30	SALES	CHICAGO
5	7521	WARD	SALESMAN	7566	1250	30	SALES	CHICAGO
6	7839	KING	PRESIDENT	null	5000	10	ACCOUNTING	NEW YORK
7	7934	MILLER	CLERK	1001	1300	10	ACCOUNTING	NEW YORK

All Rows are Matching Rows from Both the Queries

## Left Anti (rows only in first)

If you are only interested in rows from the LEFT (first) Query, then this is the option to select. This means rows that are in the First or Left Query and DO NOT match with the Second or Right Query. So this Join Kind returns only non-matching rows from the first or Left Query. With Anti options you always get null for the second data set, because these rows don't exist there. Anti-options are good for finding rows that exists in one Query but not in the other one.

This Join Kind will find the only one row that exists in the EMP Query and does not match with any of rows in the DEPT Query.

	EMPNO	ENAME	JOB	MGR	SAL	DEPTNO	DNAME	LOC
1	1001	SUNIL	TRAINER	7839	10000	50	null	null

Non-Matching rows from the first or Left Query

## Right Anti (rows only in second)

This Join Kind will give you only non-matching rows, this time from the Second (Right) Query. You can find out what rows in the Right or Second Query are not matching with the Left or First Query.

	EMPNO	ENAME	JOB	MGR	SAL	DEPTNO	DEPTNO.1	DNAME	LOC
1	null	null	null	null	null	null	40	OPERATIONS	BOSTON

Non-Matching Rows from the  
Second OR Right Query

## Cartesian Join or Cross Join

By Default, in Power BI we have 6 types of joins and in that we don't have one more type of join that is Cross Join or Cartesian Join. To get the cross join output we have a work around that we need to add a custom column for both the queries and populate that column with a unique value.

## Inbuilt Column Transformations

- Remove Columns / Remove Other Columns
- Choose Columns, Go To Column
- Name / Rename a Column
- Reorder Columns or Sort Columns - NoOfDays
- Column from Examples

Order Date(Year), Customer ID (Text Before -), CustomerName(First 5 Chars)

(GUI Options , Column from Examples, Custom Column)

- Add Column / Custom Column
- Duplicate Column

When you want Output in Same Column use any of the Options in Transform Tab

When you want Output in the New Column use the Options in Add Column Tab

Split Columns - Mostly we need to Split the Columns Based on Delimiters or Based on Number of Characters

Merge Columns

Pivot - Rows to Columns - Table Format Data to Pivot Format - Number of Rows will Reduce

Unpivot - Columns to Rows - If You Want to Convert Pivot Format Data to Table Format - Number of Rows will Increase

Transpose - Rows to Columns & Columns to Rows

Replace Values

Any arithmetic Operation with Null is NULL

Remove Empty

+++++

## In built Row Transformations

- Header Row or Use First Row as Headers
- Use Headers as First Row
- Keep Top Rows - Whole Table
- Keep Bottom Rows - Whole Table
- Keep Range of Rows - Whole Table
- Keep Errors - whole table or selected columns
- Remove Errors - whole table or selected columns
- Remove Blank Rows - Whole Table
- Remove Empty - Down Arrow - Selected Column
- Keep Duplicates - whole table or selected columns
- Remove Duplicates - whole table or selected columns
- Remove Top Rows - Whole Table
- Remove Bottom Rows - Whole Table
- Remove Alternative Rows - Whole Table
- Group Rows / Group By

1. Few options will work on Complete Table
2. Few Options work on Complete Table as well as a specific Column

## Introduction to DAX

### What is DAX?

DAX stands for **Data Analysis eXpressions**, and it is the formula language used in Power BI. DAX is also found in other offerings from Microsoft, such as Power Pivot and SSAS Tabular.

DAX is a functional language, which means the full executed code is contained inside a function.

There are two primary calculations you can create using DAX

- Calculated Columns
- Measures

DAX 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. Stated more simply, DAX helps you create new information from data already in your model.

### DAX Table and Column Name syntax

Whether you're creating a New Column or Measure, it's important to know the general format of table names in DAX.

**'Table Name'[ColumnName]**

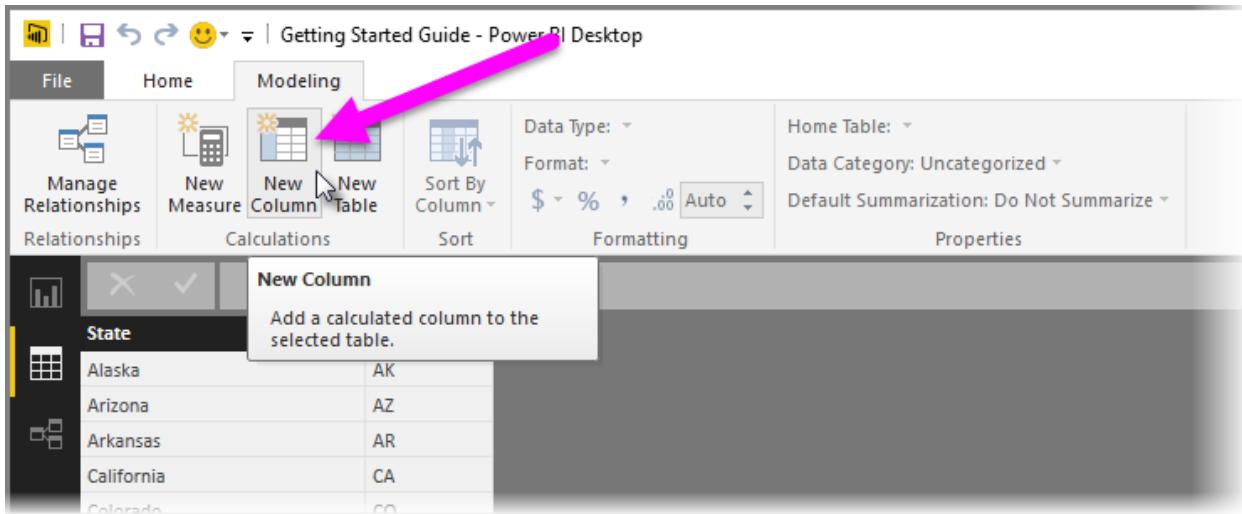
If there are spaces in the table name (as shown above), the single quotes around the table name are mandatory. If the table name has no spaces, the single quotes can be omitted, so the syntax looks like the following. Column names must always include the square brackets.

**TableName[ColumnName]**

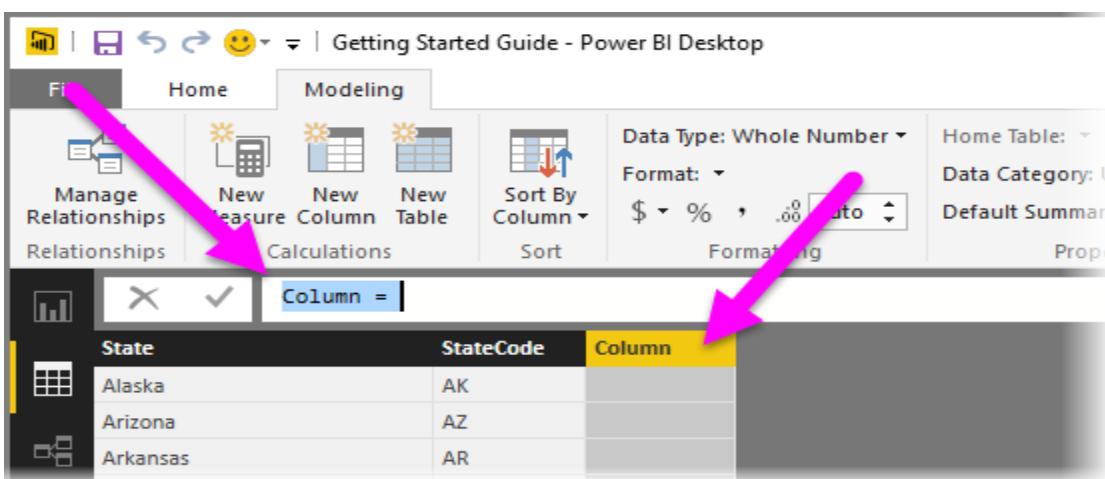
### Creating Calculated Columns

Calculated Columns are useful when you want a calculation for every row in your table.

You can create Calculated Columns in Power BI Desktop by selecting **New Column** from the Modeling tab. It's best to be in Data view (rather than Report or Relationships view), since you can see the **New Column** created and the Formula Bar is populated and ready for your DAX formula.



Once you select the New Column button, the Formula Bar is populated with a basic column name (which you change to suit your formula, of course) and the = operator, and the new column appears in the data grid, as shown in the following image.



The required elements for a calculated column are the following

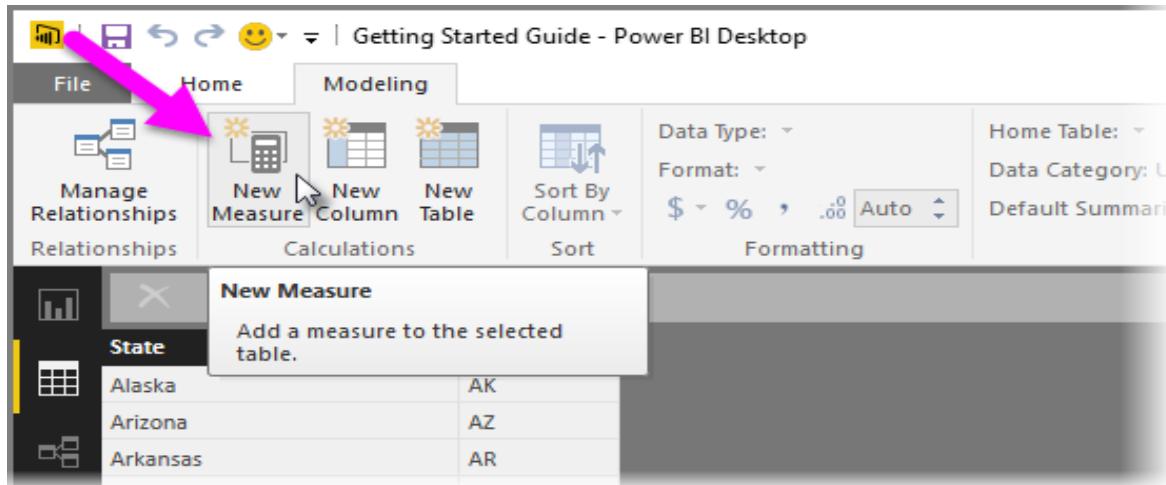
- A new column name
- At least one function or expression

### Calculated Columns Examples

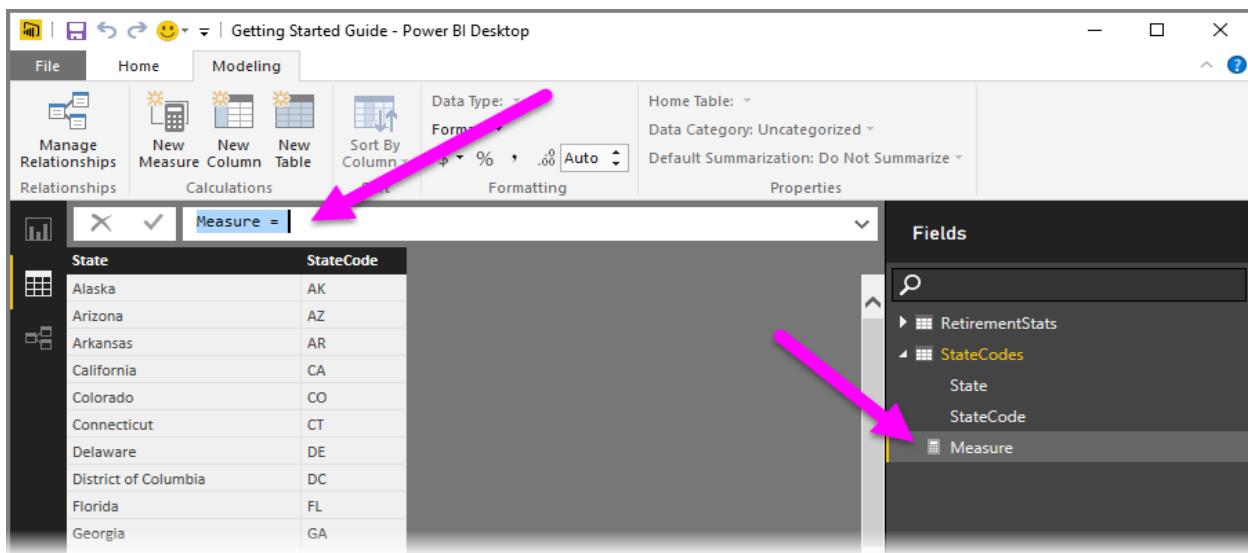
1. Load the Orders table into Power BI
2. Create the following Calculated Columns
  - a) Unit Price = Sales / Quantity
  - b) Cost Price = Sale - Profit
  - c) Quantity Type = IF (Orders [Quantity] = 1, "Single Item", "Multiple Items")
  - d) State and Country = CONCATENATE (Orders [State], CONCATENATE (", ", Orders [Country]))

## Creating Measures

Use a calculated measure when you are calculating percentages or ratios, or you need complex aggregations. To create a measure using a DAX formula, select the New Measure button from the Modeling tab. Again, it's best to be in the Data view of Power BI Desktop since it shows the Formula Bar and makes it easy to write your DAX formula.



With measures, you see a new measure icon appear in the Fields pane with the name of the measure. The Formula Bar is again populated with the name of your DAX formula (this time, with your measure).



The required elements for a calculated measure are the same as they are for a calculated column

- A new measure name
- At least one function or expression

## Calculated Measures Examples

- Total Sales = SUM(Orders[Sales])
- Total Profit = SUM(Orders[Profit])
- Average Sales = AVERAGE(Orders[Sales])

## Implicit Vs Explicit Measures

Implicit Measures are created when you drag raw numerical fields (like Sales, Profit, Quantity) into the values field well of a visual and manually select the aggregation mode (Sum, Average, Min/Max, etc.).

Explicit measures are created by actually entering DAX functions like below

- Sum Of Sales = SUM(Orders[Sales])

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

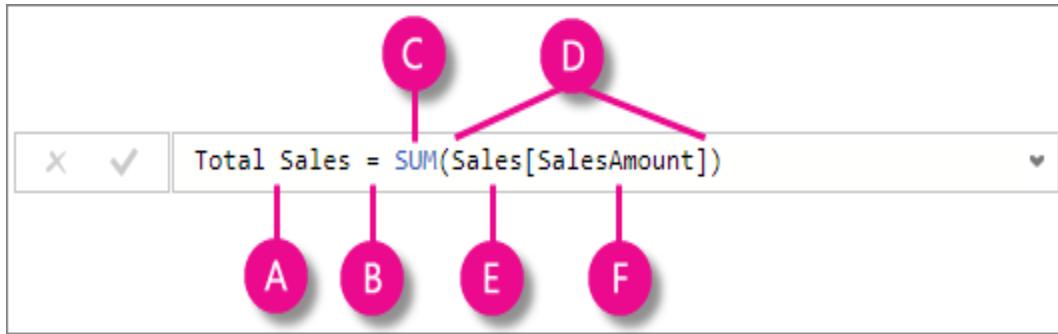
Explicit measures can be used anywhere in the report and referenced with other DAX calculations.

- Sum Of Sales = SUM (Orders[Sales])
- % Sales = [Sum Of Sales]/CALCULATE([Sum Of Sales],ALL(Orders))

## Calculated Columns / New Column Vs New Measures

Calculated Columns / New Column	New Measures
For Each Row when you need a Value	To Calculate Aggregates and Percentages. (OR) For Group of Rows if you need one Value
Values are calculated based on information for each row of a Table. Row Context.	Values are calculated based on information from any filters in the report or the fields used in the visualizations. Filter Context.
Appends Static values to each row in a table and stores them in the model which increases the file size	Does not create new data in the tables which doesn't increase the file size
Recalculate on data source refresh or when changes are made to component columns which are used to derive Calculated Columns	Recalculate in response to any change to filters within the report or when we Add / Remove another dimension to the visualization.
New Column Name Should be unique at Table Level	Measure Name Should be unique throughout the Model

## DAX Syntax & Operators



This formula includes the following syntax elements

- A) The measure name Total Sales.
- B) The equals sign operator (=) indicates the beginning of the formula. When calculated, it will return a result.
- C) The DAX function SUM adds up all of the numbers in the Sales [SalesAmount] column. You'll learn more about functions later.
- D) Parenthesis () surround an expression containing one or more arguments. All functions require at least one argument. An argument passes a value to a function.
- E) The referenced table Sales.
- F) The referenced column [SalesAmount] in the Sales table. With this argument, the SUM function knows on which column to aggregate a SUM.

## DAX Operators

The Data Analysis Expression (DAX) language uses operators to create expressions that compare values, perform arithmetic calculations, or work with strings.

### Types of Operators

There are four different types of operators → arithmetic, comparison, text concatenation, and logical.

### Arithmetic Operators

To perform basic mathematical operations such as addition, subtraction, or multiplication, combine numbers, and produce numeric results, use the following arithmetic operators.

Arithmetic operator	Meaning	Example
+ (plus sign)	Addition	3+3
- (minus sign)	Subtraction or sign	3-1
* (asterisk)	Multiplication	3*3
/ (forward slash)	Division	39/3
^ (caret)	Exponentiation	16^4

## Comparison Operators

You can compare two values with the following operators. When two values are compared by using these operators, the result is a logical value, either TRUE or FALSE.

Comparison operator	Meaning	Example
=	Equal to	[Region] = "USA"
>	Greater than	[Sales Date] > "Jan 1 2009"
<	Less than	[Sales Date] < "Jan 1 2009"
>=	Greater than or equal to	[Amount] >= 20000
<=	Less than or equal to	[Amount] <= 100
<>	Not equal to	[Region] <> "USA"

## Text Concatenation Operator

Use the ampersand (&) to join, or concatenate, two or more text strings to produce a single piece of text.

Text operator	Meaning	Example
& (ampersand)	Connects, or concatenates, two values to produce one continuous text value	[State] & ", " & [Country]

EX:State and Country = CONCATENATE (Orders [State], CONCATENATE (", ", Orders [Country]))

Or

State and Country = Orders [State] & ", " & Orders [Country]

## Logical Operators

Use logical operators (`&&`) and (`||`) to combine expressions to produce a single result.

Logical operator	Meaning	Example
<code>&amp;&amp;</code> (double ampersand)	Creates an AND condition between two expressions that each have a Boolean result. If both expressions return TRUE, the combination of the expressions also returns TRUE; otherwise the combination returns FALSE.	<code>([Region] = "France") &amp;&amp; ([BikeBuyer] = "yes")</code>
<code>  </code> (double pipe symbol)	Creates an OR condition between two logical expressions. If either expression returns TRUE, the result is TRUE; only when both expressions are FALSE is the result FALSE.	<code>(([Region] = "France")    ([BikeBuyer] = "yes"))</code>
<code>IN</code>	Creates a logical OR condition between each row being compared to a table. Note: the table constructor syntax uses curly braces.	<code>'Product'[Color] IN { "Red", "Blue", "Black" }</code>

## DAX Functions

### Functions in DAX / DAX Functions Categories

DAX provides the following types or Categories of functions which we mostly used.

- Date and Time Functions
- Logical Functions
- Text Functions
- Math & Statistical Functions
- Filter Functions
- Time Intelligence Functions

### Basic Date & Time Functions

Mostly we used Date and Time functions to create Calculated Columns / New Columns. Let's us discuss about below date functions.

- YEAR
- MONTH
- DAY
- WEEKDAY
- WEEKNUM
- FORMAT (Text Function) → Month Name, Weekday Name
- DATE
- TODAY
- NOW
- HOUR
- MINUTE
- SECOND
- TIME
- DATEDIFF
- CALENDAR
- EOMONTH
- STARTOFTMONTH

The screenshot shows the Microsoft Power BI Data Editor interface. The ribbon at the top has the 'Modeling' tab selected. Below the ribbon, there are several icons: Manage Relationships, New Measure, New Column, New Table, New Parameter, What If, Sort by Column, Sort, Data type, Format, and a dollar sign icon. In the foreground, a context menu is open over a column in a table. The menu items listed are: [Date], [Day], [Month], [MonthNo], [Quarter], [QuarterNo], and [Year]. The table itself contains columns for Category, Sub-Category, and Product Name, with some rows visible.

## Creating Calculated Columns from Date Fields

Here we will create fields for Year, Month and Day.

- a) Year = Year (Orders [Order Date])
- b) Month = Month (Orders [Order Date])
- c) Day = Day (Orders [Order Date])
- d) Week Day = Weekday (Orders [Order Date])
- e) WeekNum = WEEKNUM (Orders [Order Date])

### Create the Month Name field

Month Name = Format (Orders [Order Date], "MMMM") → Gives full Month Name

Month Name = Format (Orders [Order Date], "MMM") → Gives Month Name abbreviations

### Create a Weekday Name field

Weekday Name = FORMAT (Orders [Order Date], "DDDD") → Gives Full Weekday Name

Weekday Name = FORMAT (Orders [Order Date], "DDD") → Gives Weekday Name abbreviations

### DATEDIFF

No of Days = DATEDIFF (Orders [Order Date], Orders [Ship Date], DAY)

(Select don't summarize for No of Days to get correct results)

## Calculated Tables

In a calculated table, the table values are generated by Data Analysis Expression (DAX) and the values are stored in the Power BI model. Usually using Calculated Tables, we will create Date Dimension table and use this table in the model for time series analysis.

### Date Dimension Table

Every Time creating this date fields for different data sets is difficult, so what we do is we create a Date Dimension Table which contains all these fields and use it.

### DAX CALENDAR Function

The calendar function returns a table with a single column that contains a continuous set of dates. The start and end date range will be supplied as parameters.

The following formula returns a calculated table **Date\_Dim** with dates between January 1st, 2011 and December 31st, 2020.

Date\_Dim = CALENDAR (DATE (2011, 1, 1), DATE (2020, 12, 31))

From the above **DateColumn**, we will be deriving below columns.

- Day = DAY(Date\_Dim[Date])
- WeekDay = WEEKDAY(Date\_Dim[Date])
- WeekDayName = FORMAT(Date\_Dim[Date], "DDDD")
- Month = MONTH(Date\_Dim[Date])
- MonthName = FORMAT(Date\_Dim[Date], "MMMM")
- Quarter = ROUNDUP((Date\_Dim[Month]/3),0)
- Quarter = IF (Month Number<4,"Qtr 1", IF (Month Number<7,"Qtr 2", IF (Month Number<10,"Qtr 3", "Qtr 4")))
- QuarterName = "Qtr" & Date\_Dim[Quarter]
- QuarterName = CONCATENATE ("Qtr", Date\_Dim[Quarter])
- Year = YEAR(Date\_Dim[Date])
- WeekNumber = WEEKNUM(Date\_Dim[Date])
- WeekNumMonth = 1 + WEEKNUM (Dim\_Date[Date]) - WEEKNUM(STARTOFMONTH(Dim\_Date[Date]))
- WeekNameMonth = "Week" & " " & Date\_Dim[WeekNumMonth]

### **Week of Month**

The week number of the month is, one plus the difference between the weeknum for the date and the weeknum of the first of the month.

WeekOfMonth =1+ WEEKNUM (Dim\_Date[Date]) - WEEKNUM(STARTOFMONTH(Dim\_Date[Date]))

### **EOMONTH**

EOMONTH Returns the date in datetime format of the last day of the month, before or after a specified number of months. Use EOMONTH to calculate maturity dates or due dates that fall on the last day of the month.

#### **Syntax**

EOMONTH (<start\_date>, <months>)

#### **Example**

1. If customer taken a loan for 12 months, to find when the loan will complete use below syntax.

EOM = EOMONTH (Orders [Order Date], 12)

2. Based on Order Date filed in Orders table, get the Number of Days in that month.

No of Days in Month = DAY ((EOMONTH (Orders [Order Date], 12)))

## DAX Logical Functions

Microsoft Power BI DAX provides various **Logical Functions** as shown below

- IF
- TRUE
- FALSE
- NOT
- OR
- IN
- AND
- IFERROR
- SWITCH

Now we will see how to use Power BI DAX Logical Functions with examples.

### IF Function

The DAX IF function is used to check the given expression is True or False. The basic syntax of the Power BI DAX If Function is as shown below

#### Syntax

```
IF (LogicalTest, ResultIfTrue, [ResultIfFalse])  
IF (Expression, True_Info, False_Info)
```

As you can see from the above syntax, this function accepts three arguments. First argument is the Boolean expression (which return true or false). If the expression results TRUE then second argument will return otherwise, third argument will return.

Syntax in Square Brackets is optional. If you are not providing third argument, then if the condition is not satisfied it will return NULL values.

#### Examples

If Example = IF(Orders[Quantity]>1,"Multiple Products", "Single Product")

### Nested IF Function

In Power BI DAX, you can use the Nested If concept, One If statement inside another. Below statement will check whether the Sales amount of each column is less than 100 or not. If true then column will return "Low Sales" otherwise, it will enter into Nested If.

#### Examples

Nested If Example = IF(Orders[Sales]<100,"Low Sales", IF(Orders[Sales]>300, "High Sales", "Avg Sales"))

## DAX TRUE & False Function

The DAX **TRUE** function will return a logical value **True**. The DAX **FALSE** function will return logical **False**.

True and False Example = `IF(Orders[Sales]>100, TRUE (), FALSE ())`

## DAX NOT Function

The DAX **NOT** function will convert True to False, and False to True. I mean, it returns opposite result. The syntax of the Power BI DAX NOT Function is as shown below

`NOT(Condition)`  
`NOT(<logical>)`

### Example

NOT example = `NOT(IF(Orders[Sales]>100, TRUE (), FALSE ()))`

## DAX OR Function

The DAX **OR** function is like either or statement in English, which is used to check multiple expressions. The syntax of the Power BI DAX OR Function is as shown below.

`OR (Condition 1, Condition 2)`  
`OR (Logical1, Logical2)`

### Example

Weekday = `WEEKDAY (Orders [Order Date])`

Weekend = `IF(OR(Orders[Weekday]=1, Orders[Weekday]=7), "Weekend", "Weekday")`

Weekend = `IF(Orders[Weekday]=1 || Orders[Weekday]=7, "Weekend", "Weekday")`

## DAX IN Function

IN Example = `IF(Orders[Weekday] IN {2,3,4,5,6}, " Weekday ", Weekend")`

## DAX AND Function

The DAX **AND** function is used to check multiple expressions. The syntax of the Power BI DAX AND Function is as shown below

`AND (Condition 1, Condition 2)`  
`AND (Logical1, Logical2)`

As you can see from the above syntax, DAX **AND** function accepts two arguments. If both the conditions are True then it will return True otherwise, it return False.

## Examples

And Example = IF(AND(Orders[Category] = "Office Supplies", Orders[Sub-Category] = "Storage"), TRUE(), FALSE())

And example = IF(Orders[Category] = "Office Supplies" && Orders[Sub-Category] = "Storage", TRUE(), FALSE())

## DAX IFERROR Function

The DAX IFERROR function is very useful to handle the arithmetic overflow, or any other errors. It simply performs calculation and return the result, if there is an error then it will return the value inside the second argument. The syntax of the DAX IFERROR Function is as shown below.

IFERROR (Calculation, Value\_If\_Error\_Occurs)

Below statement will return 100 if error occurs. Indeed, all the records will throw error because we are dividing them 0. If there is no error, it will give calculation or expression value.

Error Example = IFERROR(Orders[Sales]/0, 100)

## DAX SWITCH Function

The DAX SWITCH function helps you to return multiple options. For example, IF statement will return either True or false. However, you can use this switch case to multiple results. The syntax of the DAX SWITCH Function is as shown below.

SWITCH (Expression, Option 1, Result 1, Option 2, Result 2, ...., Else Result)

If Month of Order Date is 1 then below statement will return January, 2 means February, 3 means March, 4 means April, 5 means May, 12 means December otherwise, Unknown.

Switch Example = SWITCH (MONTH (Orders [Order Date]), 1, "January", 2, "February", 3, "March", 4, "April", 5, "May", 12, "December", "Unknown")

Switch Example = SWITCH (TRUE (), Orders [Sales]<100, "Low Sales", Orders[Sales]>300, "High Sales", "Medium Sales")

SWITCH () always test for equivalence. SWITCH () is still testing for equivalence! By providing the first argument as TRUE (), now each subsequent "test" is going to be checking for TRUE (). And since each of our inequality tests results in either TRUE () or FALSE () as a value, the test case that evaluate to TRUE () is the one that gets matched, and therefore the one that gets used. For instance, if [Measure] <1 evaluates to TRUE (), then expr1 gets returned.

## Text Functions in DAX

### LEN

LEN function will Returns the Number of characters in string.

#### Syntax

LEN (<Text>)

#### Example

Length = LEN (Orders [Customer Name])

### CONCATENATE (&)

CONCATENATE function joins two text strings into one text string.

#### Syntax

CONCATENATE (<Text1>, <Text2>)

#### Example

Concat = CONCATENATE (Orders [Category], CONCATENATE (" - ", Orders [Sub-Category]))

Concatenate = Orders [Category] & " - " & Orders [Sub-Category]

### LEFT

LEFT Function Returns the specified number of characters from the start of a text string.

#### Syntax

LEFT (<text>, <num\_chars>)

num\_chars optional, if omitted, 1.

#### Example

Left = LEFT (Orders [Country], 5)

### RIGHT

RIGHT function returns the last character or characters in a text string, based on the number of characters you specify.

#### Syntax

RIGHT (<text>, <num\_chars>)

num\_chars optional, if omitted, 1.

## **Example**

Right = RIGHT (Orders [Country], 5)

## **MID**

MID Function 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>)

## **Example**

Mid = MID (Orders [Country], 8, 5)

## **UPPER**

UPPER Function Converts a text string to all uppercase letters

## **Syntax**

UPPER (<text>)

## **Example**

Upper = UPPER (Orders [Country])

## **LOWER**

Lower Function Converts all letters in a text string to lowercase.

## **Syntax**

LOWER (<text>)

## **Example**

Lower = LOWER (Orders [Country])

## **TRIM**

TRIM Function Removes all spaces from text except for single spaces between words.

## **Syntax**

TRIM (<text>)

## **Example**

Trim = TRIM (Orders [Customer Name])

## **SUBSTITUTE**

SUBSTITUTE Function Replaces existing text with new text in a text string.

### **Syntax**

SUBSTITUTE (<text>, <old\_text>, <new\_text>, <instance\_num>)

instance\_num optional, if omitted, every instance of old\_text is replaced with new\_text.

### **Example**

Substitute = SUBSTITUTE (Orders [Country], "t", "T") - Replace existing text with new text for all occurrences

Substitute = SUBSTITUTE (Orders [Country], "t", "T", 2) - Replace existing text with new text for second occurrence only.

## **BLANK**

Returns a blank.

### **Syntax**

BLANK ()

### **Example**

The following example illustrates how you can work with blanks in formulas. The formula calculates unit price. However, before attempting to calculate the ratio the denominator should be checked for zero values. If the denominator is zero then a blank value should be returned, otherwise, the ratio is calculated.

Blank = IF (Orders[Quantity] = 0, BLANK (), Orders[Sales]/ Orders[Quantity])

### **Usage**

Avoids 'Divide by zero' error.

## Math and Statistical Functions

### INT

INT Rounds a number down to the nearest integer.

#### Syntax

INT (<number>)

#### Example

INT Example = INT (Orders [Sales])

12.34 =12                  12.78 = 12                  12.55=12

### ROUND

Round function will Rounds a number to the given number of digits.

#### Syntax

ROUND (<number>, <num\_digits>)

#### Example

Round Example = ROUND (Orders [Sales], 0)

12.34 =12                  12.78 = 13                  12.55=13

Round Example = ROUND (Orders [Sales], 1)

12.34 =12.3                  12.78 = 12.8                  12.55=12.6

### ROUNDUP

Roundup will Rounds a number to next integer value if num\_digits = 0.

#### Syntax

ROUNDUP (<number>, <num\_digits>)

#### Example

ROUNDUP Example = ROUNDUP (Orders [Sales], 0)

12.34 =13                  12.78 = 13                  12.55=13

ROUNDUP Example = ROUNDUP (Orders [Sales], 1)

12.34 =12.4                  12.78 = 12.8                  12.55=12.6

## ROUNDDOWN

ROUNDDOWN Rounds a number down, toward zero.

### Syntax

ROUNDDOWN (<number>, <num\_digits>)

### Example

ROUNDDOWN Example = ROUNDDOWN (Orders [Sales], 0)

12.34 =12                  12.78 = 12                  12.55=12

ROUNDDOWN Example = ROUNDDOWN (Orders [Sales], 1)

12.34 =12.3                  12.78 = 12.7                  12.55=12.5

## DIVIDE

Performs division and returns alternate result or BLANK () on division by 0.

### Syntax

DIVIDE (<numerator>, <denominator>, [<alternateresult>])

### Example

Divide Example = DIVIDE (Orders [Sales], Orders [Quantity], 0)

Divide Example = DIVIDE (200,4,0) = 50

Divide Example = DIVIDE (100,0, -1) = -1

## EVEN

Returns number rounded up to the nearest even integer.

### Syntax

EVEN (number)

### Example

Even Example = EVEN (Orders [Sales])

12.34 =14                  12.78 = 14                  12.55=14

13.34 =14                  13.78 = 14                  13.55=14

## ODD

Returns number rounded up to the nearest odd integer.

### Syntax

ODD (number)

### Example

Odd Example = ODD (Orders [Sales])

12.34 =13                  12.78 = 13                  12.55=13

13.34 =15                  13.78 = 15                  13.55=15

## POWER

Returns the result of a number raised to a power.

### Syntax

POWER (<number>, <power>)

### Example

Power Example = POWER (Orders [Quantity], Orders [Quantity])

Power Example = POWER (Orders [Quantity],2)

Power Example = POWER (4,2) →16

Power Example = POWER (3,3) →27

## SIGN

SIGN determines the sign of a number, the result of a calculation, or a value in a column. The function returns 1 if the number is positive, 0 (zero) if the number is zero, or -1 if the number is negative.

### Syntax

SIGN (<number>)

### Example

Sign Example = SIGN (Orders [Sales])

## SQRT

SQRT returns the square root of a number. If the number is negative, the SQRT function returns an error.

### Syntax

SQRT (<number>)

### Example

SQRT Example = SQRT (Orders [Quantity])

SQRT Example = ROUND (SQRT (Orders [Quantity]), 2)

## FACT

Returns the factorial of a number, equal to the series  $1*2*3*...$ , ending in the given number.

### Syntax

FACT (<number>)

### Example

FACT Example = FACT (Orders [Quantity])

## SUM

SUM Function Adds all the numbers in a column.

### Syntax

SUM (<column>)

### Example

Sum of Sal = SUM (EMP [SAL])

## SUMX

Returns the sum of an expression evaluated for each row in a table.

### Syntax

SUMX (<table>, <expression>)

SUMX of Sal = SUMX (EMP, EMP [SAL] +EMP [COMM])

## **MIN**

Returns the smallest numeric value in a column.

### **Syntax**

MIN (<column>)

### **Example**

Min of Sal = MIN(EMP[SAL])

## **MINX**

Returns the smallest numeric value that results from evaluating an expression for each row of a table.

### **Syntax**

MINX (<table>, <expression>)

### **Example**

MINX of Sal = MINX (EMP, EMP [SAL]+ EMP [COMM])

## **MAX**

Returns the largest numeric value in a column.

### **Syntax**

MAX (<column>)

### **Example**

MAX of Sal = MAX(EMP[SAL])

## **MAXX**

Evaluates an expression for each row of a table and returns the largest numeric value.

### **Syntax**

MAXX (<table>, <expression>)

### **Example**

MAXX of Sal = MAXX (EMP, EMP [SAL]+ EMP [COMM])

Sum Of Sal	SumX of Sal	Min of Sal	MinX of Sal	MAX of Sal	MAXX of Sal
29025	31225	800	800	5000	5000

DEPTNO	Sum Of Sal	SumX of Sal	Min of Sal	MinX of Sal	MAX of Sal	MAXX of Sal
10	8750	8750	1300	1300	5000	5000
20	10875	10875	800	800	3000	3000
30	9400	11600	950	950	2850	2850
<b>Total</b>	<b>29025</b>	<b>31225</b>	<b>800</b>	<b>800</b>	<b>5000</b>	<b>5000</b>

## COUNT

Counts the number of cells in a column that contain numbers.

### Syntax

COUNT (<column>)

### Example

COUNT Example = COUNT(EMP[SAL])

Count Example1 = COUNT (EMP [COMM])

## COUNTX

Counts the number of rows that contain a number or an expression that evaluates to a number, when evaluating an expression over a table.

### Syntax

COUNTX (<table>, <expression>)

### Example

COUNTX Example = COUNTX (EMP, EMP[SAL]+EMP[COMM])

## AVERAGE

AVERAGE Function Will Returns the average of all the numbers in a column.

### Syntax

AVERAGE (<column>)

## Example

Average Example = AVERAGE(EMP[SAL])

Average of COMM = AVERAGE(EMP[COMM])

## AVERAGEX

Calculates the average of a set of expressions evaluated over a table.

### Syntax

AVERAGEX (<table>, <expression>)

## Example

AVERAGEX Example = AVERAGEX (EMP, EMP[SAL]+EMP[COMM])

Sum Of Sal	COUNT Example	Count Example1	COUNTX Example	Average Example	Average of Comm	AVERAGEX Example
29025	14	4	14	2,073.21	550.00	2,230.36

DEPTNO	Sum Of Sal	SumX of Sal	COUNT Example	Count Example1	COUNTX Example	Average Example	Average of Comm	AVERAGEX Example
10	8750	8750	3		3	2,916.67		2,916.67
20	10875	10875	5		5	2,175.00		2,175.00
30	9400	11600	6	4	6	1,566.67	550.00	1,933.33
Total	29025	31225	14	4	14	2,073.21	550.00	2,230.36

## COUNTROWS

Counts the number of rows in the specified table, or in a table defined by an expression.

### Syntax

COUNTROWS (<table>)

## **Example**

COUNTROWS Example = COUNTROWS(EMP)

## **COUNTBLANK**

Counts the number of blank cells in a column.

### **Syntax**

COUNTBLANK (<column>)

## **Example**

COUNTBLANK Example = COUNTBLANK(EMP[COMM])

## **RANKX**

The RANKX function is used in DAX to create rankings.

Ranks allow you to easily compare products, salespeople or anything else that you want to evaluate the performance.

RANKX is a scalar function and it is also an iterator. The RANKX function can optionally take a Value argument that represents a scalar value whose rank is to be found. The optional Order argument specifies how to rank Value, descending (0) or ascending (1). The optional Ties argument defines how to determine ranking when there are ties (Same Ranks). Skip (default) will use the next rank value after a tie, and Dense will use the next rank value (i.e. there will be no gaps in the rank numbers).

Optional arguments might be skipped by placing an empty comma (,) in the argument list, i.e. RANKX = RANKX (EMP, EMP[SAL],,,Dense)

### **Syntax**

RANKX (TABLE, EXPRESSION [, VALUE] [, ORDER] [, TIES]...)

Dense - will not Skip Between Rank Numbers

### **Examples**

RANKX = RANKX (EMP, EMP[SAL])

Category Rank =

-- STEP 3. Rank my expression against the sorted list from 2.

RANKX (

-- STEP 1. Loop the rows in this table

ALL(EMP[DEPTNO]),

```
-- STEP 2. Run this expression for each loop
CALCULATE (
    SUM(EMP[SAL])
)
)
```

## SUMMARIZE

Summarize Functions Returns a "summary table or Aggregate Table" for the requested totals over a set of groups.

```
DeptJobSAL =
SUMMARIZE(emp,Dept[DEPTNO],Emp[JOB],"CountOfEmp",COUNT(Emp[EMPNO]),"SumOfSal",SUM(Emp[SAL]))
```

## CALCULATE

**CALCULATE** Function evaluates a given expression or formula under a set of defined filters. We Know that in many visualizations, we want to show only a sub set of data instead of creating visualizations and filtering we will filter the measure value using **CALCULATE** Function.

### Syntax

```
CALCULATE (<expression>, <filter1>, <filter2>...)
```

### Examples

SumOfSales = SUM (Orders [Sales])

EastRegionSales = CALCULATE ([SumOfSales], Orders [Region] ="East")

EastCentralSales=

```
CALCULATE ([SumOfSales], Orders [Region] ="East" || Orders [Region] ="Central")
```

EastTechSales=

```
CALCULATE ([SumOfSales], Orders [Region] ="East", Orders [Category] ="Technology")
```

Between each filter condition it will use AND operation.

## ALL

**ALL** Function returns all the rows in a table, or all the values in a column, ignoring any filters that might have been applied.

This function is useful for clearing filters and creating calculations on all the rows in a table or column.

### Syntax

```
ALL ({<table> | <column> [, <column> [, <column> [...]]]})
```

<Table> | <column>

The “table or column” that you want to clear filter on. If you want you can add multiple columns to clear filter.

## Examples

%Sales = [SumOfSales] /CALCULATE ([SumOfSales], ALL (Orders [Category]))

% Sales = [SumOfSales] /CALCULATE ([SumOfSales], ALL (Orders))

Achieving %Sales is a reasonably easy task using simple implicit measures in Power BI as shown in below.

Drag Category, Sales twice into Visuals. The Visual will give Below Output.

Category	Sales	Sales
Furniture	7,41,999.80	7,41,999.80
Office Supplies	7,19,047.03	7,19,047.03
Technology	8,36,154.03	8,36,154.03
<b>Total</b>	<b>22,97,200.86</b>	<b>22,97,200.86</b>

To get %Sales Go to the Field wells and Click on Expand button as shown below.

The screenshot shows the Power BI Field井 (Field Well) on the right side of the interface. In the 'Values' section, there are three entries: 'Category', 'Sales', and 'Sales'. The second 'Sales' entry has a circled expand icon (a small triangle) next to it. A red box surrounds the second 'Sales' entry, and a red arrow points from the text 'To get % Sales' to this expand icon.

By Default as shown in below image "Show Value as" Selected as "No Calculation", so that it will result Default "Sum of Sales".

Change "Show Value as" to "Percent of grand total" to get expected % Sales results

Output

## ALLSELECTED

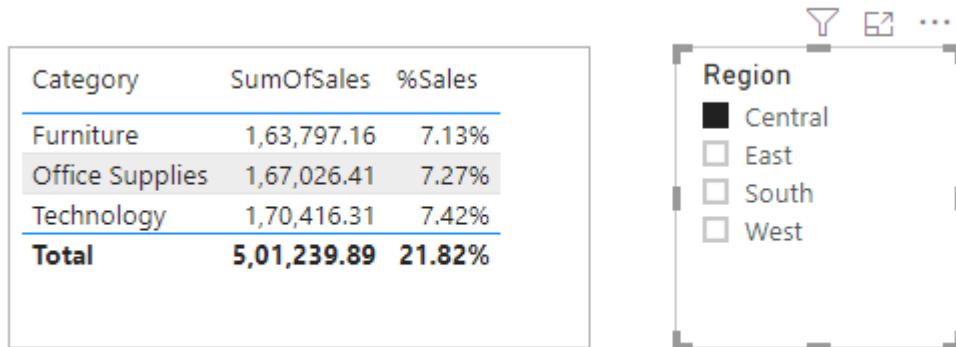
**ALLSELECTED** will return all the rows in a table, or all the values in a column, ignoring any filters that might have been applied inside the query or Visualization, but keeping filters that come from outside slicer.

$\%Sales = [SumOfSales]/CALCULATE ([SumOfSales], ALL (Orders))$

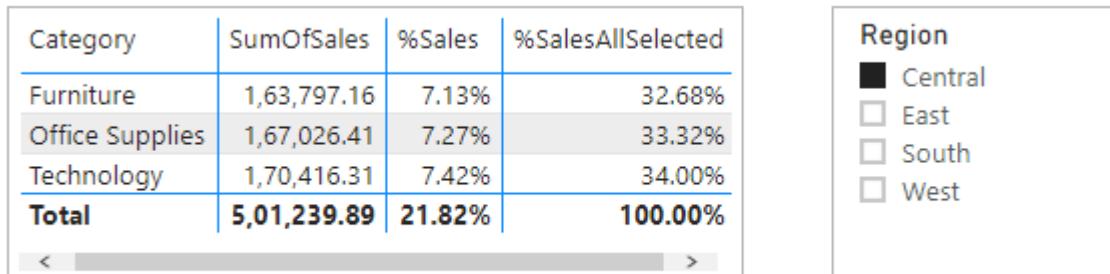
Category	SumOfSales	%Sales
Furniture	7,41,999.80	32.30%
Office Supplies	7,19,047.03	31.30%
Technology	8,36,154.03	36.40%
<b>Total</b>	<b>22,97,200.86</b>	<b>100.00%</b>

When You Filter the Visual using the Slicer based on Region Column, you will see the below results.

%Sales = [SumOfSales]/CALCULATE ([SumOfSales], ALL (Orders))



%SalesAllSelected =  
[SumOfSales]/CALCULATE (SUM (Orders [Sales]), ALLSELECTED (Orders))



## ALLEXCEPT

Removes all context filters in the table except filters that have been applied to the specified columns.

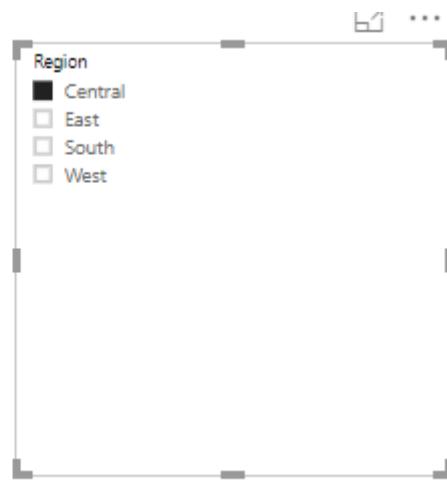
1. Calculate Category wise %Sales

%Sales = [SumOfSales]/CALCULATE ([SumOfSales], ALL (Orders))

Category	SumOfSales	%Sales
Furniture	741,999.80	32.30%
Office Supplies	719,047.03	31.30%
Technology	836,154.03	36.40%
<b>Total</b>	<b>2,297,200.86</b>	<b>100.00%</b>

2. Calculate Category wise %Sales for each Region

Category	SumOfSales	%Sales
Furniture	163,797.16	7.13%
Office Supplies	167,026.41	7.27%
Technology	170,416.31	7.42%
<b>Total</b>	<b>501,239.89</b>	<b>21.82%</b>



%Sales = [SumOfSales]/CALCULATE ([SumOfSales], ALLEXCEPT (Orders, Orders [Region]))

Category	SumOfSales	%Sales
Furniture	163,797.16	32.68%
Office Supplies	167,026.41	33.32%
Technology	170,416.31	34.00%
<b>Total</b>	<b>501,239.89</b>	<b>100.00%</b>

## RELATED

Returns a related value from another table.

The RELATED function requires that a relationship exists between the current table and the table with related information. You specify the column that contains the data that you want, and the function follows an existing **many-to-one** relationship to fetch the value from the specified column in the related table. If a relationship does not exist, you must create a relationship.

### Syntax

RELATED (<column>)

### Example

Person = RELATED(People[Person])

## \*\*\*USERELATIONSHIP

Specifies the relationship to be used in a specific calculation as the one that exists between columnName1 and columnName2.

Sum of Sales Ship Date = CALCULATE(SUM(Orders[Sales]), USERELATIONSHIP(Dim\_Date[Date], Orders [Ship Date]))

## NATURALINNERJOIN

## NATURALLEFTOUTERJOIN

## CROSSJOIN

## CALCULATETABLE

## RELATEDTABLE

## LOOKUPVALUE

LOOKUPVALUE Function will Returns the value in result\_columnName for the row that meets all criteria specified by search\_columnName and search\_value.

### Syntax

`LOOKUPVALUE (<result_columnName>, <search_columnName>, <search_value> [, <search_columnName>, <search_value>]... [, <alternateResult>])`

`DName = LOOKUPVALUE (Dept [DNAME], Dept [DEPTNO], Emp [DEPTNO])`

The value will return for result\_column for the Rows where all pairs of search\_column and search\_value have a match.

If there is no match a BLANK will return when You Don't Supply Alternative Result, if supplied Alternative Result will be returned.

If multiple rows match the search values and in all cases result\_column values are identical then that value is returned.

However, if result\_column returns different values an error or alternateResult, if supplied, is returned.

To get Related Value using LOOKUPVALUE function we no need to Have Relationship B/W Tables.

In LOOKUPVALUE function we can write Multiple Conditions

`QtySold =  
LOOKUPVALUE(Sales[QtySold],Sales[Category],Product[Category],Sales[ProductCode],Product[  
ProductCode])`

CALCULATE

CALCULATE Function Evaluates an expression in a context modified by filters.

CALCULATETABLE

Evaluates a table expression in a context modified by filters.

FILTER

Returns a table that has been filtered.

## FILTER

The FILTER function is used to return a subset of a table or expression.

```
CentralOrders = FILTER(Orders, Orders[Region] = "Central")
```

Above Formula will create a New table with only Central Region Orders.

In many cases you can use the CALCULATE function instead of the FILTER function to produce the same results (the resulting formula is usually easier to understand, too).

```
Filter = CALCULATE(SUM(Orders[Sales]), FILTER(orders, Orders[Region] = "South" || Orders[Segment] = "Home Office"))
```

## Time Intelligence

### What is Time Intelligence?

Time Intelligence means doing calculations over periods of time or dates. All the Time Intelligence Functions will need a date column to perform the calculations, and this date column should contain **unique**, **no null** and **contiguous** date values to get the accurate results.

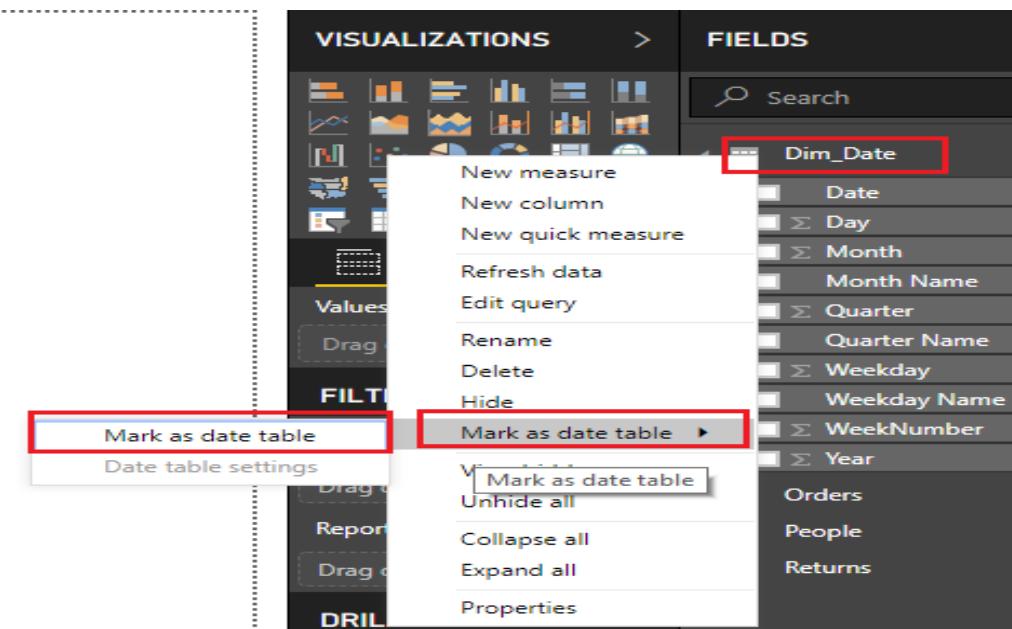
Time Intelligence simply means doing BI calculations over periods of time, or over dates. For example, a common request in many reports is to show a value, such as Sales, aggregated month-by-month, from the beginning of each year, so that you know, in this case, what were the total sales, in that year, up to the given month. DAX, Data Analysis Expressions, the language of Power BI, includes many predefined time intelligence functions, including **TOTALYTD**, which is exactly the function you need to use in this particular scenario.

Many data analysts prefer to create their own date tables, which is fine. In Power BI Desktop, you can specify the table you want your model to use as its date table, and subsequently create date-related visuals, tables, measures, and so on, using that table's date data. When you specify your own date table, you control the date hierarchies created in your model, and use them in measures and other operations that use your model's date table.

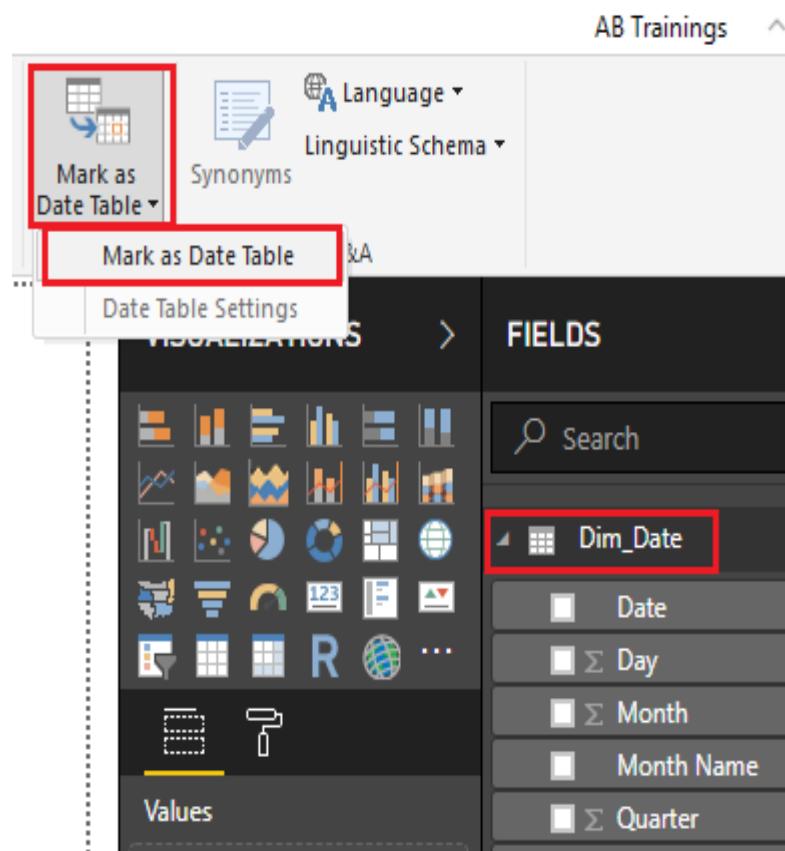
To make DAX Time Intelligence functions work properly set date dimension table as "Mark as date table".

### Setting your own date table

To set a date table select the table you want to use as a date table in the Fields pane, then right-click the table and select **Mark as date table** → **Mark as date table** in the menu that appears, as shown in the following image.



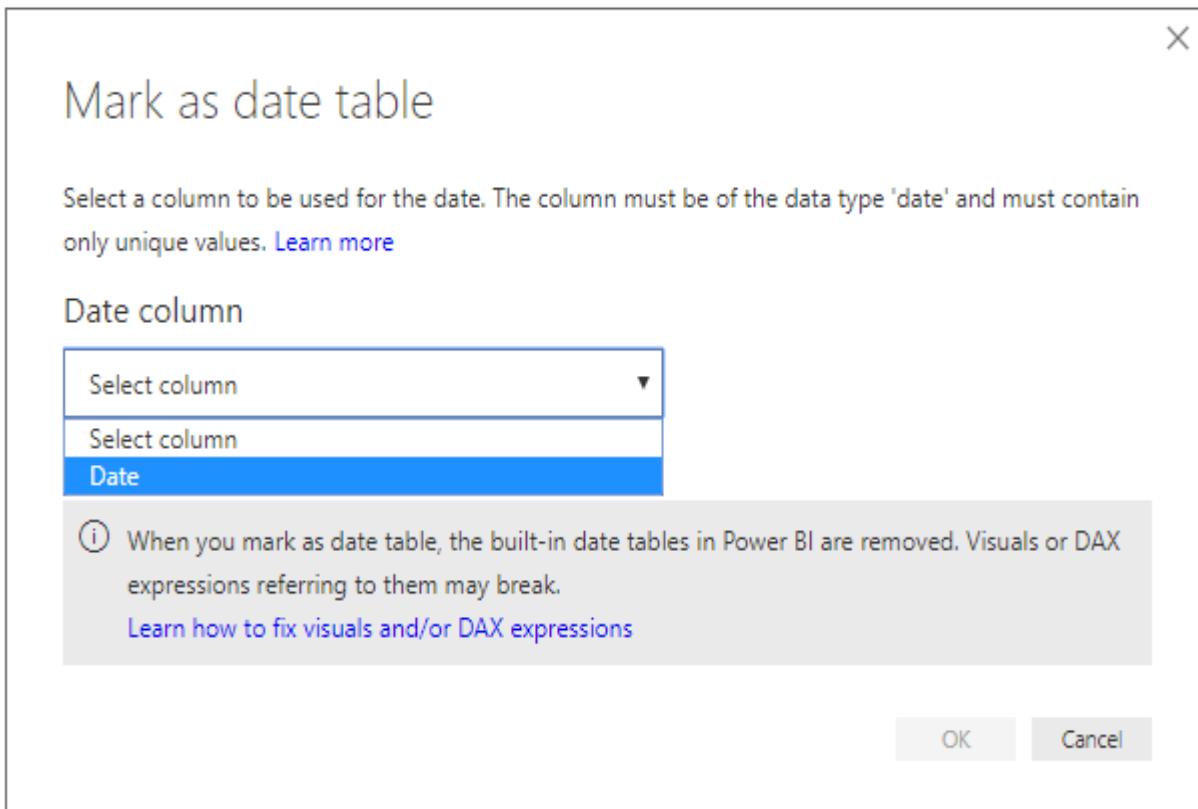
You can also select the table and then select **Mark as Date Table** from the Modeling ribbon, shown here.



When you specify your own date table, Power BI Desktop performs the following validations of that column and its data, to ensure that the data

- contains unique values
- contains no null values
- contains contiguous date values (from beginning to end)
- if it is a Date/Time data type, it has the same timestamp across each value

Once you specify a date table, you can select which column in that table is the date column. You can specify which column to use by selecting the table in the Fields pane, then right-click the table and select **Mark as date table** → **Date table settings**. The following window appears, where you can select the column to use as the date table from the drop-down box.



## DAX Time Intelligence Functions

All the Time Intelligence Functions will need a **date column** to perform the calculations, and this date column should contain **unique, no null** and **contiguous** date values to get the accurate results.

### TOTALMTD

Evaluates the value of the expression for the month to date, in the current context. TOTALMTD will give Running Totals or cumulative sum for each Month.

#### Example

Total MTD = TOTALMTD(SUM(Orders[Sales]), Dim\_Date[Date])

### TOTALQTD

Evaluates the value of the expression for the dates in the quarter to date, in the current context. TOTALQTD will give Running Totals or cumulative sum for each Quarter.

#### Example

Total QTD = TOTALQTD(SUM(Orders[Sales]), Dim\_Date[Date])

## TOTALYTD

Evaluates the year-to-date value of the expression in the current context. TOTALYTD will give Running Totals or cumulative sum for each year.

### Example

Total YTD = TOTALYTD(SUM(Orders[Sales]), Dim\_Date[Date])

Year	Quarter Name	Month Name	Day	Sales	Total YTD Ex	Total QTD Ex	Total MTD
2014	Qtr1	January	3	16.45	16.45	16.45	16.45
2014	Qtr1	January	4	288.06	304.51	304.51	304.51
2014	Qtr1	January	5	19.54	324.04	324.04	324.04
2014	Qtr1	January	6	4,407.10	4,731.14	4,731.14	4,731.14
2014	Qtr1	January	7	87.16	4,818.30	4,818.30	4,818.30
2014	Qtr1	January	8		4,818.30	4,818.30	4,818.30
2014	Qtr1	January	9	40.54	4,858.85	4,858.85	4,858.85
2014	Qtr1	January	10	54.83	4,913.68	4,913.68	4,913.68
2014	Qtr1	January	11	9.94	4,923.62	4,923.62	4,923.62
2014	Qtr1	January	12		4,923.62	4,923.62	4,923.62
2014	Qtr1	January	13	3,553.80	8,477.41	8,477.41	8,477.41
2014	Qtr1	January	14	61.96	8,539.37	8,539.37	8,539.37
2014	Qtr1	January	15	149.95	8,689.32	8,689.32	8,689.32
2014	Qtr1	January	16	299.96	8,989.29	8,989.29	8,989.29
2014	Qtr1	January	17		8,989.29	8,989.29	8,989.29
2014	Qtr1	January	18	64.86	9,054.15	9,054.15	9,054.15
2014	Qtr1	January	19	378.59	9,432.74	9,432.74	9,432.74
2014	Qtr1	January	20	2,673.87	12,106.61	12,106.61	12,106.61
2014	Qtr1	January	21	25.25	12,131.86	12,131.86	12,131.86
2014	Qtr1	January	22		12,131.86	12,131.86	12,131.86
2014	Qtr1	January	23	46.02	12,177.88	12,177.88	12,177.88
<b>Total</b>				<b>2,297,200.86</b>			

## PREVIOUSDAY

Returns a table that contains a column of all dates representing the day that is previous to the first date in the dates column, in the current context.

### Example

Previous Day = CALCULATE(SUM(Orders[Sales]), PREVIOUSDAY(Dim\_Date[Date]))

## **PREVIOUSMONTH**

Returns a table that contains a column of all dates from the previous month, based on the first date in the dates column, in the current context.

### **Example**

Previous Month = CALCULATE(SUM(Orders[Sales]), PREVIOUSMONTH(Dim\_Date[Date]))

## **PREVIOUSQUARTER**

Returns a table that contains a column of all dates from the previous quarter, based on the first date in the dates column, in the current context.

### **Example**

Previous Quarter = CALCULATE(SUM(Orders[Sales]), PREVIOUSQUARTER(Dim\_Date[Date]))

## **PREVIOUSYEAR**

Returns a table that contains a column of all dates from the previous year, given the last date in the dates column, in the current context.

### **Example**

Previous Year = CALCULATE(SUM(Orders[Sales]), PREVIOUSYEAR(Dim\_Date[Date]))

## **NEXTDAY**

Returns a table that contains a column of all dates from the next day, based on the first date specified in the dates column in the current context.

### **Example**

Next Day = CALCULATE ([Sum of Sales], NEXTDAY(Dim\_Date[Date]))

## **NEXTMONTH**

Returns a table that contains a column of all dates from the next month, based on the first date in the dates column in the current context.

### **Example**

Next Month = CALCULATE ([Sum of Sales], NEXTMONTH(Dim\_Date[Date]))

## **NEXTQUARTER**

Returns a table that contains a column of all dates in the next quarter, based on the first date specified in the dates column, in the current context.

### **Example**

Next Quarter = CALCULATE ([Sum of Sales], NEXTQUARTER(Dim\_Date[Date]))

## NEXTYEAR

Returns a table that contains a column of all dates in the next year, based on the first date in the dates column, in the current context.

### Example

Next Year = CALCULATE ([Sum of Sales], NEXTYEAR(Dim\_Date[Date]))

## SAMEPERIODLASTYEAR

Returns a table that contains a column of dates shifted one year back in time from the dates in the specified dates column, in the current context.

### Example

LastYearSales = CALCULATE(SUM(Orders[Sales]), SAMEPERIODLASTYEAR(Dim\_Date[Date]))

By using **same period last year**, you will be knowing

- Last Year Same Month what is the value
- Last Year Same Quarter what is the value
- Last Year what is the Value

Year	Sales	LastYearSales
2014	484,247.50	
2015	470,532.51	484,247.50
2016	609,205.60	470,532.51
2017	733,215.26	609,205.60
2018		733,215.26
<b>Total</b>	<b>2,297,200.86</b>	<b>2,297,200.86</b>

Year	Quarter Name	SumOfSales	LastYearSales
2014	Qtr1	74,447.80	
2014	Qtr2	86,538.76	
2014	Qtr3	143,633.21	
2014	Qtr4	179,627.73	
2015	Qtr1	68,851.74	74,447.80
2015	Qtr2	89,124.19	86,538.76
2015	Qtr3	130,259.58	143,633.21
2015	Qtr4	182,297.01	179,627.73
2016	Qtr1	93,237.18	68,851.74
2016	Qtr2	136,082.30	89,124.19
2016	Qtr3	143,787.36	130,259.58
2016	Qtr4	236,098.75	182,297.01
2017	Qtr1	123,144.86	93,237.18
2017	Qtr2	133,764.37	136,082.30
2017	Qtr3	196,251.96	143,787.36
2017	Qtr4	280,054.07	236,098.75
2018	Qtr1		123,144.86
2018	Qtr2		133,764.37
2018	Qtr3		196,251.96
2018	Qtr4		280,054.07
<b>Total</b>		<b>2,297,200.86</b>	<b>2,297,200.86</b>

## Year Over Year (YOY Growth)

Current Year Sales = SUM(Orders[Sales])

Last Year Sales = CALCULATE (SUM(Orders[Sales]), SAMEPERIODLASTYEAR(Dim\_Date[Date]))

% YOY = DIVIDE ([Current Year Sales], [Last Year Sales], "NO LY Data")

Year	Sum of Sales	Last Year Sales	%YOY
2014	484,247.50		
2015	470,532.51	484,247.50	97.17%
2016	609,205.60	470,532.51	129.47%
2017	733,215.26	609,205.60	120.36%
2018		733,215.26	
<b>Total</b>	<b>2,297,200.86</b>	<b>2,297,200.86</b>	<b>100.00%</b>

## MOM Growth

Sum of Sales = SUM(Orders[Sales])

Last Month Sales = CALCULATE(SUM(Orders[Sales]), PREVIOUSMONTH(Dim\_Date[Date]))

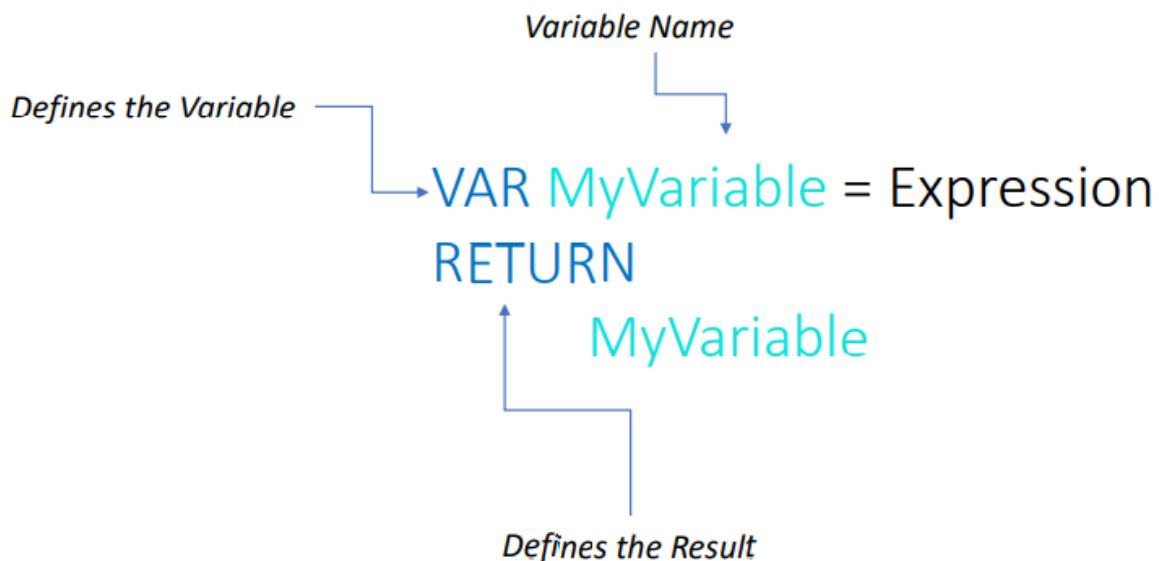
% MOM = DIVIDE ([Sum of Sales], [Last Month Sales])

Year	Month Name	Sum of Sales	Last Month Sales	% MOM
2014	January	14,236.90		
2014	February	4,519.89	14,236.89	31.75%
2014	March	55,691.01	4,519.89	1232.13%
2014	April	28,295.35	55,691.01	50.81%
2014	May	23,648.29	28,295.35	83.58%
2014	June	34,595.13	23,648.29	146.29%
2014	July	33,946.39	34,595.13	98.12%
2014	August	27,909.47	33,946.39	82.22%
2014	September	81,777.35	27,909.47	293.01%
2014	October	31,453.39	81,777.35	38.46%
2014	November	78,628.72	31,453.39	249.98%
2014	December	69,545.62	78,628.72	88.45%
2015	January	18,174.08	69,545.62	26.13%
2015	February	11,951.41	18,174.08	65.76%
2015	March	38,726.25	11,951.41	324.03%
2015	April	34,195.21	38,726.25	88.30%
2015	May	30,131.69	34,195.21	88.12%
2015	June	24,797.29	30,131.69	82.30%
2015	July	28,765.33	24,797.29	116.00%
2015	August	36,898.33	28,765.33	128.27%
<b>Total</b>		<b>2,297,200.86</b>		

## Understanding Variables in DAX

- Variables are used to store results from DAX Expressions.
- A Variable can be a hardcoded value or the result of a complex DAX Expression.
- Variables can be used in any type of DAX calculation including calculated columns, measures, and tables.
- Advantages - Code Readability and Performance.

## Syntax



The `VAR` keyword introduces the definition of a variable. You can have as many variables as needed in a single expression, and each one has its own `VAR` definition. The `RETURN` keyword defines the expression to return as the result. Inside `RETURN` expression, you can use the variables, which are replaced by the computed value.

## Uses of Variables

A very frequent usage of variables is to divide the calculation of a complex formula into logical steps, by assigning the result of each step to a variable.

```
%Sales =  
VARSumOfSales=SUM(Orders[Sales])  
VARAllSales=CALCULATE(SUM(Orders[Sales]),ALL(Orders))  
Return  
SumOfSales/AllSales
```

```
% YoY Growth =
VARCYSALES = SUM(Orders[Sales])
VARPYSALES =
CALCULATE(SUM(Orders[Sales]),PREVIOUSYEAR(Dim_Date[Date]))
RETURN
DIVIDE(CYSALES-PYSALES,PYSALES)
```

```
DelSpeed =
IF (
( Orders[Ship Date] - Orders[Order Date] <= 2 ),
    "QuickDel",
    IF ( ( Orders[Ship Date] - Orders[Order Date] <= 5 ),
"NormalDel", "LateDel" )
)
```

```
DelSpeedVar =
VAR DeltaDays = Orders[Ship Date] - Orders[Order Date]
RETURN
IF (DeltaDays<= 2, "QuickDel", IF (DeltaDays<= 5,
"NormalDel", "LateDel" ) )
```

## ALL, ALLSELECTED&ALLEXCEPT

### Requirement 1

Calculating the percentage of the total (ALL)

#### ALL

ALL Function will Returns all the rows in a table, or all the values in a column, ignoring any filters that might have been applied.

ALL ( [<TableNameOrColumnName>] [, <ColumnName> [, <ColumnName> [, ... ]]] )

Category	SumOfSales	%Sales
Furniture	741,999.80	32.30%
Office Supplies	719,047.03	31.30%
Technology	836,154.03	36.40%
Total	2,297,200.86	100.00%

### Requirement 2

Consider the filters applied to the visual (ALLSELECTED)

#### ALLSELECTED

ALLSELECTED Function will Returns all the rows in a table, or all the values in a column, ignoring any filters that might have been applied inside the query, but keeping filters that come from outside.

ALLSELECTED ( [<TableNameOrColumnName>] [, <ColumnName> [, <ColumnName> [, ... ]]] )

Category	SumOfSales	%Sales	%SalesAllSelected
Furniture	117,298.68	5.11%	29.94%
Office Supplies	125,651.31	5.47%	32.08%
Technology	148,771.91	6.48%	37.98%
Total	391,721.91	17.05%	100.00%

Region
Central
East
South
West

## Requirement 3

How to calculate the percentage of the parent total? (ALLEXCEPT)

When dealing with hierarchical data, the requirement is to calculate % of the parent total.

Category	SumOfSales	%Sales	%SalesParent
□ Furniture	741,999.80	32.30%	100.00%
Bookcases	114,880.00	5.00%	15.48%
Chairs	328,449.10	14.30%	44.27%
Furnishings	91,705.16	3.99%	12.36%
Tables	206,965.53	9.01%	27.89%
□ Office Supplies	719,047.03	31.30%	100.00%
Appliances	107,532.16	4.68%	14.95%
Art	27,118.79	1.18%	3.77%
Binders	203,412.73	8.85%	28.29%
Envelopes	16,476.40	0.72%	2.29%
Fasteners	3,024.28	0.13%	0.42%
Labels	12,486.31	0.54%	1.74%
Paper	78,479.21	3.42%	10.91%
Storage	223,843.61	9.74%	31.13%
Supplies	46,673.54	2.03%	6.49%
□ Technology	836,154.03	36.40%	100.00%
Accessories	167,380.32	7.29%	20.02%
Copiers	149,528.03	6.51%	17.88%
Machines	189,238.63	8.24%	22.63%
Phones	330,007.05	14.37%	39.47%
Total	2,297,200.86	100.00%	100.00%

In the example above, there are two levels of filter contexts applied

Level 1: Product Category

Level 2: Product Sub-Category

The objective is to keep filters at Level 1 and not at Level 2

%ParentSales =

`SUM(Orders [Sales])  
/ CALCULATE([SumOfSales], ALLEXCEPT(Orders, Orders [Category]))`

When used as a modifier in CALCULATE, ALLEXCEPT removes the filters from the expanded table specified in the first argument, keeping only the filters in the columns specified in the following arguments.

`ALLEXCEPT ( <TableName>, <ColumnName> [, <ColumnName> [, ... ] ] )`

New Columns

New Measures

New Tables

DAX Functions

---

Text Functions

Logical Functions

Date and Time Functions

Filter Functions

Math & Statistical Functions

Time Intelligence Functions

## Date and Time Functions

---

Year - Year Function will derive Year from given Date

Quarter - Quarter Function will Derive Quarter Number from a Given data

Month - Month Function will derive "Month Number" from given Date(1-12)

Day - Day Function will derive Day from given date (1-31)

Weekday - Weekday function will derive weekday number from given date(1-7)

Weeknum - Weeknum function will derive week number of year from given date (1-52/53)

Format - Use Format function to derive Month Name and Weekday Name

Month Name = Format (Orders [Order Date],"MMMM") --> Gives full Month Name

Month Name = Format (Orders [Order Date],"MMM") --> Gives Month Name abbreviations

Weekday Name = FORMAT (Orders [Order Date],"DDDD") --> Gives Full Weekday Name

Weekday Name = FORMAT (Orders [Order Date],"DDD") --> Gives Weekday Name abbreviations

DATE - Date function will accept Year,Month and Day and will return Date

TODAY - Today function will return todays Date and day start time

NOW - Now function will return todays Date and current Time - If You need Difference in Hours

HOUR - Hour funtion will return Hours from given Date and time

MINUTE - Minute funtion will return Minutes from given Date and time

SECOND - Second funtion will return Seconds from given Date and time

TIME - Takes Hours, Minutes and Seconds and gives you the time

DATEDIFF - Datediff function will give the difference between two dates.

No of Days = DATEDIFF (Orders [Order Date], Orders [Ship Date], DAY)

(Select don't summarize for No of Days to get correct results)

YEARFRAC - Calculates the fraction of the year represented by the number of whole days between two dates.

YearsOfExp = YEARFRAC(TODAY(),Emp[HIREDATE])

CALENDAR - The calendar function returns a table with a single column that contains a continuous set of dates. The start and end date range will be supplied as parameters.

CALENDARAUTO - Least Date - 3Jan2014 & Highest Data 28Nov2018

EDATE - Returns the date that is the indicated number of months before or after the start date.

EDATE ( <StartDate>, <Months> )

EOMONTH - Gives end of the Month for given Date. You can add even Months to the date "0" for current Month and 1 for Next month so on.

Returns the date in datetime format of the last day of the month before or after a specified number of months.

NoofDayInMonth = DAY(EOMONTH(Emp[HIREDATE],0))

STARTOFMONTH - Returns the start of month.

DATEVALUE - Converts a date in the form of text to a date in datetime format.

25-March-2020

25-Mar-2020

03-25-2020

DATEADD - Moves the given set of dates by a specified interval.

Date Dimension (Dim\_Date)

---

CALENDAR(Start\_Date,End\_Date)

Day = DAY(Dim\_Date[Date])

WeekDay = WEEKDAY(Dim\_Date[Date])

WeekDayName = FORMAT(Dim\_Date[Date],"DDDD")

Month = MONTH(Dim\_Date[Date])

MonthName = FORMAT(Dim\_Date[Date],"MMMM")

Quarter = QUARTER(Dim\_Date[Date])

QuarterName = "Qtr " & Dim\_Date[Quarter]

QuarterName = CONCATENATE ("Qtr", Dim\_Date[Quarter])

```
Year = YEAR(Dim_Date[Date])
WeekNumberYear = WEEKNUM(Dim_Date[Date])
WeekNameYear = "Week " & WEEKNUM(Dim_Date[Date])
WeekNumMonth = 1 + WEEKNUM(Dim_Date[Date]) - WEEKNUM(STARTOFMONTH(Dim_Date[Date]))
WeekNameMonth = "Week" & " " & Dim_Date[WeekNumMonth]
```

---

(April 2020 - March 2021) - FY 2021

```
FYear = IF(Dim_Date[MonthNo]<4,Dim_Date[Year],Dim_Date[Year]+1)
FYQuarter = SWITCH(TRUE(),Dim_Date[Month]<=3,"Qtr 4",Dim_Date[Month]<=6,"Qtr
1",Dim_Date[Month]<=9,"Qtr 2","Qtr 3")
```

## DAX Logical Functions

### IF

IF function is used to check the given expression is True or False.

If the expression results TRUE then second argument will return otherwise, third argument will return.

IF (LogicalTest, ResultIfTrue, [ResultIfFalse])

Syntax in Square Brackets is optional. If you are not providing third argument, then if the condition is not satisfied it will return NULL values.

SalRange = IF(Emp[SAL]<1000,"LowSal","HighSal")

### Nested IF Function

In Power BI DAX, you can use the Nested If concept, One If statement inside another.

Nested If Example = IF(Orders[Sales]<100,"Low Sales", IF(Orders[Sales]>300, "High Sales", "Avg Sales"))

SalRange = IF(Emp[SAL]<1000,"LowSal",IF(Emp[SAL]>=3000,"HighSal","MediumSal"))

DNAME =

IF(Emp[DEPTNO]=10,"SALES",IF(Emp[DEPTNO]=20,"HR",IF(Emp[DEPTNO]=30,"MARK","TRAINING")))

---

### TRUE, FALSE

The DAX TRUE function will return a logical value True. The DAX FALSE function will return logical False.

True and False Example = IF(Orders[Sales]>100, TRUE (), FALSE ())

True/False = NOT(IF(Emp[DEPTNO]=10,TRUE(),FALSE()))

True/False = IF(Orders[Discount]>0,TRUE(),FALSE())

True/False = IF(ISBLANK(Emp[COMM]),FALSE(),TRUE())

CommB = IF(ISBLANK(Emp[COMM]),1,0)

---

### NOT

The DAX NOT function will convert True to False, and False to True. I mean, it returns opposite result.

NOT example = NOT(IF(Orders[Sales]>100, TRUE (), FALSE ()))

CommBlank = NOT(IF(ISBLANK(Emp[COMM]),TRUE(),FALSE()))

---

### OR - ||

The DAX OR function is like either or statement in English, which is used to check multiple expressions.

Weekday = WEEKDAY (Orders [Order Date])

Weekend = IF(OR(Orders[Weekday]=1, Orders[Weekday]=7),"Weekend", "Weekday")

Weekend = IF(Orders[Weekday]=1 || Orders[Weekday]=7,"Weekend", "Weekday")

T	T	T
T	F	T
F	T	T
F	F	F

---

## IN

IN Example = IF(Orders[Weekday] IN {2,3,4,5,6}, " Weekday ", Weekend")

WWInOpe = IF(Orders[DaYName] IN {"Friday","Saturday","Sunday"}, "Weekend", "Weekday")

---

AND &&

The DAX AND function is used to check multiple expressions.

And Example = IF(AND(Orders[Category]="Office Supplies", Orders[Sub-Category] = "Storage"), TRUE (), FALSE ())

And example = IF(Orders[Category]="Office Supplies" && Orders[Sub-Category] = "Storage", TRUE(), FALSE())

RvsdSal = IF(AND(Emp[DEPTNO]=30,Emp[job]="SALESMAN"),Emp[SAL]+1000,Emp[SAL])

RSal = IF(Emp[DEPTNO]=30 && Emp[job]="SALESMAN",Emp[SAL]+1000,Emp[SAL])

T	T	T
T	F	F
F	T	F
F	F	F

---

## IFERROR

The DAX IFERROR function is very useful to handle the arithmetic overflow, or any other errors. It simply performs calculation and return the result, if there is an error then it will return the value inside the second argument.

Error Example = IFERROR(Orders[Sales]/Orders[Quantity], 100)

Error Example = IFERROR(Orders[Sales]/0, 100)

---

## SWITCH

The DAX SWITCH function helps you to return multiple options.

SWITCH () always test for equivalence.

DNAME =

IF(Emp[DEPTNO]=10,"SALES",IF(Emp[DEPTNO]=20,"HR",IF(Emp[DEPTNO]=30,"MARK",IF(Emp[DEPTNO]=40,"IT","UnKnown"))))

Switch Example = SWITCH (MONTH (Orders [Order Date]), 1, "January", 2, "February", 3, "March", 4, "April", 5, "May", 12, "December", "Unknown")

Switch Example = SWITCH (TRUE (), Orders [Sales]<100, "Low Sales", Orders[Sales]>300,"High Sales", "Medium Sales")

DNAME =

```
IF(Emp[DEPTNO]=10,"Sales",IF(Emp[DEPTNO]=20,"HR",IF(Emp[DEPTNO]=30,"Marketing","UnKnown")))
DName Switch = SWITCH(Emp[DEPTNO],10,"Sales",20,"HR",30,"Marketing","UnKnown")
SalRangeIf = IF(Emp[SAL]<1000,"LowSal",IF(Emp[SAL]>=3000,"HighSal","MediumSal"))
SalRangSwitch = SWITCH(TRUE(),Emp[SAL]<1000,"LowSal",Emp[SAL]>=3000,"HighSal","MediumSal")
```

## Math and Statistical Functions

### INT

INT Rounds a number down to the nearest integer.

INT Example = INT (Orders [Sales])

12.34 =12      12.78 = 12      12.55=12

---

### ROUND

Round function will Rounds a number to the given number of digits.

Round Example = ROUND (Orders [Sales], 0)

12.34 =12      12.78 = 13      12.55=13

Round Example = ROUND (Orders [Sales], 1)

12.34 =12.3      12.78 = 12.8      12.55=12.6

>=5 Next Number

<5 Same Number

---

### ROUNDUP

Roundup will Rounds a number to next integer value if num\_digits = 0.

ROUNDUP Example = ROUNDUP (Orders [Sales], 0)

12.34 =13      12.78 = 13      12.55=13

ROUNDUP Example = ROUNDUP (Orders [Sales], 1)

12.34 =12.4      12.78 = 12.8      12.55=12.6

---

### ROUNDDOWN

ROUNDDOWN Rounds a number down, toward zero.

ROUNDDOWN Example = ROUNDDOWN (Orders [Sales], 0) = INT(Orders [Sales])

12.34 =12      12.78 = 12      12.55=12

ROUNDDOWN Example = ROUNDDOWN (Orders [Sales], 1)

12.34 =12.3      12.78 = 12.7      12.55=12.5

---

### DIVIDE /

Performs division and returns alternate result or BLANK () on division by 0.

Divide Example = DIVIDE (Orders [Sales], Orders [Quantity])

Divide Example = DIVIDE (Orders [Sales], Orders [Quantity], 0)

Divide Example = DIVIDE (200,4,0) = 50

Divide Example = DIVIDE (100,0, -1) = -1

---

## EVEN

Returns number rounded up to the nearest even integer.

Even Example = EVEN (Orders [Sales])

12.34 =14	12.78 = 14	12.55=14
13.34 =14	13.78 = 14	13.55=14

---

## ODD

Returns number rounded up to the nearest odd integer.

### Example

Odd Example = ODD (Orders [Sales])

12.34 =13	12.78 = 13	12.55=13
13.34 =15	13.78 = 15	13.55=15

---

## POWER (^)

Returns the result of a number raised to a power.

### Example

Power Example = POWER (Orders [Quantity], Orders [Quantity])

Power Example = POWER (Orders [Quantity],2)

Power Example = POWER (4,2) ->16

Power Example = POWER (3,3) ->27

---

## SIGN

SIGN determines the sign of a number, the result of a calculation, or a value in a column.

The function returns 1 if the number is positive, 0 (zero) if the number is zero, or -1 if the number is negative.

### Example

Sign Example = SIGN (Orders [Profit])

## SQRT

SQRT returns the square root of a number. If the number is negative, the SQRT function returns an error.

### Example

SQRT Example = SQRT (Orders [Quantity])

SQRT Example = ROUND (SQRT (Orders [Quantity]), 2)

## FACT

Returns the factorial of a number, equal to the series 1\*2\*3\*..., ending in the given number.

### Example

FACT Example = FACT (Orders [Quantity])

Measures - %, Ratios or Aggregate - Group of Rows when we need one value

## Types of Measures

1. Implicit Measures

2. Explicit Measures

Create Table Visualization which shows SumSal, MinSal, MaxSal, AvgSal, CountOfEmp from Emp Table

Create Table Visualization which shows dept wise SumSal, MinSal, MaxSal, AvgSal, CountOfEmp from Emp Table

TotSal= SAL+COMM

## SUM

SUM Function Adds all the numbers in a column.

SUM (<column>)

Sum of Sal = SUM (EMP [SAL])

What is the Difference Between SUM and SUMX???

## SUMX

Returns the sum of an expression evaluated for each row in a table.

SUMX (<table>, <expression>)

SUMX of Sal = SUMX (EMP, EMP [SAL] +EMP [COMM])

SumOfTotSal = SUM(EMP [SAL]) + SUM(EMP [COMM]))

## MIN

Returns the smallest numeric value in a column.

MIN (<column>)

Min of Sal = MIN(EMP[SAL])

---

## MINX

Returns the smallest numeric value that results from evaluating an expression for each row of a table.

MINX (<table>, <expression>)

MINX of Sal = MINX (EMP, EMP [SAL]+ EMP [COMM])

---

## MAX

Returns the largest numeric value in a column.

MAX (<column>)

MAX of Sal = MAX(EMP[SAL])

---

## MAXX

Evaluates an expression for each row of a table and returns the largest numeric value.

MAXX (<table>, <expression>)

MAXX of Sal = MAXX (EMP, EMP [SAL]+ EMP [COMM])

---

## COUNT

Counts the number of cells in a column that contain numbers or values.

COUNT (<column>)

COUNT Example = COUNT(EMP[SAL])

Count Example1 = COUNT (EMP [COMM])

---

## COUNTX

Counts the number of rows that contain a number or an expression that evaluates to a number, when evaluating an expression over a table.

COUNTX (<table>, <expression>)

COUNTX Example = COUNTX (EMP, EMP[SAL]+EMP[COMM])

All Statistical Functions they Don't Consider NULL Values

Any arithmetic operation with NULL results you NULL

Null +100 = Null

---

## AVERAGE

AVERAGE Function Will Returns the average of all the numbers in a column.

AVERAGE (<column>)

Average Example = AVERAGE(EMP[SAL])

Average of COMM = AVERAGE(EMP[COMM])

---

## AVERAGEX

Calculates the average of a set of expressions evaluated over a table.

AVERAGEX (<table>, <expression>)

AVERAGEX Example = AVERAGEX (EMP, EMP[SAL]+EMP[COMM])

## COUNTROWS

Counts the number of rows in the specified table, or in a table defined by an expression.

COUNTROWS (<table>)

COUNTROWS Example = COUNTROWS(EMP)

---

## COUNTBLANK

Counts the number of blank cells in a column.

COUNTBLANK (<column>)

COUNTBLANK Example = COUNTBLANK(EMP[COMM])

---

## RANKX

The RANKX function is used in DAX to create rankings.

Rank the Salaries for Each Employee

RANKX (TABLE, EXPRESSION [, VALUE] [, ORDER] [, TIES]...)

ORDER - ASC or DESC - Default Desc

ASC - Least Values first Rank and Highest Values Last Rank

DESC - Highest Values first Rank and Least Values Last Rank

TIES - Dense or Skip - Default Skip

Dense – will not Skip Between Rank Numbers

Skip - Skip Between Rank Numbers

Optional arguments might be skipped by placing an empty comma (,) in the argument list,  
i.e. RANKX = RANKX (EMP, EMP[SAL],,,Dense)

---

## SUMMARIZE

Returns a "summary table or Aggregate Table" for the requested totals over a set of groups.

DEPTSAL =

SUMMARIZE(emp,Dept[DEPTNO],Emp[JOB],"CountOfEmp",COUNT(Emp[EMPNO]),"SumOfSal",SUM(Emp[SAL]))



## Text Functions

### LEN

LEN function will Returns the Number of characters in string.

#### Example

Length = LEN (Orders [Customer Name])

---

### CONCATENATE (&)

CONCATENATE function joins two text strings into one text string.

Concat = CONCATENATE (Orders [Category], CONCATENATE (" - ", Orders [Sub-Category]))  
Concatenate = Orders [Category] & " - " & Orders [Sub-Category]

### CLARK - MANAGER

### CONCATENATEX

Create a Measure to Show Concatenated list of Employees Name

ConEmpName = CONCATENATEX(emp,Emp[ENAME],", ")

### COMBINEVALUES

Address2 = COMBINEVALUES(", ",Orders[City],Orders[State],Orders[Country])

Address = CONCATENATE(CONCATENATE(CONCATENATE(CONCATENATE(Orders[City],", "),Orders[State]),", "),Orders[Country])

Address1 = Orders[City]& ", "&Orders[State]&", "&Orders[Country]

---

### LEFT

LEFT Function Returns the specified number of characters from the start of a text string.

Left = LEFT (Orders [Country], 5)

---

### RIGHT

RIGHT function returns the last character or characters in a text string, based on the number of characters you specify.

Right = RIGHT (Orders [Country], 5)

## MID

MID Function Returns a string of characters from the middle of a text string, given a starting position and length.

Mid = MID (Orders [Country], 8, 5)

## UPPER

UPPER Function Converts a text string to all uppercase letters.

Upper = UPPER (Orders [Country])

## LOWER

Lower Function Converts all letters in a text string to lowercase.

Lower = LOWER (Orders [Country])

## TRIM

TRIM Function Removes all spaces from text except for single spaces between words.

Trim = TRIM (Orders [Customer Name])

## SUBSTITUTE

SUBSTITUTE Function Replaces existing text with new text in a text string.

Substitute = SUBSTITUTE (Orders [Country], "t", "T") – Replace existing text with new text for all occurrences  
Substitute = SUBSTITUTE (Orders [Country], "t", "T", 2) - Replace existing text with new text for second occurrence only.

SubEx = SUBSTITUTE(Orders[Order ID],"-",", ")

SubEx = SUBSTITUTE(Orders[Order ID],"-",", ",2)

## REPLACE

REPLACE(Emp[JOB],1,3,"@")

Replace = REPLACE(Orders[Sub-Category],1,3,"@")

## BLANK

Returns a blank.

## Usage

Avoids ‘Divide by zero’ error.

Blank = IF (Orders[Quantity] = 0, BLANK (), Orders[Sales]/ Orders[Quantity])

## Time Intelligence

Time Intelligence means doing calculations over periods of time or dates.

Time Intelligence Functions will need a DATE column to perform the calculations.

This Date column should contain unique, no null and contiguous date values to get the accurate results.

You will find a Date Column in Date Dimension Table Which will Satisfy the above 3 Conditions.

To make DAX Time Intelligence functions work properly set Date Dimension Table as “Mark as date table”.

When you specify your own date table, Power BI Desktop performs the following validations of that column and its data, to ensure that the data

- contains unique values

- contains no null values

- contains contiguous date values (from beginning to end)

- if it is a Date/Time data type, it has the same timestamp across each value

---

## DAX Time Intelligence Functions

---

### TOTALMTD - MonthToDate

Evaluates the value of the expression for the month to date, in the current context.

TOTALMTD will give Running Totals or cumulative sum for each Month.

Total MTD = TOTALMTD(SUM(Orders[Sales]), Dim\_Date[Date])

---

### TOTALQTD

Evaluates the value of the expression for the dates in the quarter to date, in the current context.

TOTALQTD will give Running Totals or cumulative sum for each Quarter.

Total QTD = TOTALQTD(SUM(Orders[Sales]), Dim\_Date[Date])

---

### TOTALYTD

Evaluates the year-to-date value of the expression in the current context.

TOTALYTD will give Running Totals or cumulative sum for each year.

Total YTD = TOTALYTD(SUM(Orders[Sales]), Dim\_Date[Date])

---

### PREVIOUSDAY

Returns a table that contains a column of all dates representing the day that is previous to the first date in the dates column, in the current context.

Previous Day = CALCULATE(SUM(Orders[Sales]), PREVIOUSDAY(Dim\_Date[Date]))

---

### PREVIOUSMONTH

Returns a table that contains a column of all dates from the previous month, based on the first date in the dates column,

in the current context.

Previous Month = CALCULATE(SUM(Orders[Sales]), PREVIOUSMONTH(Dim\_Date[Date]))

Prev6MonthSales = CALCULATE(SUM(Orders[Sales]),PREVIOUSMONTH(DATEADD(Dim\_Date[Date],-6,MONTH)))

---

#### PREVIOUSQUARTER

Returns a table that contains a column of all dates from the previous quarter, based on the first date in the dates column, in the current context.

Previous Quarter = CALCULATE(SUM(Orders[Sales]), PREVIOUSQUARTER(Dim\_Date[Date]))

---

#### PREVIOUSYEAR

Returns a table that contains a column of all dates from the previous year, given the last date in the dates column, in the current context.

Previous Year = CALCULATE(SUM(Orders[Sales]), PREVIOUSYEAR(Dim\_Date[Date]))

% YoY = ([SumSales]-[PYSales])/[PYSales]

---

#### NEXTDAY

Returns a table that contains a column of all dates from the next day, based on the first date specified in the dates column in the current context.

Next Day = CALCULATE ([Sum of Sales], NEXTDAY(Dim\_Date[Date]))

---

#### NEXTMONTH

Returns a table that contains a column of all dates from the next month, based on the first date in the dates column in the current context.

Next Month = CALCULATE ([Sum of Sales], NEXTMONTH(Dim\_Date[Date]))

---

#### NEXTQUARTER

Returns a table that contains a column of all dates in the next quarter, based on the first date specified in the dates column, in the current context.

Next Quarter = CALCULATE ([Sum of Sales], NEXTQUARTER(Dim\_Date[Date]))

---

#### NEXTYEAR

Returns a table that contains a column of all dates in the next year, based on the first date in the dates column, in the current context.

Next Year = CALCULATE ([Sum of Sales], NEXTYEAR(Dim\_Date[Date]))

---

## SAMEPERIODLASTYEAR

Returns a table that contains a column of dates shifted one year back in time from the dates in the specified dates column, in the current context.

LastYearSales = CALCULATE(SUM(Orders[Sales]), SAMEPERIODLASTYEAR(Dim\_Date[Date]))

By using same period last year, you will be knowing

    Last Year Same Month what is the value

    Last Year Same Quarter what is the value

    Last Year what is the Value

---

## Year Over Year (YOY Growth)

CurrentYearSales or SumofSales = SUM(Orders[Sales])

LastYearSales = CALCULATE (SUM(Orders[Sales]), SAMEPERIODLASTYEAR(Dim\_Date[Date]))

LastYearSales = CALCULATE(SUM(Orders[Sales]), PREVIOUSYEAR(Dim\_Date[Date]))

% YOYGrowth = DIVIDE( ([SumofSales]-[LastYearSales]),[LastYearSales])

---

## MOM Growth

SumofSales = SUM(Orders[Sales])

LastMonthSales = CALCULATE(SUM(Orders[Sales]), PREVIOUSMONTH(Dim\_Date[Date]))

% MOM = DIVIDE ([[SumofSales]- [Last Month Sales]] , [Last Month Sales])

---

PrevYearSales = CALCULATE(SUM(Orders[Sales]),PREVIOUSYEAR(Dim\_Date[Date]))

%Sales = ([SumOfSales]-[SamePrdLastYear])/[SamePrdLastYear]

%Sales = ([SumOfSales]-[SamePrdLastYear])/[SamePrdLastYear]

% YoY = ([SumSales]-[PYSales])/[PYSales]

Quick Measures - Without Writing the Formula You can Create Measures

% YOY Growth, Sales YoY%

YTD, MTD

L3M = CALCULATE(SUM(Orders[Sales]),DATEADD(Dim\_Date[Date],-3,MONTH))



## Enhancing the Data Model - Creating the New information from Existing Information

DAX - Data Analysis eXpressions - Functional Language used in Power BI - Power Pivot, SSAS Tabular, Excel Power Pivot

SSIS - ETL - Power Query

SSAS - OLAP - Power Pivot - Tabular OLAP

SSRS - Reporting - Power View

### MSBI SSAS

SSAS Multidimensional - MDX - Multi – Dimensional eXpressions

SSAS Tabular - DAX - Data Analysis eXpressions

What you are going to do with DAX?

New Columns / Calculated Columns, New Measures & New Tables

TotSal = Emp[SAL]+Emp[COMM]

Unit Price=Orders[Sales]/'Orders Details'[Quantity]

Sum Of Sales = SUM(Orders[Sales])

ExpOfEmp = DATEDIFF(Emp[HIREDATE],TODAY(),YEAR)

Calculated Columns / New Columns, New Measures & New Table are created using DAX Functions, Operators, and Constants along with Existing Columns

Revised Sal = 'EMP Details'[SAL]+1000

ExpOfEmp = DATEDIFF(Emp[HIREDATE],TODAY(),YEAR)

RvsdSal = IF(Emp[DEPTNO]=10,Emp[SAL]+1000,Emp[SAL]+2000)

=====

DAX Table and Column Name syntax

'Table Name'[ColumnName]

TableName[Column Name]

=====

New Column / Calculated Column ->Calculated Column will be created when you want a calculation for every row in your table.

Use a calculated column when you want to evaluate each row

Creating New Column / Calculated Column

=====

Measures - Use a calculated measure when you are calculating percentages or ratios, or you need complex aggregations.

Use a measure when you need an aggregate

Creating Measure

Types of Measures

Implicit Vs Explicit Measures

Implicit Measures are created when you drag raw numerical fields (like Sales, Profit, Quantity) into the values field well of a visual and manually select the aggregation mode (Sum, Average, Min/Max, etc.).

Explicit measures are created by actually entering DAX functions like below

Sum Of Sales = SUM(Orders[Sales])

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

Explicit measures can be used anywhere in the report and referenced with other DAX calculations.

Sum Of Sales = SUM (Orders[Sales])

% Sales = [Sum Of Sales]/CALCULATE([Sum Of Sales],ALL(Orders))

=====

Calculated Column Vs Measure

Power Pivot - Enhancing Data Model - New Columns, New Measures & New Tables

Formulas - DAX Functions, Operators & Constants, Existing Columns

Types of Operators

Arithmetic Operators

+ (Addition)

- (Subtraction)

\* (Multiplication)

/ (Division)

<sup>^</sup> (Exponentiation)

=====

Comparision Operator

= (Equal to)

<> (Not equal to)

< (Less than)

<= (Less than or equal to)

> (Greater than)

>= (Greater than or equal to)

Rvsd Sal = IF(EMP[DEPTNO]=10,EMP[Sal]+1000,'EMP'[Sal]+2000)

SalRange = IF(Emp[SAL]<3000,"LowSal","HighSal")

Existing Information - Emp[SAL]

DAX Functions - IF

Operators - Comparision Operator (<)

Constants - 3000

=====

Text Concatenation Operator

& (ampersand)

QuarterName = "Qtr "& Orders[Quarter]

=====

Logical Operators

&& (double ampersand) - AND

|| (double pipe symbol) - OR -> Deptno =10 || Deptno = 20 || Deptno =30

IN{} -> Deptno IN{10,20,30}

-----



## Power Pivot

Power Pivot is an In-Memory Columnar Database.

We used Power Pivot to Store the Data that is required for Analysis Purpose.

### In-MemoryVs. Traditional Databases / Relational Databases

In Traditional Databases / Relational Databases Data will be Stored in Disk and accessed from the Disk.

In In-Memory Databases the Data will be stored in RAM and accessed from the RAM.

In Traditional Databases / Relational Databases Data will be Stored in Row Format.

Row Storage			
ENAME	JOB	SAL	DEPTNO
MILLER	CLERK	1300	10
SMITH	CLERK	800	20
JONES	MANAGER	2975	20
SCOTT	ANALYST	3000	20
ADAMS	CLERK	1100	20
FORD	ANALYST	3000	20

MILLER   CLERK   1300   10	SMITH   CLERK   800   20	JONES   MANAGER   2975   20
Block 1	Block 2	Block 3

In In-Memory Databases the Data will be stored in Columnar Format.

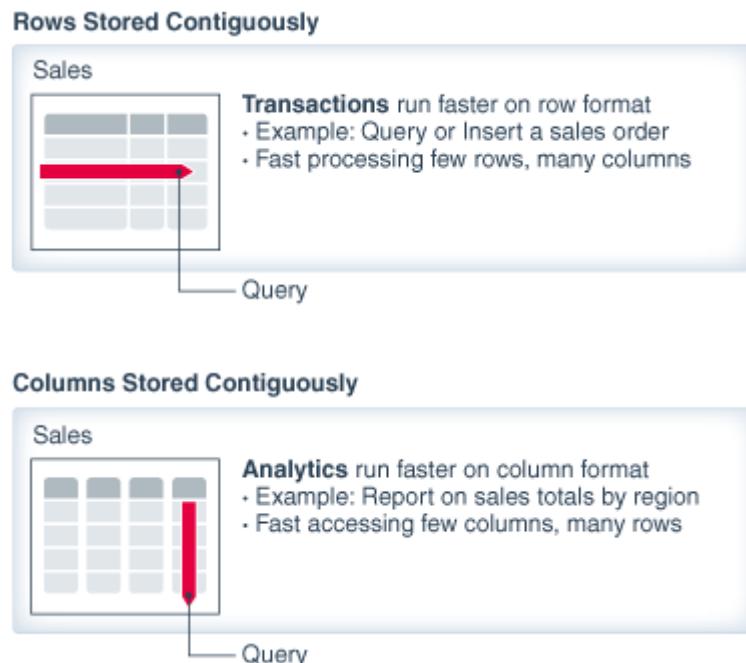
Columnar Storage			
ENAME	JOB	SAL	DEPTNO
MILLER	CLERK	1300	10
SMITH	CLERK	800	20
JONES	MANAGER	2975	20
SCOTT	ANALYST	3000	20
ADAMS	CLERK	1100	20
FORD	ANALYST	3000	20

MILLER   SMITH   JONES   SCOTT   ADAMS   FORD	CLERK   CLERK   MANAGER   ANALYST   CLERK   ANALYST	1300   800   2975   3000   1100   3000
Block 1	Block 2	Block 3

Columnar Storage makes it possible to read data faster, which is crucial for analytical queries.

Columnar storage also takes up less disk space, because each block contains the same type of data, meaning it can be compressed into a specific format.



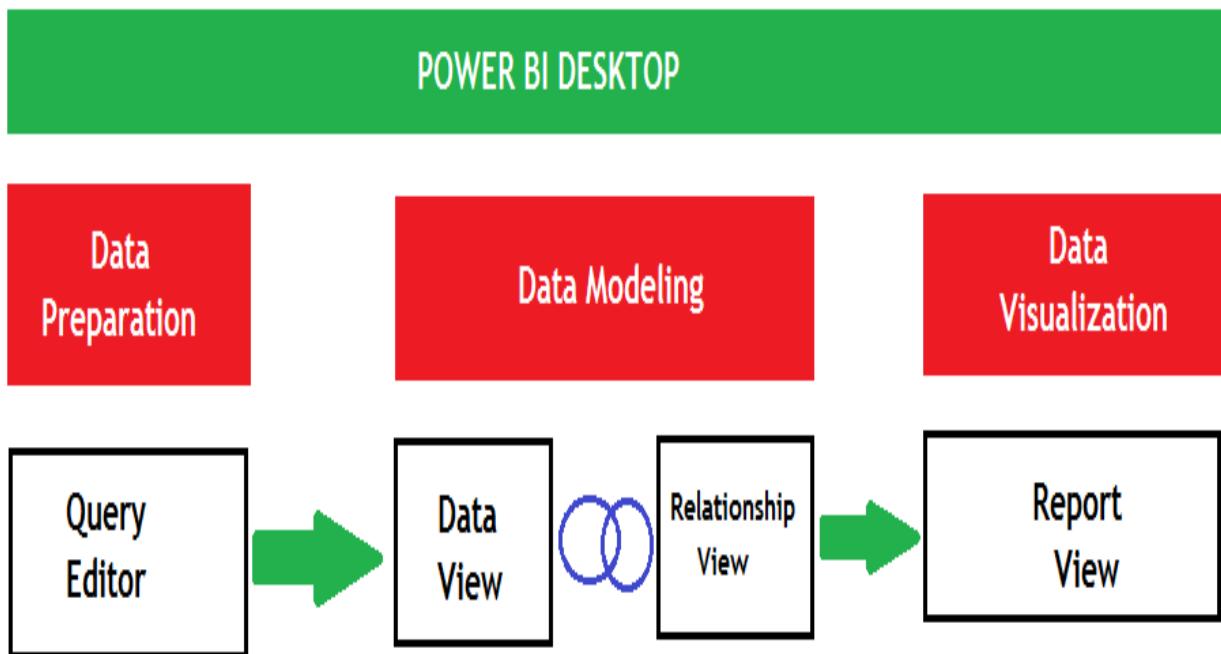
## Advantages of In-Memory Columnar Databases

1. When you create Reports or Visuals on Power Pivot You Can See the Data in Reports or Visuals Faster as you access the Data from RAM.
2. Data is Highly Compressed in Power Pivot (Approximately 1/10<sup>th</sup> Source Data).
3. When You Perform the Aggregate Calculations on Power Pivot they are Much Faster.

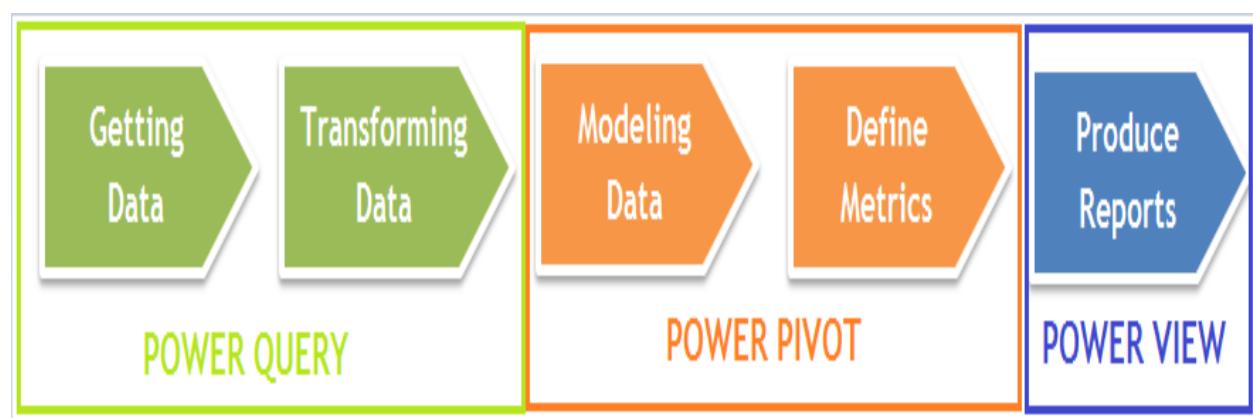
## Power BI Modeling

In this section, we will learn about Modeling Data, relationships, their definition, use, and how to create relationships, different types of relationships in Power BI. After this, we will learn how to manage relationships in Power BI. Managing relationships means adding, editing, and deleting relationships as well as making these active or inactive.

Below image explains different steps involved in Power BI Desktop.



## Power BI Desktop Steps



Power BI can connect to different data source, Transform or Prepare the Data using Power Query or Query Editor and then Load the transformed data into Power Pivot, to perform data modeling.

## Data Modeling

Establishing the Relationships Between the Tables we call as Data Modeling.

Modeling is to create a connection between different tables, for that you create a relationship.

Often, you'll will use fields from more than one Tables to create your Visualizations, and you'll need all of that data to work together. Modeling is how you get it there.

Usually in OLAP systems we stores descriptive information in Dimension Tables and Numeric information that is helpful for Analyzing the business will be stored in Fact Table. We always use the dimension table along with fact table to build reports or visualizations.

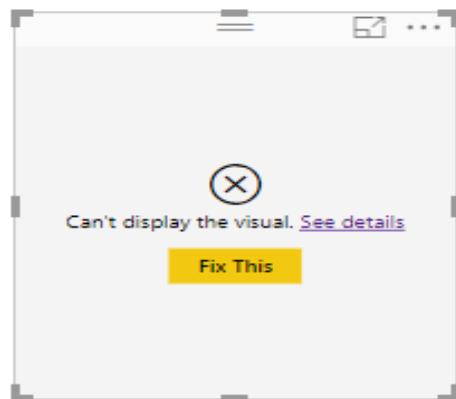
## Relationship

As the name suggests, Relationship in Power BI is used to define the connections or the relation between two or more tables. The relationship is used when we want to perform an analysis based on multiple tables. Relationship helps us to display the data and correct information between multiple tables. It is used to calculate the accurate results also.

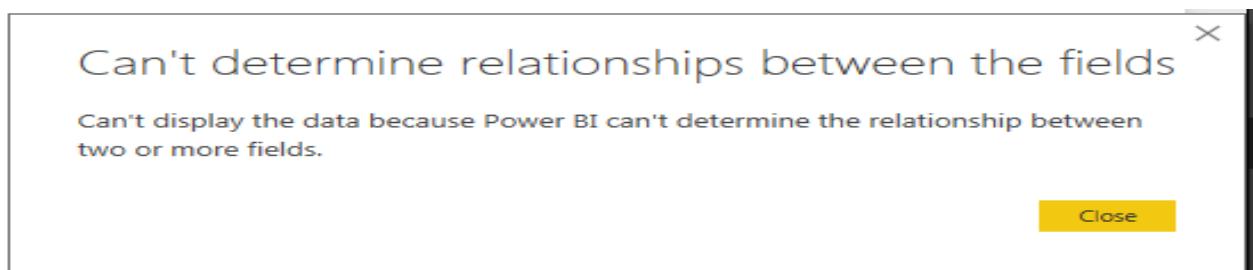
A relationship between data sources enables Power BI to know how those tables relate to one another, allowing you to create interesting visuals and reports.

## Need of Relationship

If we don't have relationship between two tables which are using in the same visualization for analysis, throws error as shown below or else it will show wrong results.



When you Click the See details hyperlink, you will be getting the message as shown in below image.

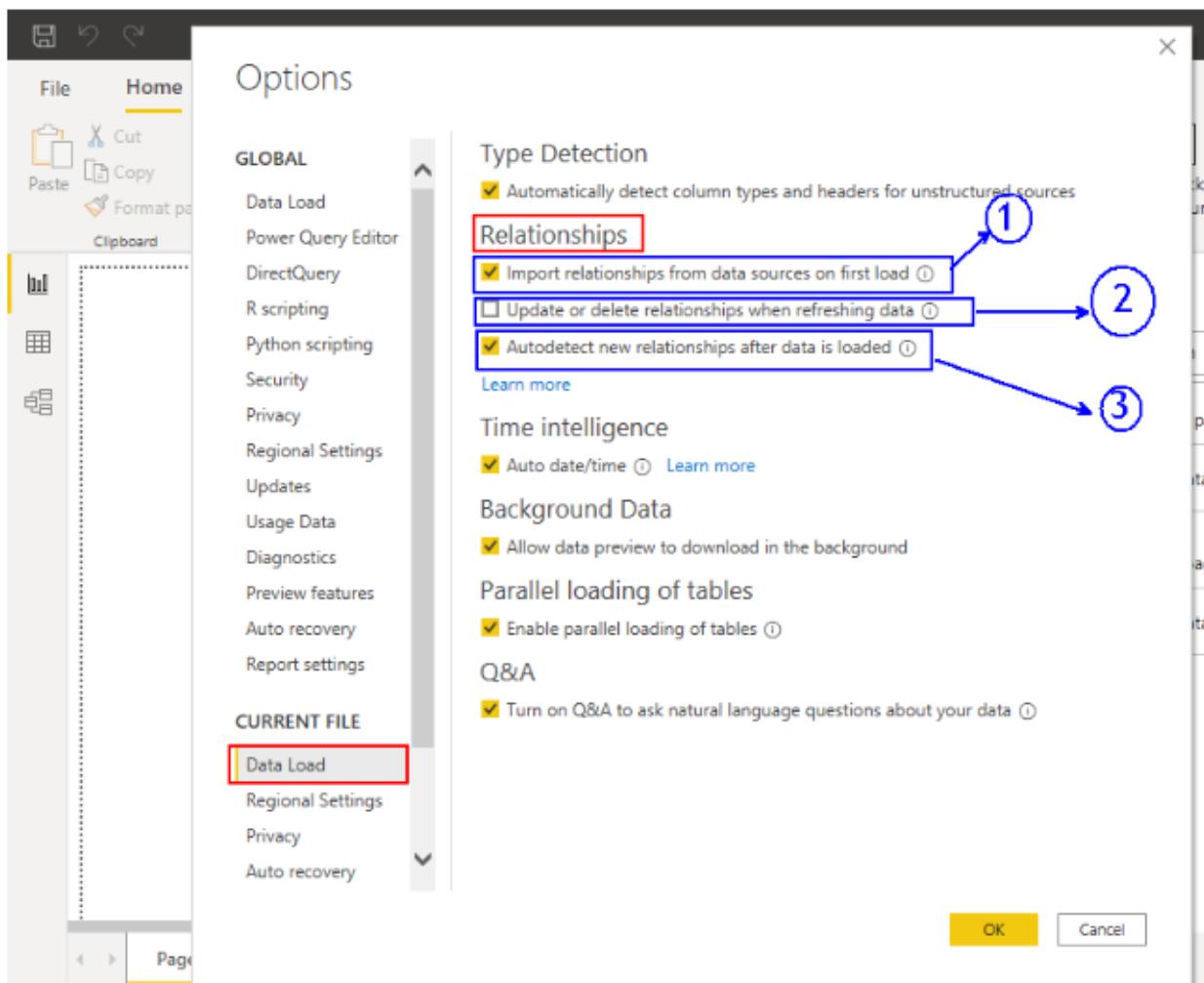


When you Load the Tables into Power Pivot, Majority of the Data Modeling Work Will be Done By the Power BI itself.

Power BI Desktop will create the relationships for you automatically. It is important to verify these relationships Created by Power BI to ensure accuracy.

Below are the Options which help to Create the Relationships by Power BI.

Go to File Menu - Options & Setting - Options - Data Load



## Relationship Types / Cardinality in General

There are three types of relationships or Cardinality between tables

- One-to-One
- One-to-Many (or Many-to-One)
- Many-to-Many

## One-to-One

A row in table A can have only one matching row in table B, and vice versa.

Ex: Person and Passport Table

## One-to-Many (or Many-to-One)

This is the most common relationship type. In this type of relationship, a row in table A can have many matching rows in table B, but a row in table B can have only one matching row in table A.

One-to-Many relationships can also be viewed as Many-to-One relationships, depending on which way you look at it.

Ex: Customer and Sales Table

Sales table is the “many” and the Customer table is the “one”. Customer Table will have customer record only once but a Sales table will have multiple sales values for the same customer. When you see from Sales table to Customer Table it is Many-to-One relationship similarly when you see from Customer Table to Sales Table it is One-to-Many relationship.

## Many-to-Many

In a many-to-many relationship, a row in table A can have many matching rows in table B, and vice versa. Customer can buy many products and a Product can be purchased by many Customers.

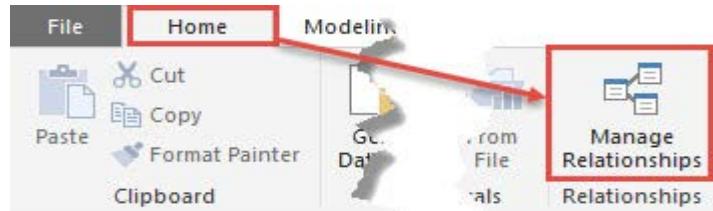
Ex: Customers and Products

Relationships cannot be built directly between tables that have a many-to-many relationship in Power BI.

A many-to-many relationship could be thought of as two one-to-many relationships, linked by an intermediary table.

The intermediary table is typically referred to as a “junction table” (also as a “cross-reference table”). This table is used to link the other two tables together. It does this by having two fields that reference the primary key of each of the other two tables.

The first thing you should do after importing data is to verify that all auto-detected relationships have been created correctly. From the modeling ribbon, select Manage Relationships.



This will open up the Manage Relationships editor. The relationship editor is where you will go to create new relationships, Autodetect Relationship and edit or delete existing relationships. The relationship editor will be used to verify the relationships that were automatically created by Power BI Desktop.

Let's take a look at the Manage Relationships editor, in which you can manage or perform the following.

1. Current relationships in the data model
2. Create a new relationship
3. Edit existing relationships
4. Delete a relationship
5. Autodetect the relationship

## Manage relationships

Active	From: Table (Column)	To: Table (Column)
<input checked="" type="checkbox"/>	FactInternetSales (CustomerKey)	DimCustomer (CustomerKey)
<input checked="" type="checkbox"/>	FactInternetSales (ProductKey)	DimProduct (ProductKey)
<input checked="" type="checkbox"/>	FactInternetSales (SalesTerritoryKey)	DimSalesTerritory (SalesTerritoryKey)

1  
2  
3  
4  
5

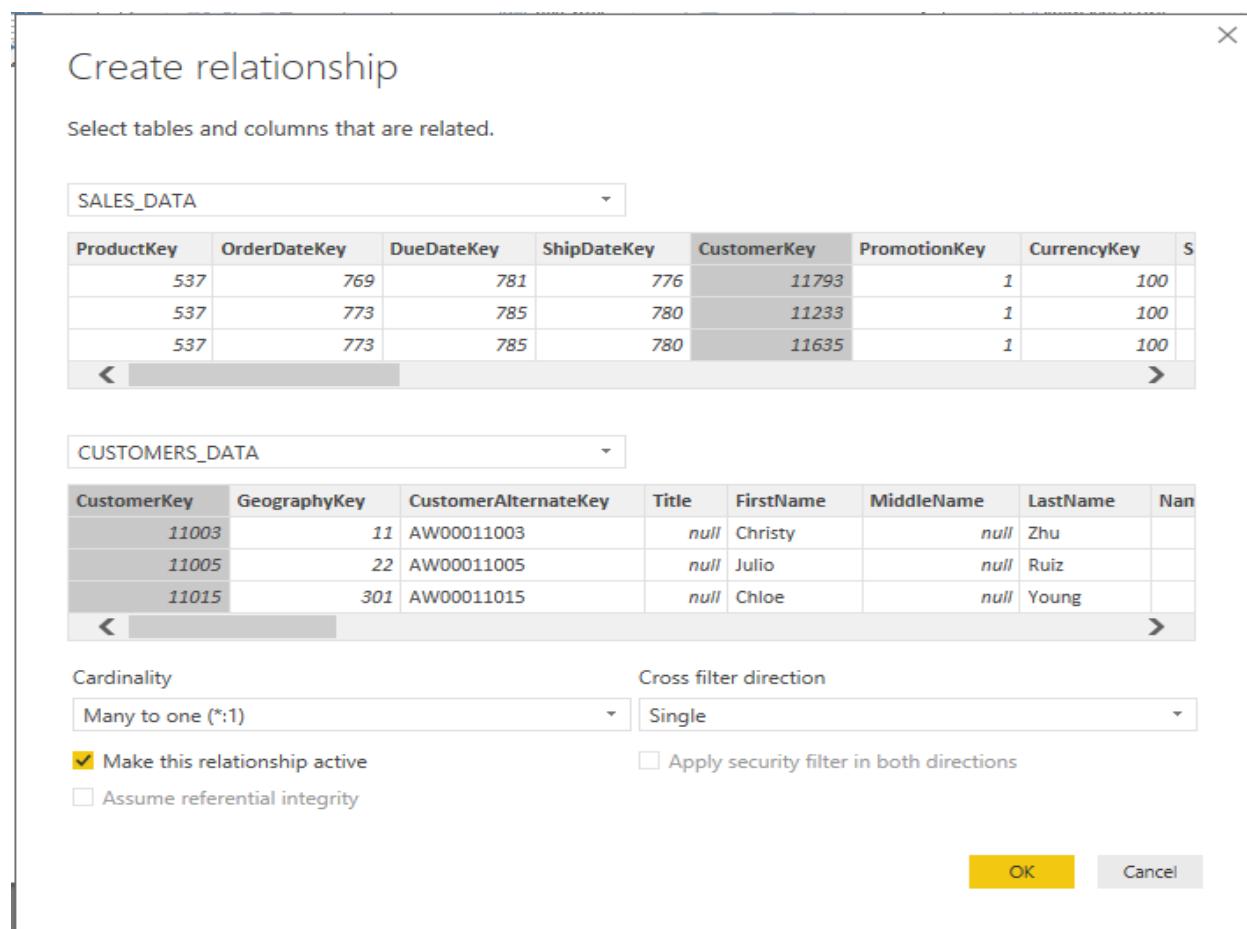
New... Autodetect... Edit... Delete Close

First, you need to verify auto-detected relationships. The top half of the relationship editor gives you a quick and easy way to see what tables have relationships between them, what columns the relationships have been created on, and if the relationship is an active relationship.

## To create the new relationship

You can select Autodetect option in Manage relationships window to identify the relationships exists between the tables automatically by Power BI or else you can create manually using below steps.

- Click Manage Relationships. In Manage Relationships Window, click New. This opens the Create Relationship dialog / Window, where we can select the tables, columns, and any additional settings we want for our relationship.
- For the first table, select SALES\_DATA, then select the CustomerKey column in SALES\_DATA table. This is the many side of our relationship.
- For the second table, select CUSTOMERS\_DATA, then select the CustomerKey column. This is the one side of our relationship.
- Go ahead and click OK in the Create Relationship dialog and the Manage Relationships dialog Click on close.



## Editing relationships

Now, let's take a look at how to edit an existing relationship. In this example, you will edit the relationship between FactInternetSales and DimCustomer. To edit an existing relationship, select that relationship and then click on Edit.

## Manage relationships

Active

From: Table (Column)	To: Table (Column)
FactInternetSales (CustomerKey)	DimCustomer (CustomerKey)
FactInternetSales (ProductKey)	DimProduct (ProductKey)
FactInternetSales (SalesTerritoryKey)	DimSalesTerritory (SalesTerritoryKey)

New... Autodetect... Edit... Delete

Once you select Edit... you will receive a new dialog box, this is the Edit Relationship editor. In this view, you will see how to change an existing relationship, how to change a relationship to active or inactive, and the cardinality of the current relationship, this is also where you can change the cross filter direction if required.

## Edit relationship

Select tables and columns that are related.

1

2

3

4

5

OK Cancel

There are five things we want to look at in the edit relationship window

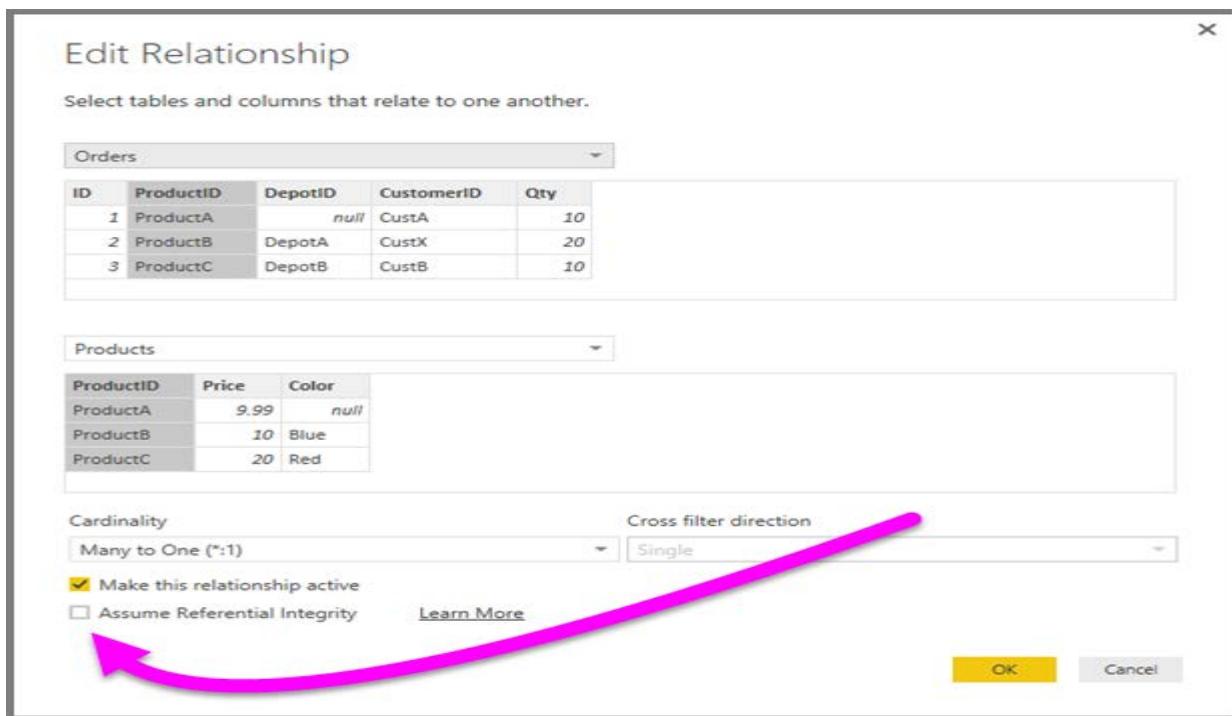
1. This identifies the FactInternetSales table and the column that the relationship was built on.

2. This identifies the DimCustomer table and the column that the relationship was built on.
3. This checkbox identifies whether the relationship is active or inactive.
4. This is the current cardinality or relationship type between the two tables. Here we see that there is a many-to-one relationship between FactInternetSales and DimCustomer. Power BI does an excellent job of identifying the correct cardinality, but it is important to always verify that the cardinality is correct.
5. The cross filter direction can be single or both. *The one side of a relationship always filters the many side of the relationship, and this is the default behavior in Power BI.* If you filter values in dimension table corresponding values in fact table will be shown. The cross filter option allows you to change this behavior. Cross filtering will be discussed later in this chapter.

### Assume referential integrity settings in Power BI Desktop

When connecting to a data source using DirectQuery, you can use the Assume Referential Integrity selection to enable running more efficient queries against your data source. When we are using DirectQuery, every time when we drag the fields to visualizations, Power BI will run Queries on top of Source Databases. This feature has a few requirements of the underlying data, and it is only available when using DirectQuery.

Setting Assume referential integrity enables queries on the data source to use INNER JOIN statements rather than OUTER JOIN, which improves query efficiency.



Requirements for using Assume referential integrity to Give Correct Values

This is an advanced setting, and is only enabled when connecting to data using DirectQuery. The following requirements are necessary for Assume referential integrity to work properly.

- Data in the **From** column in the relationship is never Null or blank.
- For each value in the **From** column, there is a corresponding value in the **To** column.
- In this context, the **From** column is the Many in a One-to-Many relationship, or it is the column in the first table in a One-to-One relationship.

### Example of using Assume referential integrity

The following example demonstrates how Assume referential integrity behaves when used in data connections. The example connects to a data source that includes an Orders table, a Products table, and a Depots table.

1. In the following image that shows the Orders table and the Products table, note that referential integrity exists between Orders[ProductID] and Products[ProductID]. The [ProductID] column in the Orders table is never Null, and every value also appears in the Products table. As such, Assume Referential Integrity should be set to get more efficient queries (using this setting does not change the values shown in visuals).

The diagram illustrates two tables side-by-side. On the left is the 'Orders table' with the following data:

ID	ProductID	DepotID	CustomerID	Qty
1	ProductA	null	CustA	10
2	ProductB	DepotA	CustX	20
3	ProductC	DepotB	CustB	10

On the right is the 'Products table' with the following data:

ProductID	Price	Color
ProductA	9.99	null
ProductB	10	Blue
ProductC	20	Red

2. In the next image, notice that no referential integrity exists between Orders[DepotID] and Depots[DepotID], because the DepotID is Null for some Orders. As such, Assume Referential Integrity should not be set.

**Orders  
table**

ID	ProductID	DepotID	CustomerID	Qty
1	ProductA	<i>null</i>	CustA	10
2	ProductB	DepotA	CustX	20
3	ProductC	DepotB	CustB	10

**Depots  
table**

DepotID	City
DepotA	Seattle
DepotB	New York

3. Finally, no referential integrity exists between Orders[CustomerID] and Customers[CustID] in the following tables, the CustomerID contains some values (in this case, CustX) that do not exist in the Customers table. As such, Assume Referential Integrity should not be set.

**Orders  
table**

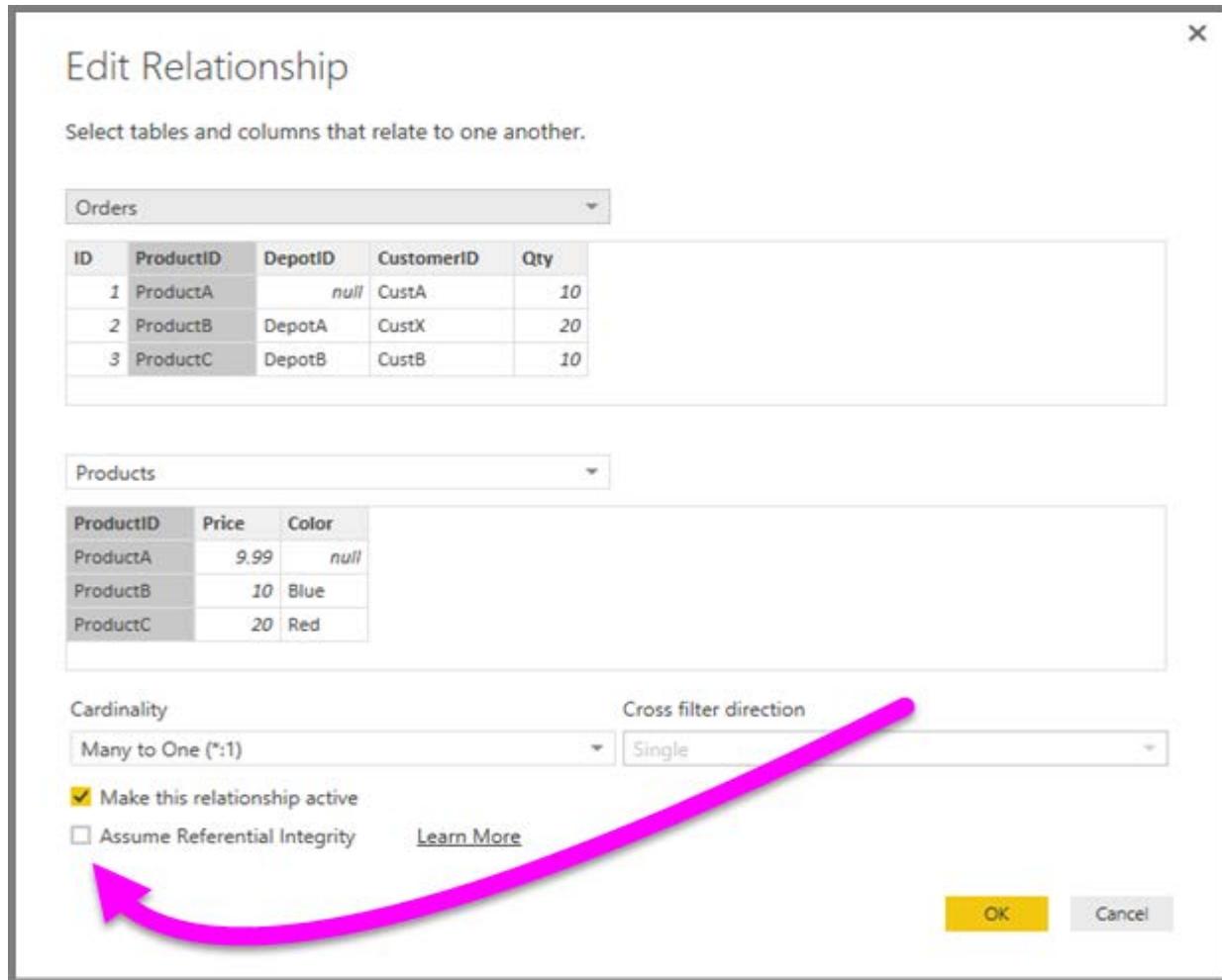
ID	ProductID	DepotID	CustomerID	Qty
1	ProductA	<i>null</i>	CustA	10
2	ProductB	DepotA	CustX	20
3	ProductC	DepotB	CustB	10

**Customers  
table**

CustID	Name
CustA	John Doe
CustB	Jack Smith

#### Setting Assume referential integrity

To enable this feature, select the checkbox next to Assume Referential Integrity as shown in the following image.



When selected, the setting is validated against the data to ensure there are *no Null or mismatched rows*. However, for cases with a very large number of values, the validation is not a guarantee that there are no referential integrity issues.

In addition, the validation occurs at the time of editing the relationship, and does not reflect any subsequent changes to the data.

#### What happens if you incorrectly set Assume referential integrity?

If you set Assume Referential Integrity when there are referential integrity issues in the data, this will not result in errors. However, it will result in apparent inconsistencies in the data. For example, in the case of the relationship to the Depots table described above, it would result in the following.

A visual showing the total Order Qty would show a value of 40

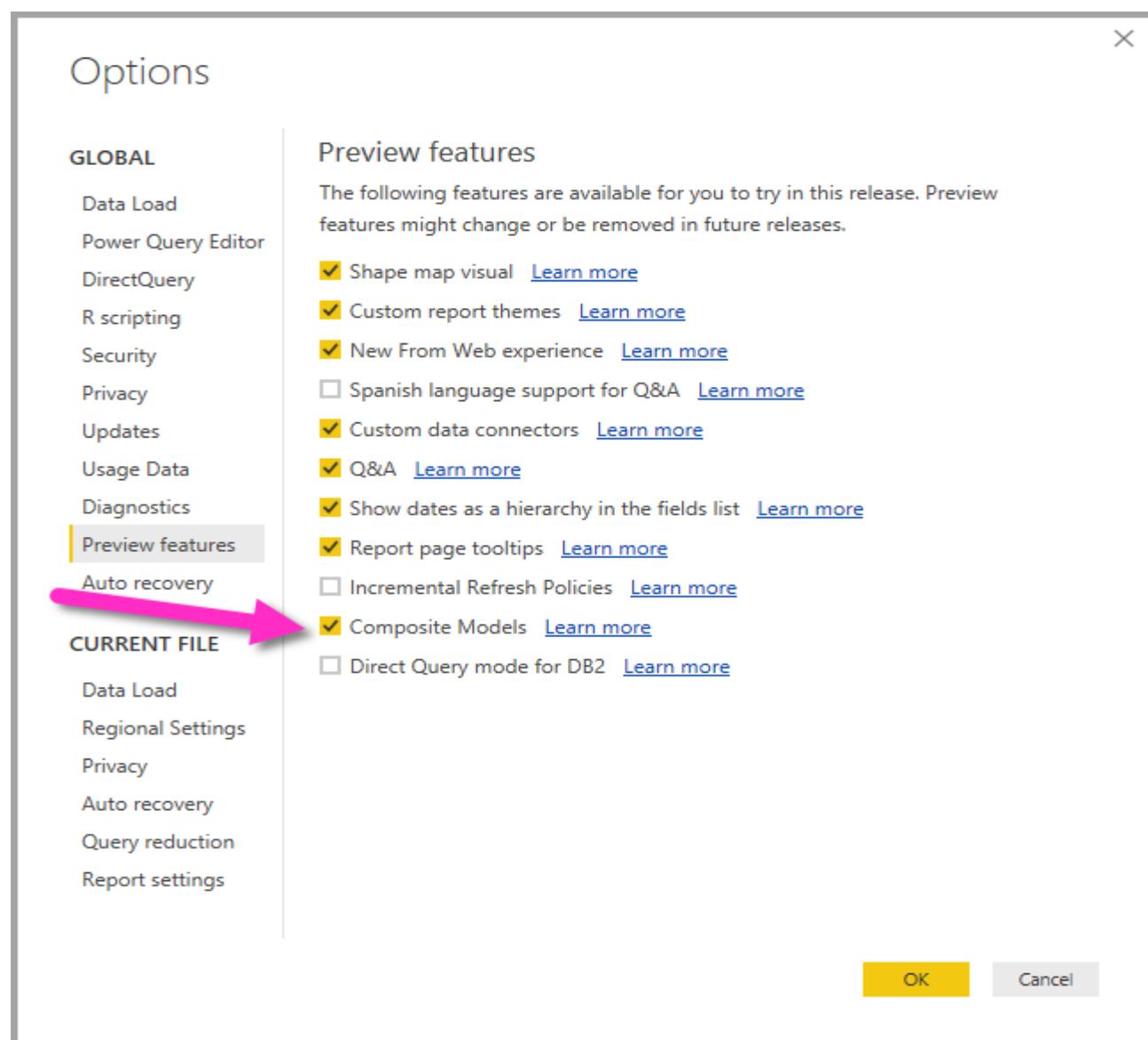
A visual showing the total Order Qty by Depot would show a total value of only 30, because it would not include Order ID 1, where DepotID is Null.

## Many-to-many relationships in Power BI Desktop - Preview July2018

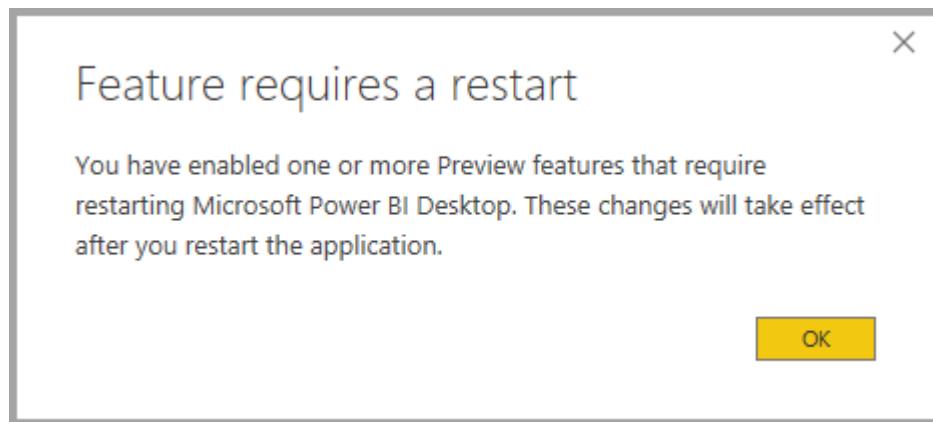
With the Many-to-many relationship feature in Power BI Desktop you can join tables using a cardinality of Many to Many, and create data models that contain multiple data sources easier and more intuitively.

### Enabling the many-to-many relationships preview feature

The many-to-many relationships feature is part of the composite models capabilities and is in Preview, and must be enabled in Power BI Desktop. To enable composite models, select File → Options and Settings → Options → Preview Features, then select the composite models checkbox.



You'll need to restart Power BI Desktop for the feature to be enabled.



### What many-to-many relationships solves

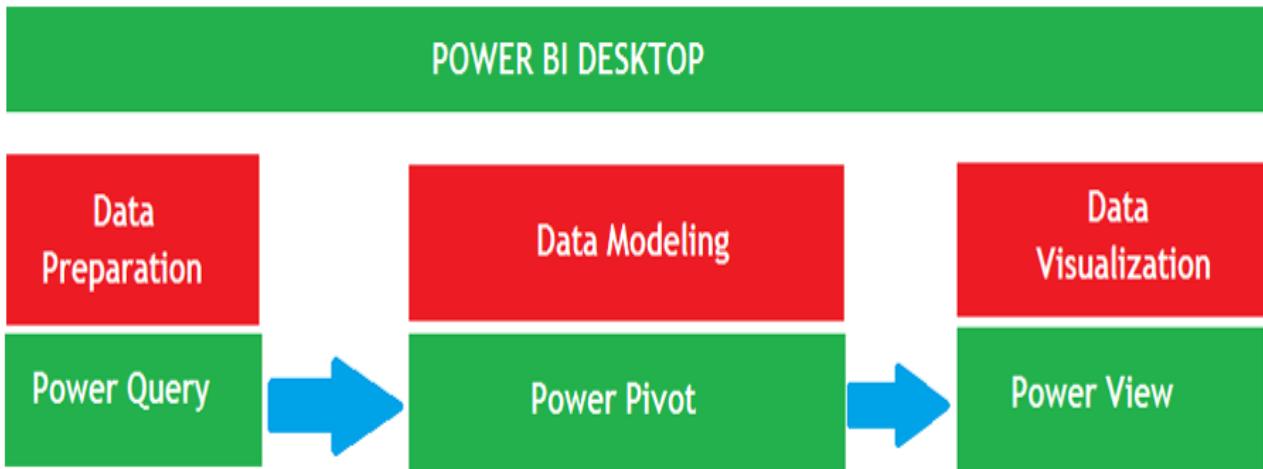
Prior to availability of many-to-many relationships, when defining a relationship between two tables in Power BI, at least one of the columns involved in the relationship had to contain unique values. In many circumstances though, no column in the table contained unique values.

For example, two tables may have a column containing the Country, but the values of Country were not unique in either table. To join between such tables, it was necessary to create a workaround such as introducing additional tables into the model that contained the necessary unique values. The feature many-to-many relationships provides an alternative approach, allowing such tables to be joined directly using a relationship with a cardinality of Many-to-many.

## Power BI Desktop Installation

Power BI Desktop lets you build advanced queries, models, and reports that visualize data. With Power BI Desktop, you can Prepare/Transform data, build data models, create reports, and share your work by publishing to the Power BI service or Power BI Report Server.

Data Preparation	→ Query Editor
Data Modeling	→ Relationship View+ Data View
Data Visualization	→ Report View



Power BI Desktop is free Software where Companies and Individuals can use it for Free.

### System Requirements

#### Supported Operating System

- Windows 10, Windows 7, Windows 8, Windows 8.1, Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2.

#### Pre-installed software's required

- Internet Explorer 9 or Later.
- Microsoft .NET Framework 4.7 or Later

#### Install Instructions

- Download the version of Power BI Desktop that matches the architecture (x86 or x64) of your Windows OS.
- Microsoft Power BI Desktop is available for 32-bit (x86) and 64-bit (x64) platforms.
- Power BI Desktop is free software and it is available at Below Microsoft's Power BI Site (<https://powerbi.microsoft.com/en-us/desktop/>)

**Go from data to insight to action with Power BI Desktop**

Create rich, interactive reports with visual analytics at your fingertips— for free.

**DOWNLOAD FREE >**

**SEE DOWNLOAD OR LANGUAGE OPTIONS >**

When you click on "See Download or Language Options", it will open the Below Link where you can download the Power BI Desktop Software.

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

*Important!* Selecting a language below will dynamically change the complete page content to that language.

Select Language:

**Download**

- By clicking on Download, it will open a Page where it will ask which Version of Software you need. If you have Windows 64 Bit Operating System installed in your Computer the select 64 Bit Power BI Software.

## Choose the download you want

File Name	Size
PBIDesktopSetup_x64.exe	289.1 MB
PBIDesktopSetup.exe	267.1 MB

Download Summary:  
KMBGB

1. PBIDesktopSetup\_x64.exe

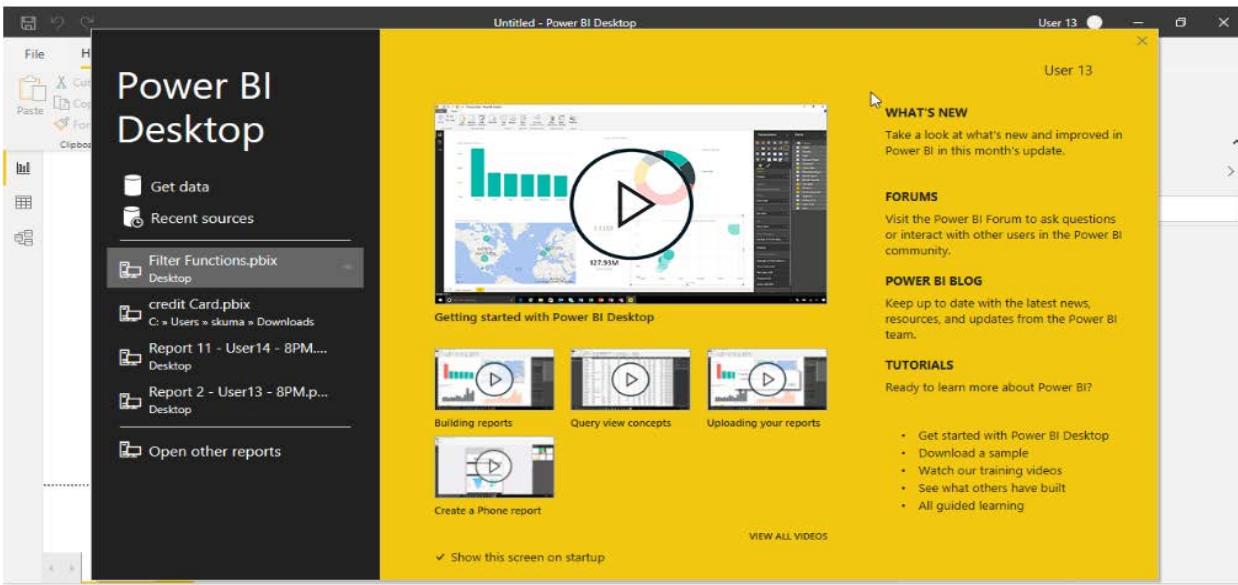
Total Size: 289.1 MB

**Next**

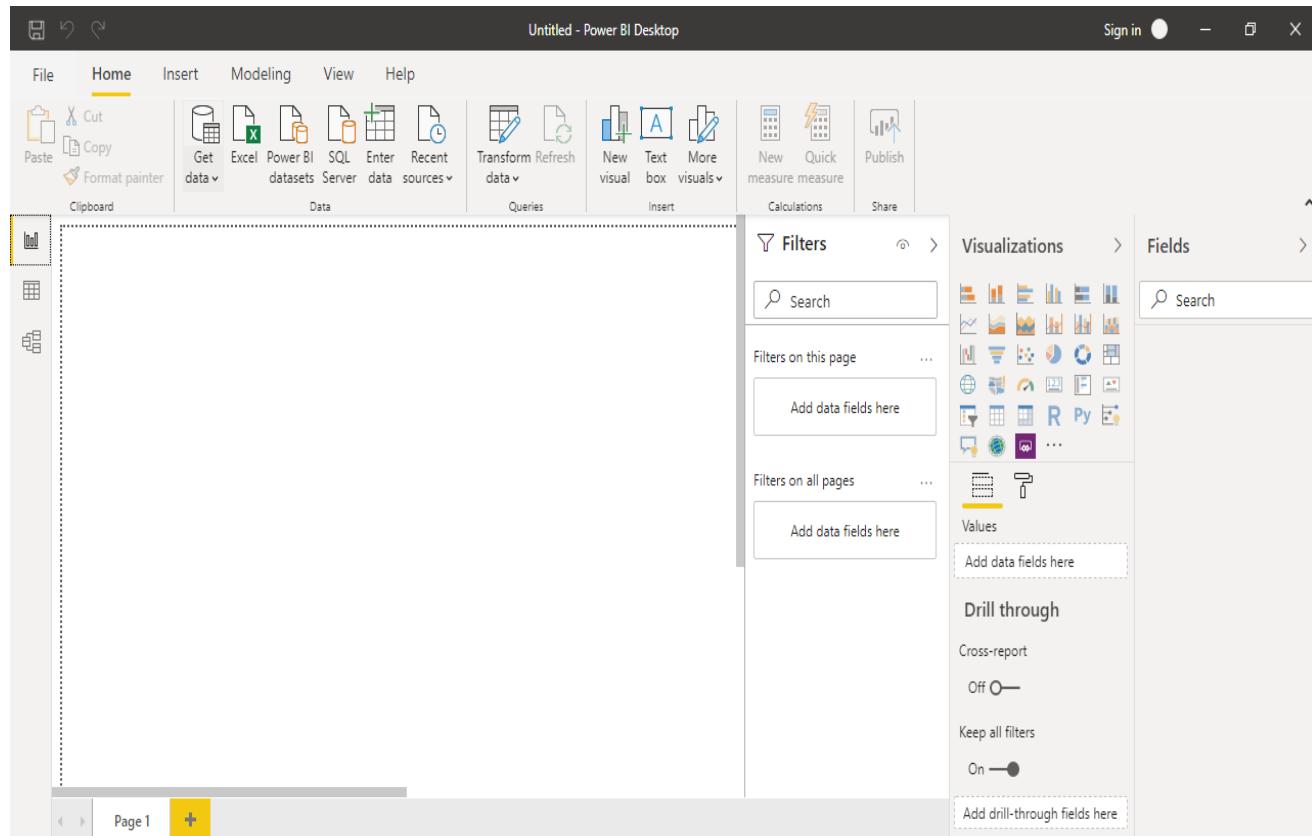
- Once Click on Next Software will be downloaded. Once after download Double Click on the Software to run the **EXE** installer and follow the setup steps, No Configuration Required.

## Opening Power BI Desktop

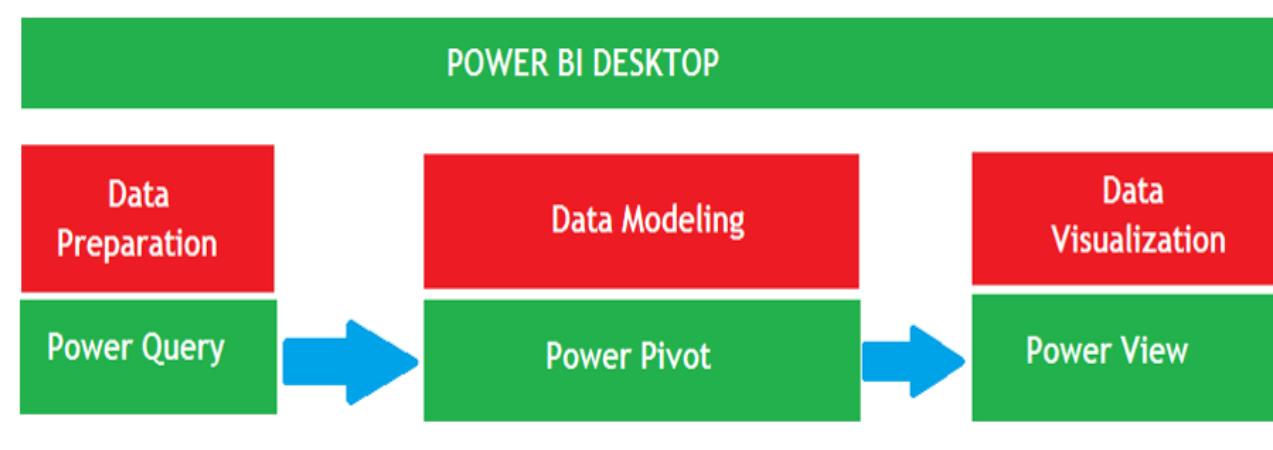
- ✓ All Programs → Microsoft Power BI Desktop →Power BI Desktop (OR)
- ✓ Double Click on the Shortcut Created in Desktop to Open Power BI Desktop
- ✓ The following image shows the Start Screen of Power BI Desktop, which appears when you start the application.



- ✓ From the start page or start screen you can connect to different data sources, can connect to recent sources, you can open existing saved reports, can access power BI Blogs, Forums, Tutorials so on.
- ✓ Once you close the start page then we will enter in Power BI Desktop Home page.
- ✓ If you close the screen (select the x in the top right corner), the Report view of Power BI Desktop is displayed as shown below.



We know that Power BI Desktop Software Contains three Software's internally. Let's Start understand about them now.

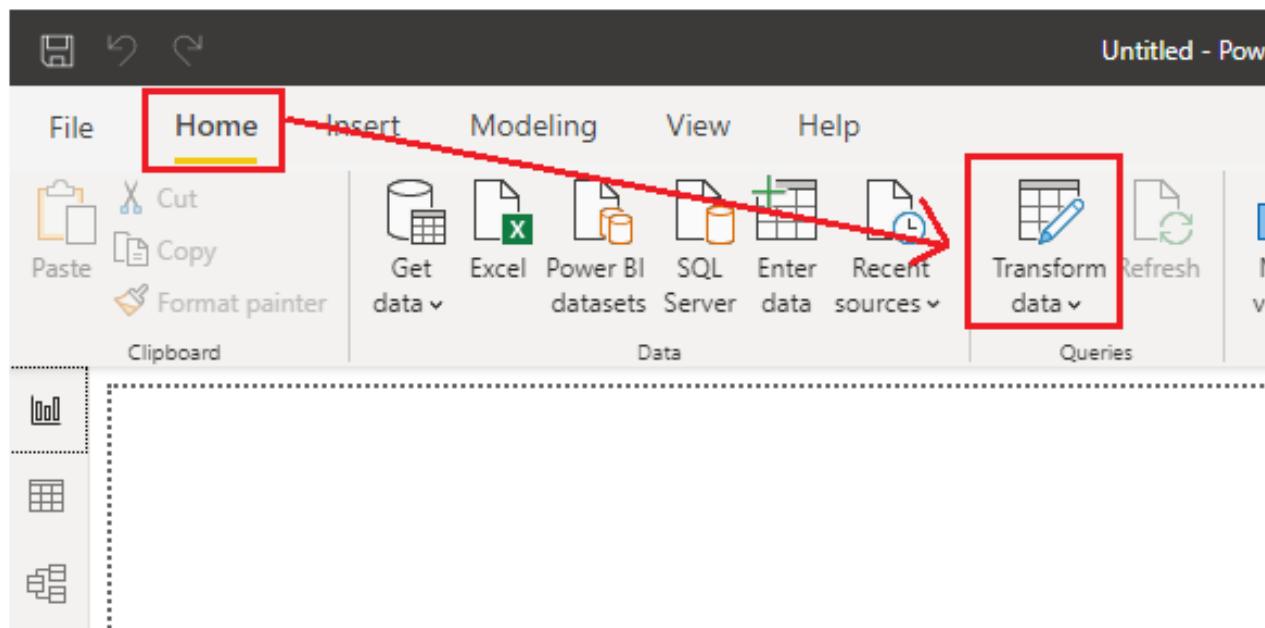


## Power Query Software Overview

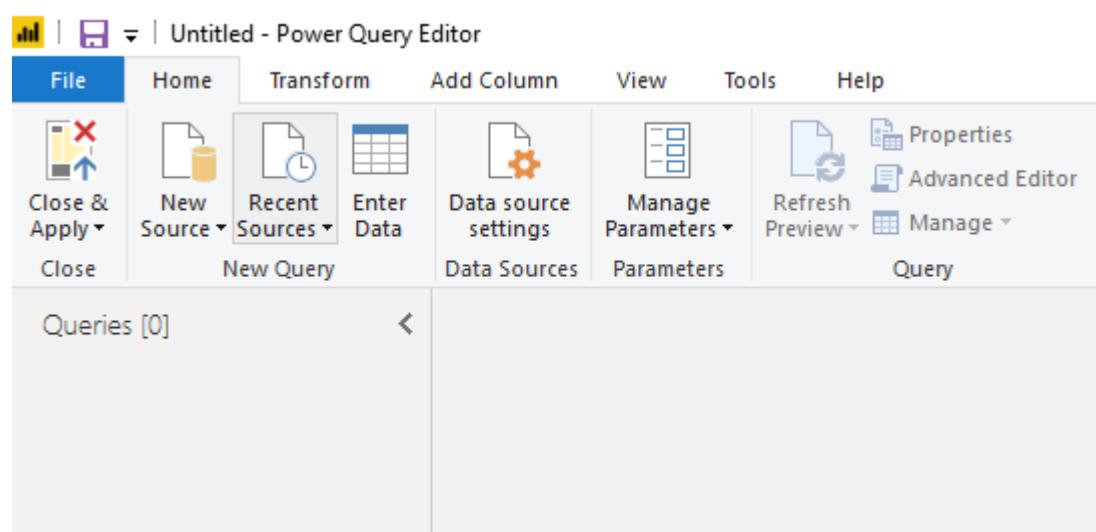
- ✓ Power Query is used for Data Extraction, Transformation and Loading. It's an ETL Software in Power BI.

## Opening Power Query Software

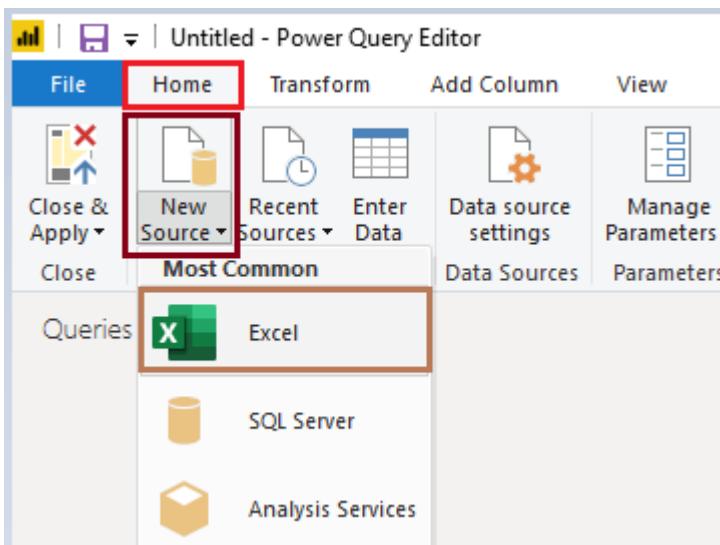
To Open Power Query Software Go to Home Tab → Transform Data



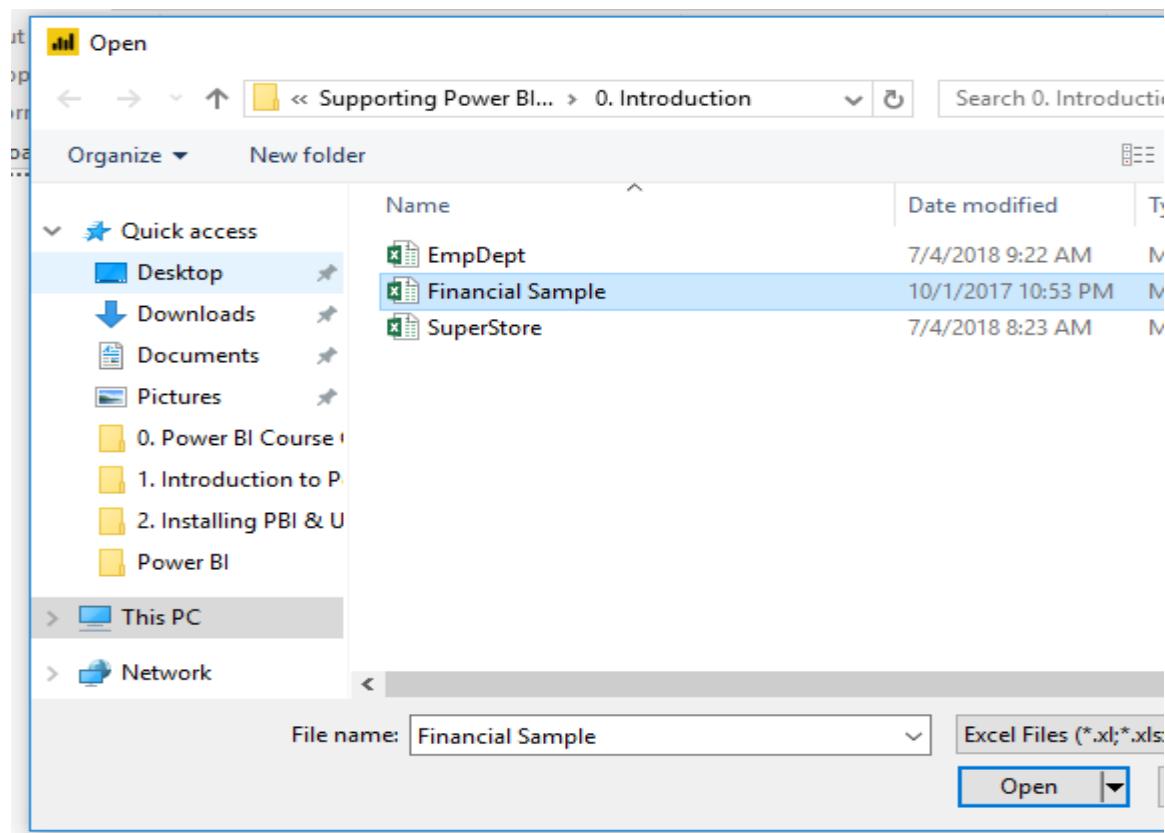
Once Click on Transform Data Power Query Software / Power Query Editor User Interface will open to perform ETL Activities as Shown Below



In Power Query first we need to extract the Data from Data Sources. Let's see how to extract the Data from Excel. To Extract the Data from Excel, Go to Home Tab → Click on New Source and select Excel as Source.

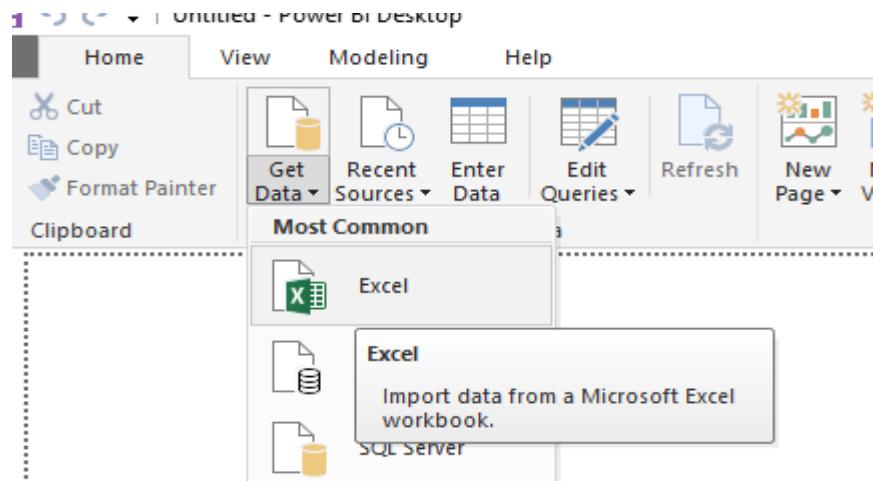


Once you select Excel it will ask to Browse where your Excel File is Located. Select your Excel workbook from the Open dialog that appears → Select Open

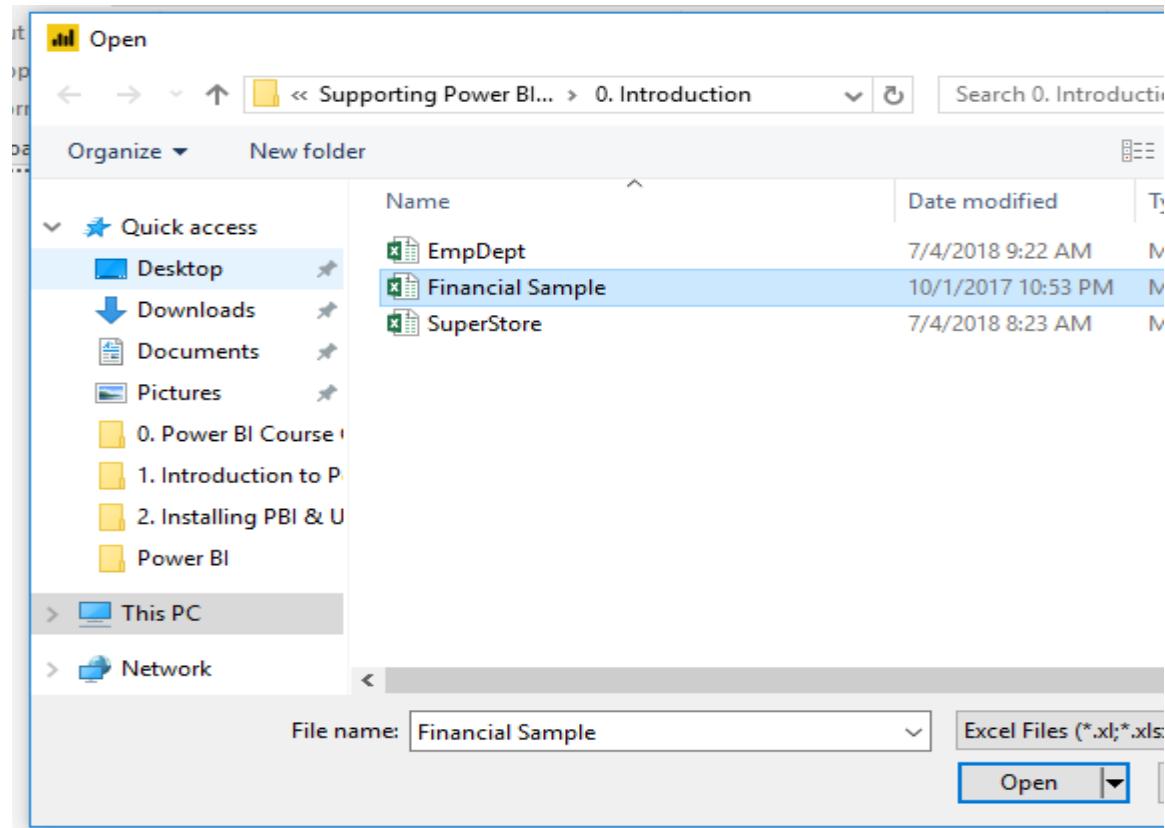


## Connecting to Excel

Select Get Data > Excel from the Home ribbon.



Select your Excel workbook from the Open dialog that appears → Select Open



Power BI Desktop presents the tables and other data elements from the workbook in the

Navigator window. When you select a table in the left pane, a preview of the data appears in the right pane.

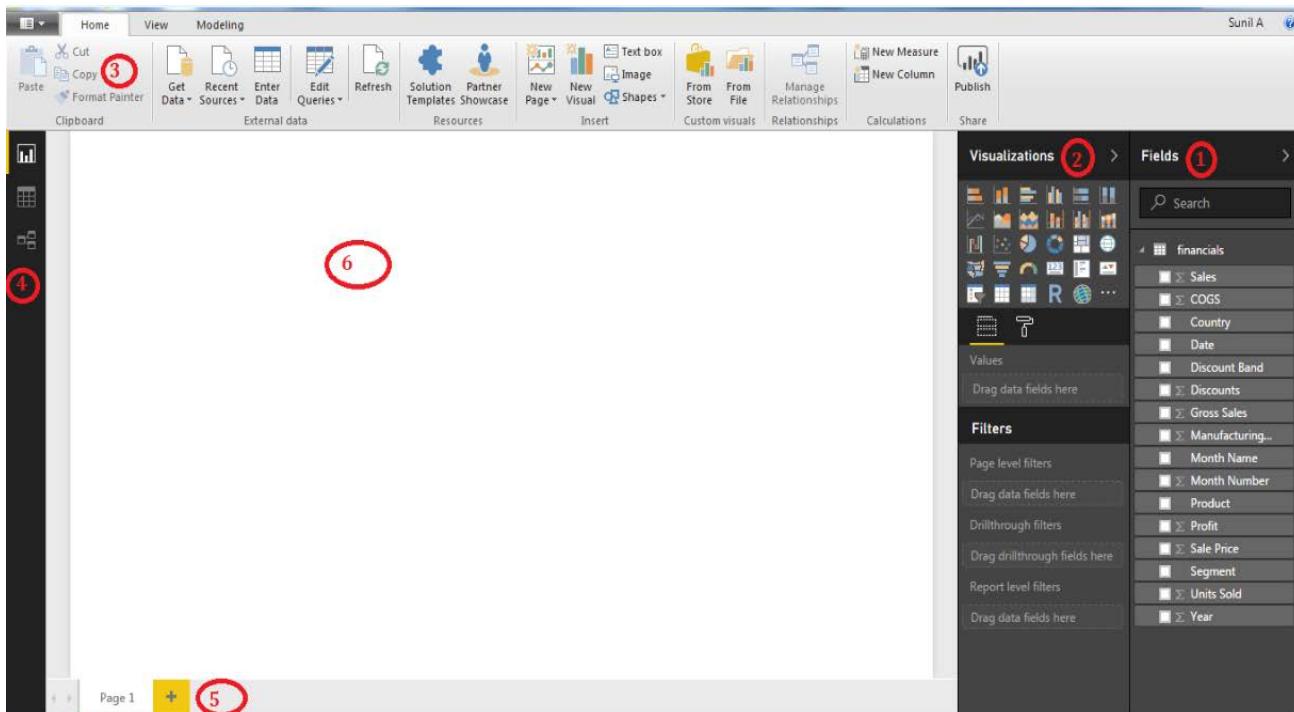
The screenshot shows the Power BI Navigator window. On the left, there is a file tree with 'Financial Sample.xlsx [2]' selected, containing 'financials' (which is checked) and 'Sheet1'. On the right, a preview of the 'financials' table is displayed, showing columns for Segment, Country, and Product. The table data includes various market segments like Government, Midmarket, and Enterprise, along with countries like Canada, Germany, France, Mexico, and the United States of America, and products like Carretera, Montana, and Paseo. At the bottom of the preview area are navigation arrows and a scroll bar. Below the preview are three buttons: 'Load' (highlighted in yellow), 'Edit', and 'Cancel'.

Segment	Country	Product
Government	Canada	Carretera
Government	Germany	Carretera
Midmarket	France	Carretera
Midmarket	Germany	Carretera
Midmarket	Mexico	Carretera
Government	Germany	Carretera
Midmarket	Germany	Montana
Channel Partners	Canada	Montana
Government	France	Montana
Channel Partners	Germany	Montana
Midmarket	Mexico	Montana
Enterprise	Canada	Montana
Small Business	Mexico	Montana
Government	Germany	Montana
Enterprise	Canada	Montana
Midmarket	United States of America	Montana
Government	Canada	Paseo
Midmarket	Mexico	Paseo

You can select the Load button to import the data, or if you want to edit the data using Query Editor before bringing it into Power BI Desktop, select the Edit button.

When you load the data, Power BI Desktop displays the Load window and displays the activity associated with loading the data.

- ✓ Once you load data into Power BI Desktop you will see the below screen.



The Visualizations and Fields pane can be collapsed by selecting the small arrow along the edge, providing more space in the Report view to build cool visualizations.

## Understanding Power BI Desktop User Interface (UI)/ Report View

### 1. Fields Pane

- ✓ Fields Area will contain the tables with columns that are loaded into Power BI Desktop.

### 2. Visualizations pane

The Visualizations pane will contain list of visualizations available in Power BI. And you can also download a custom visual from the visuals gallery. Here you can change visualizations, customize colors or axes, apply filters, drag fields, and more.

### 3. Ribbon

- ✓ It will contain the options that are associated with reports and visualizations.

### 4. Views

- ✓ Report View
- ✓ Data View
- ✓ Relationships View

### 5. Pages Tab

- ✓ The Pages tab area along the bottom, which lets you select or add a report page.

## 6. Canvas

- ✓ Canvas is the place where visualizations are created and arranged.

### With Power BI Desktop, you can do

Data Preparation → Query Editor  
Data Modeling → Relationship View+ Data View  
Data Visualization → Report View

### Get and Shape or Transform data - Power Query

- ✓ The Power BI Desktop makes discovering data easy. You can import data from a wide variety of data sources.
- ✓ After you connect to a data source, you can shape the data to match your analysis and reporting needs.

### Create relationships and enrich your data model with new measures and data formats - Power Pivot

- ✓ When you import two or more tables, oftentimes you'll need to create relationships between those tables.
- ✓ The Power BI Desktop includes the Manage Relationships dialog and the Relationships views, where you can use Auto detect to let the Power BI Desktop find and create any relationships, or you can create them yourself.
- ✓ You can also very easily create your own measures and calculations or customize data formats and categories to enrich your data for additional insights.

### Create reports - Power View

- ✓ The Power BI Desktop includes the Report View. Select the fields you want, add filters, choose from dozens of visualizations, and format your reports with custom colors, gradients and several other options.

Power View = Fields + Visualizations + Report View.

### Save your reports

- ✓ With the Power BI Desktop, you can save your work as a Power BI Desktop file. Power BI Desktop files have ".pbix" extension.
- ✓ Power BI Report template (.pbix) file, which contains the report definition minus the dataset.

### Upload or publish your reports

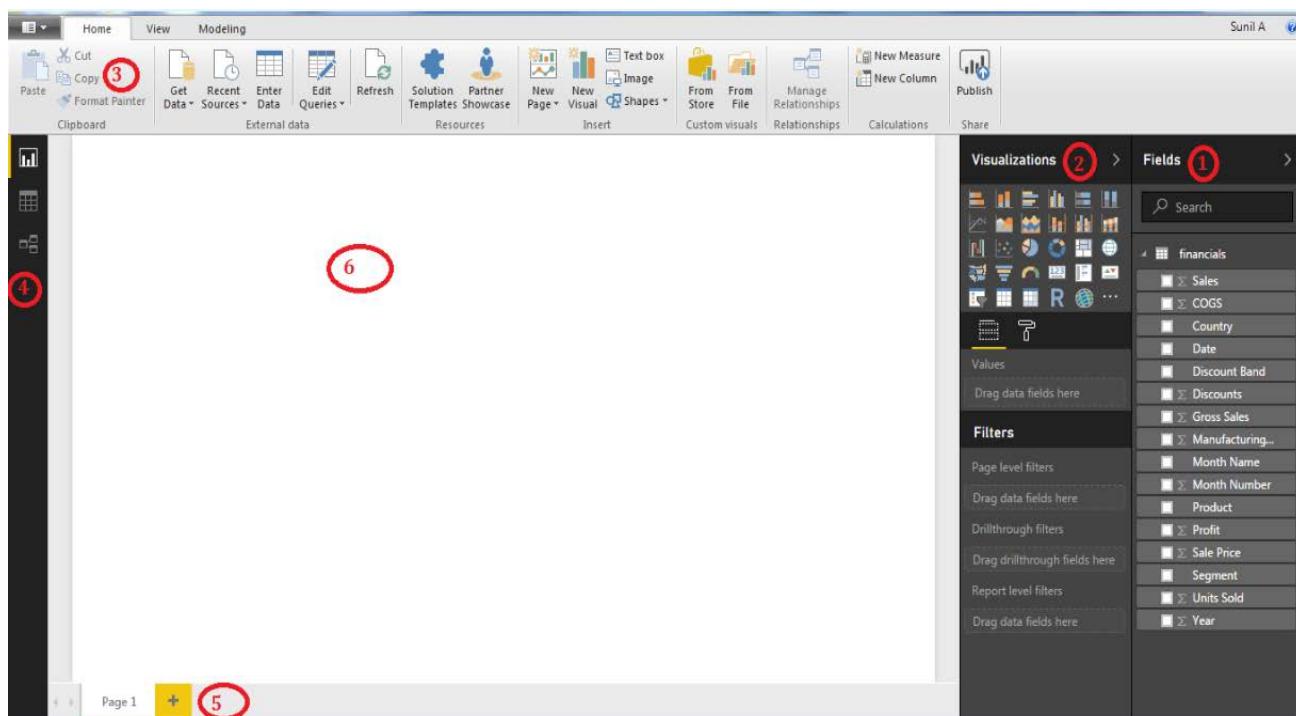
- ✓ You can upload the reports you created and saved in the Desktop to your Power BI Service or Power BI Report Server.

## Report View / Power View in Power BI Desktop

Power BI Desktop includes Report View, where you can create any number of report pages with visualizations. Report View provides pretty much the same design experience as a report's Editing View in the Power BI service. You can move visualizations around, copy and paste, merge, etc.

The difference is, when using Power BI Desktop, you can work with your queries and model your data to make sure your data supports the best insights in your reports. You can then save your Power BI Desktop file wherever you like, whether it be your local drive or to the cloud.

Once you load data into Power BI Desktop you will enter into the Report View and you will see the below screen.



The Visualizations and Fields pane can be collapsed by selecting the small arrow along the edge, providing more space in the Report view to build cool visualizations.

## Understanding Power BI Desktop User Interface (UI)/ Report View

### 1. Fields Pane

- ✓ Fields Area will contain the tables with columns that are loaded into Power BI Desktop.

### 2. Visualizations pane

The Visualizations pane will contain list of visualizations available in Power BI. And you can also download a custom visual from the visuals gallery. Here you can change visualizations, customize colors or axes, apply filters, drag fields, and more.

### 3. Ribbon

- ✓ It will contain the options that are associated with reports and visualizations.

### 4. Views

- ✓ Report View
- ✓ Data View
- ✓ Relationships View

### 5. Pages Tab

- ✓ The Pages tab area along the bottom, which lets you select or add a report page.

### 6. Canvas

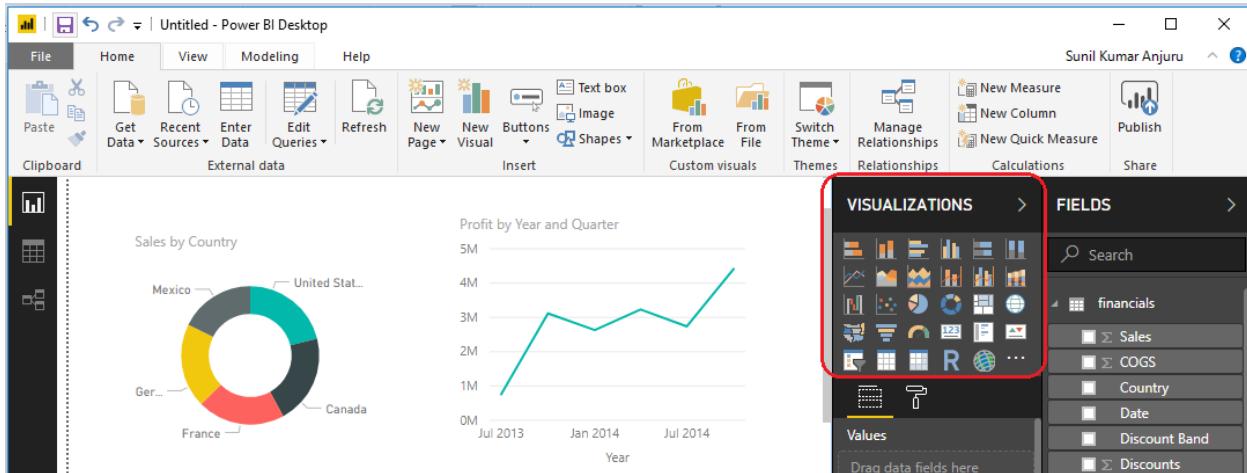
- ✓ Canvas is the place where visualizations are created and arranged.

You can switch between Report View, Data View, and Relationship View by clicking on the icons in the left hand navigation bar.



Once you've added some data to Power BI, you can add fields to a new visualization in the canvas.

To change the type of visualization, you can select it from the **Visualizations** group.

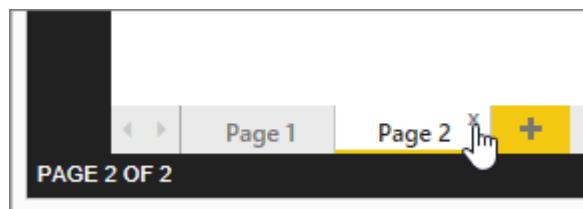


Be sure to experiment with different visualization types. It's important your visualization convey information in your data clearly.

A report will have at least one blank page to start. Pages appear in the navigator pane just to the left of the canvas. You can add all sorts of visualizations to a page, but it's important not to overdo it. Too many visualizations on a page will make it look busy and difficult to find the right information. You can add new pages to your report, just click **New Page** on the ribbon.



To delete a page, click the X on the page's tab at the bottom of the Report View.

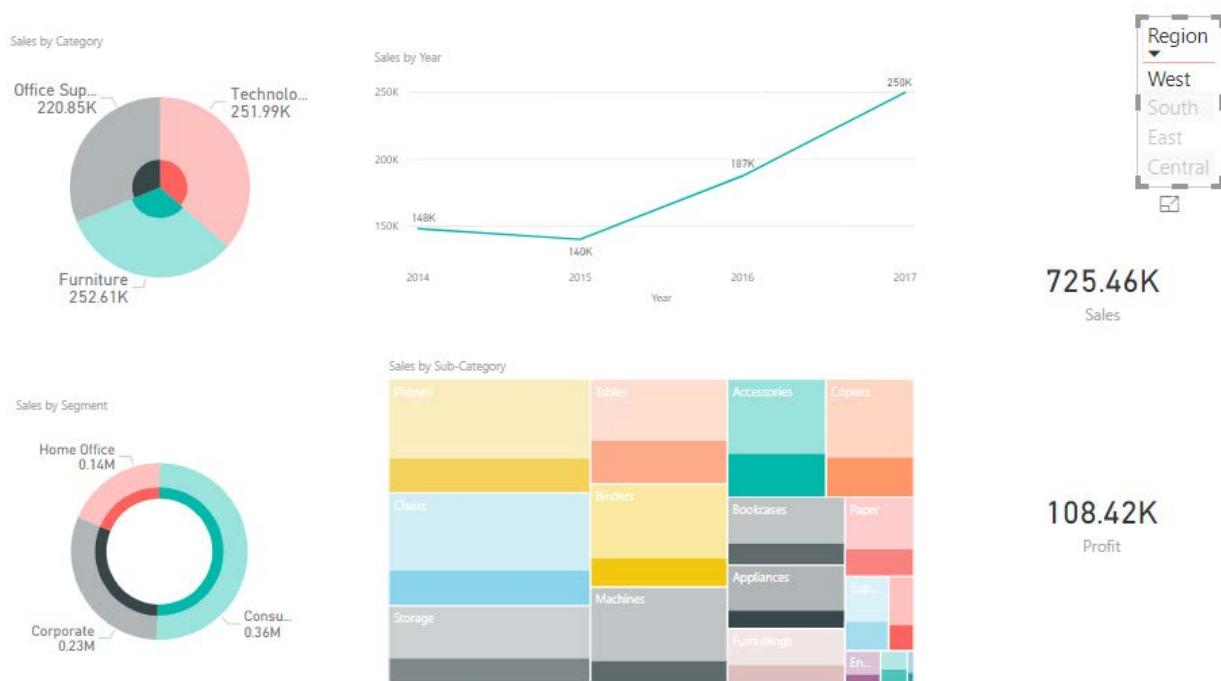


## Power BI Visual Interactions (Ad-hoc filtering and highlighting)

Power BI visuals are interactive with each other. Selecting an item in a visual will effect on the display of another Visual. Sometimes this effect is highlighting items in another Visuals, and sometimes filtering values in the other visuals. By default, all visuals in a report page are interactive with each other. However, this interactivity can be controlled and modified.

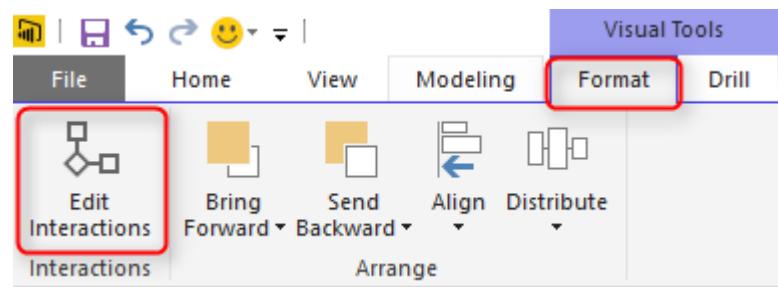
### Default Behavior

All visuals in Power BI report page are interactive with each other. If you click on a chart other charts will be highlighted or filtered.

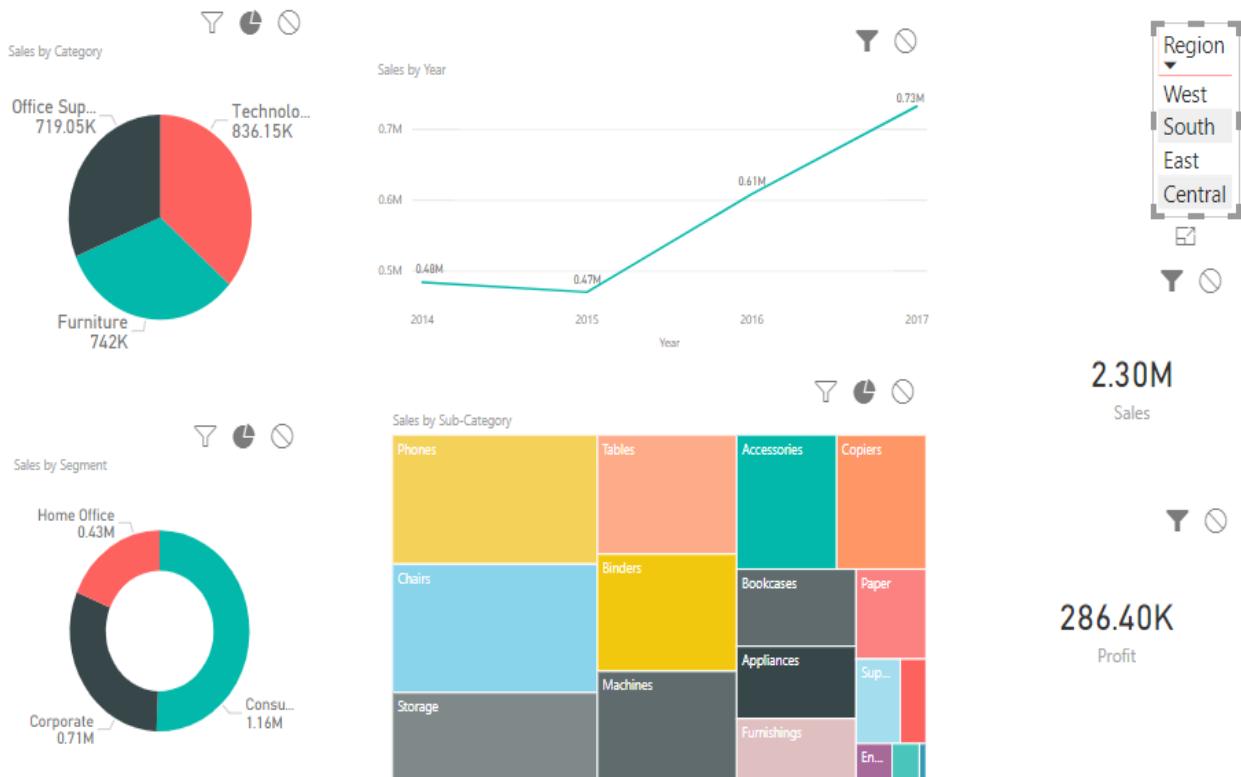


### Changing the Interaction

To change the interaction of a visual with other visuals, select the main Visual (the chart that you want to control effect of that on other Visuals), and then from menu select Format, click on Edit Interactions.



You will now see all other visuals in the page with two or three buttons on the top right hand side corner of each. These are controls of interaction. Based on the requirement we will select the interaction Type.



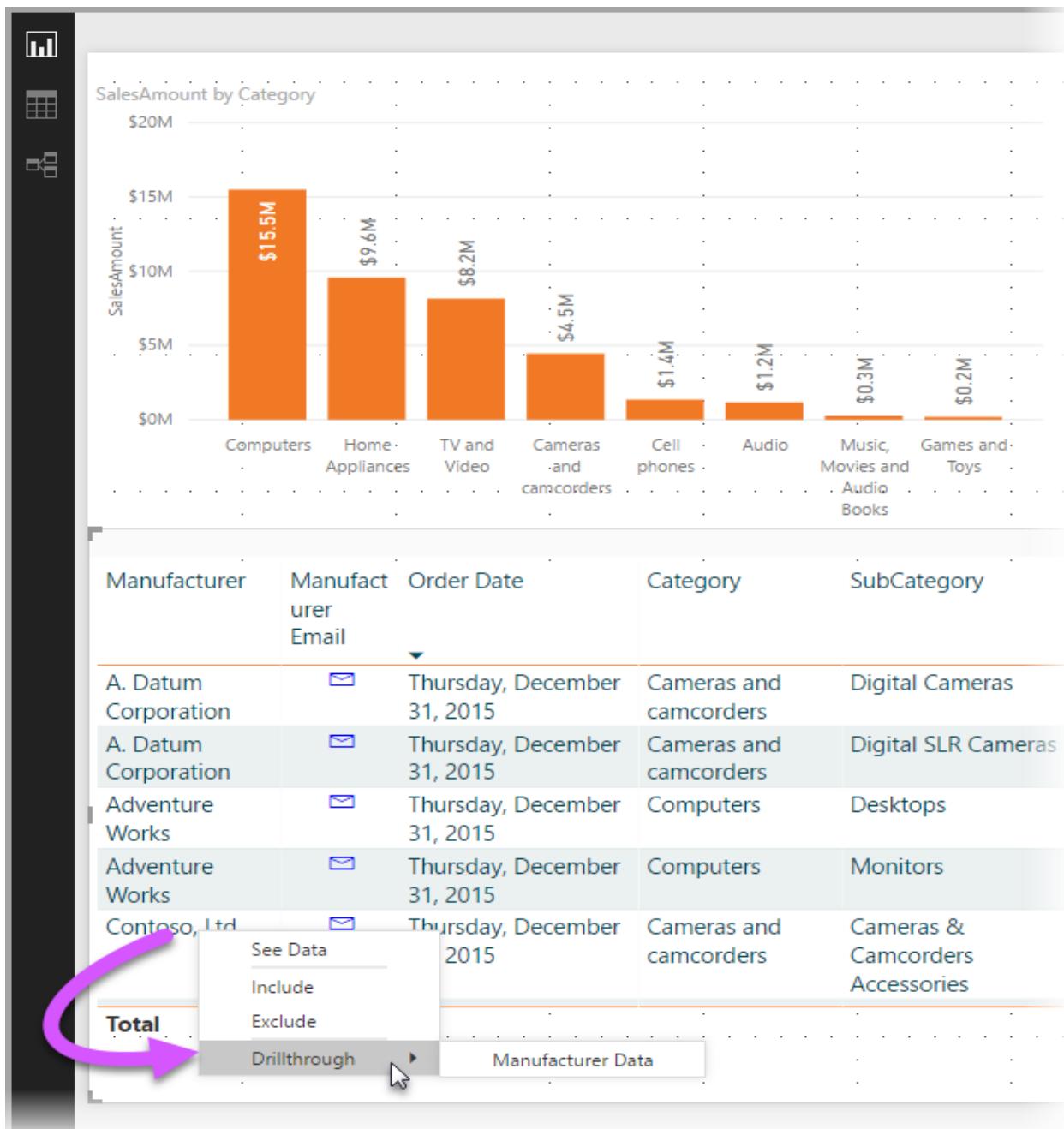
If you want to cross-filter the visualization, select the filter icon

If you want to cross-highlight the visualization, select the highlight icon

If you don't want to filter or highlight the Visualization, select the no impact icon

## Drillthrough in Power BI Desktop

With drillthrough in Power BI Desktop, you can create a page in your report that focuses on a specific entity - such as a supplier, customer, or manufacturer. Users can right-click on a data point in other report pages and drillthrough to the focused page to get details that are filtered to that context.

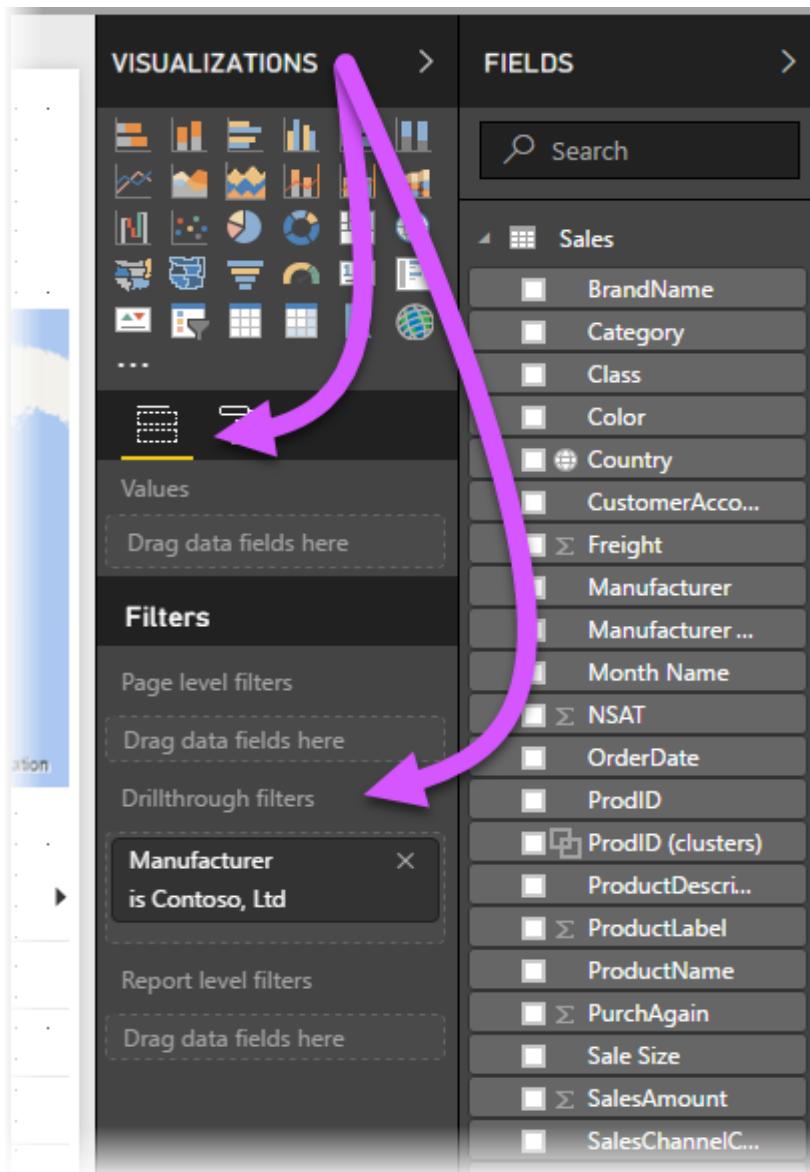


## Using drillthrough

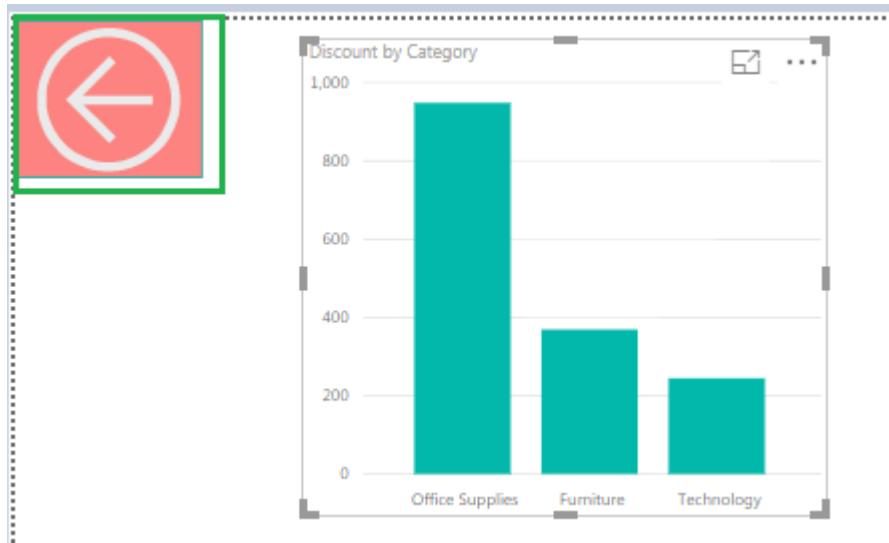
To use drillthrough, create a report page that has visuals you'd like to see for the type of entity on which you'll provide drillthrough.

For example, if you're interested in providing drillthrough for manufacturers, you might create a drillthrough page with visuals that show total sales, total units shipped, sales by category, sales by region, and so on. That way, when you drillthrough to that page, the visuals will be specific to the manufacturer you selected.

Then, on that drillthrough page, in the Fields section of Visualizations pane, drag the field about which you want to drillthrough into the Drillthrough filters well.



When you add a field to the Drillthrough filters well, Power BI Desktop automatically creates a back button visual. That visual becomes a button in published reports, and lets users who are consuming your report in the Power BI service easily get back to the report page from which they came (the page from which they selected to drillthrough).



## Filters in Power BI

Data filtering will involve taking out information that is useless to a reader or information that can be confusing. Generated reports and query results from database tools often result in large and complex data sets. Redundant or impartial pieces of data can confuse a user. Filtering data can also make results more efficient.

Whether you're using Desktop or Power BI service, the Filters pane displays along the right side of the report canvas.

### There are four types of filters

- Visual Level Filters
- Page Level Filters
- Report Level Filters
- Drillthrough Filters

### Visual Level Filters

These filters work on only an individual visualization, reducing the amount of data that the visualization can see.

### Page Level Filters

These filters work at the report-page level. Different pages in the same report can have different page-level filters. These filters work on all the visualizations in that page.

## Report Level Filters

These filters work on the entire report, filtering all pages and visualizations included in the report.

### Drillthrough filter

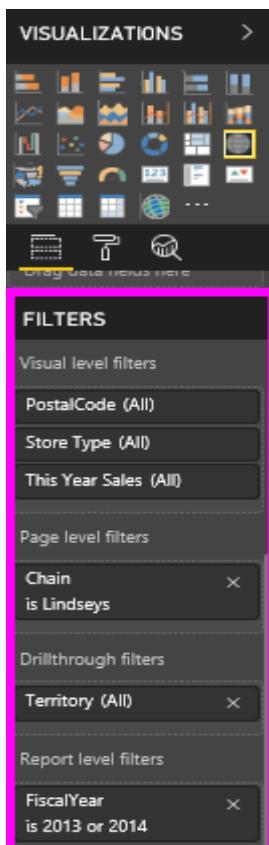
Drillthrough filter applies to a single entity in a report.

In above filters you will see the below filter Types based on requirement we will use any of these types.

- ✓ Basic Filtering
- ✓ Advanced Filtering
- ✓ Top N
- ✓ Relative Date Filtering

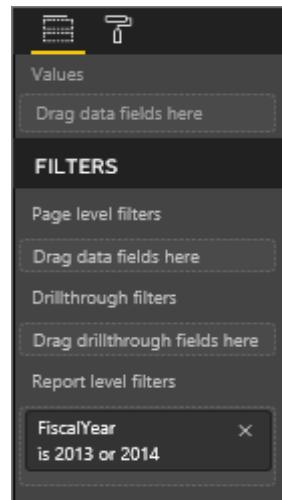
### Working with Filters

When a report is open in Desktop or in Power BI service Editing view, the Filters pane displays along the right side of the report canvas in the bottom half of the Visualization pane. If you don't see the pane, select the arrow in the top-right corner to expand it.

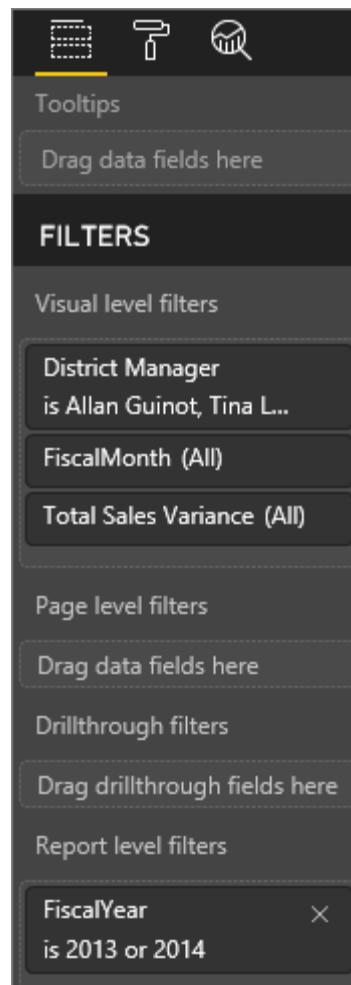


If no visual is selected in the canvas, then the Filters pane displays just the filters that apply to the entire report page or entire report, and any drillthrough filters (if any have been

set). In the example below, no visual is selected and there is no page level or drillthrough filters but there is a report level filter.



If a visual is selected in the canvas, you will also see the filters that apply to just that visual.



To display options for a particular filter, select the down arrow next to the filter name. In the example below, the report level filter is set to 2013 and 2014. And this is an example of basic filtering. To display the advanced options, select Advanced Filtering.



#### Clear a filter

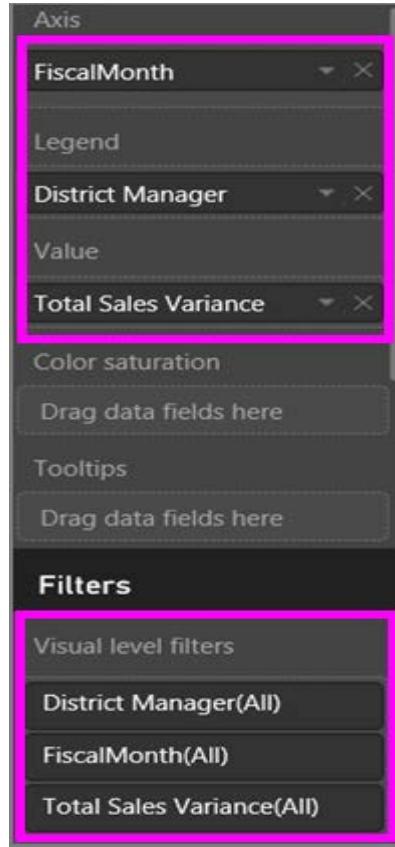
In either advanced or basic filtering mode, select the eraser icon to clear the filter.

#### Add a filter

In Desktop and in Power BI service Editing view, add a filter to a visual, page, drillthrough, or report by selecting a field from the Fields pane and dragging it into the appropriate filter well, where you see the words Drag fields here. Once a field has been added as a filter, fine-tune it using the Basic filtering and Advanced filtering controls (described below).

Dragging a new field into the Visual level filter area does not add that field to the visual, but it does allow you to filter the visual with this new field.

All the fields that are used to create a visualization are also available as filters. First, select a visual to make it active. The fields that are being used in the visual are listed in the Visualizations pane and in the Filters pane under the Visual level filters heading.



## Types of filters: text field filters

### List mode

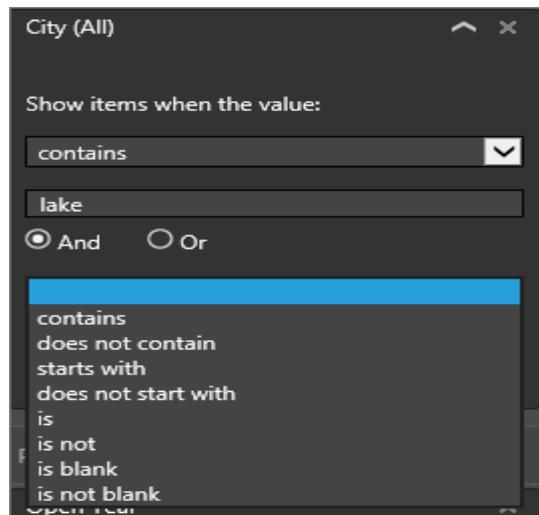
Ticking a checkbox either selects or deselects the value. The All checkbox can be used to toggle the state of all checkboxes on or off. The checkboxes represent all the available values for that field. As you adjust the filter, the restatement updates to reflect your choices.

Product	
is Amarilla or Carretera	☒
<input type="checkbox"/> (All)	
<input checked="" type="checkbox"/> Amarilla	94
<input checked="" type="checkbox"/> Carretera	93
<input type="checkbox"/> Montana	93
<input type="checkbox"/> Paseo	202
<input type="checkbox"/> Velo	109
<input type="checkbox"/> VTT	109
Advanced Filtering	

Note how the restatement now says "is Amarilla or Carretera"

## Advanced mode

Select Advanced Filtering to switch to advanced mode. Use the dropdown controls and text boxes to identify which fields to include. By choosing between **And** and **Or**, you can build complex filter expressions. Select the **Apply Filter** button when you've set the values you want.



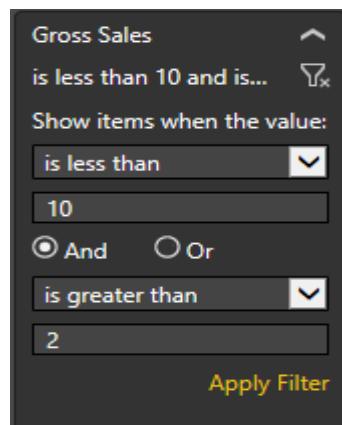
## Types of filters: numeric field filters

### List mode

If the values are finite, selecting the field name displays a list. See Text field filters > List mode above for help using checkboxes.

### Advanced mode

If the values are infinite or represent a range, selecting the field name opens the advanced filter mode. Use the dropdown and text boxes to specify a range of values that you want to see.



By choosing between **And** and **Or**, you can build complex filter expressions. Select the **Apply Filter** button when you've set the values you want.

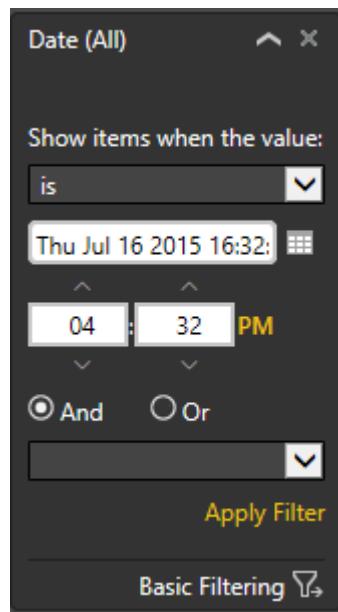
## Types of filters: date and time

### List mode

If the values are finite, selecting the field name displays a list. See Text field filters > List mode above for help using checkboxes.

### Advanced mode

If the field values represent date or time, you can specify a start/end time when using Date/Time filters.



## Applying Filters on Number Datatype Columns

1. Create a Visual to Show ENAME & DEPTNO where DEPTNO =10
2. Create a Visual to Show ENAME , SAL & DEPTNO Where SAL=3000
3. Create a Visual to Show ENAME , SAL & DEPTNO Where SAL<>3000
4. Create a Visual to Show ENAME , SAL & DEPTNO Where SAL<3000
5. Create a Visual to Show ENAME , SAL & DEPTNO Where SAL<=3000
6. Create a Visual to Show ENAME , SAL & DEPTNO Where SAL>3000
7. Create a Visual to Show ENAME , SAL & DEPTNO Where SAL>=3000
8. Create a Visual to Show ENAME , SAL, COMM & DEPTNO Where COMM IS BLANK
9. Create a Visual to Show ENAME , SAL, COMM & DEPTNO Where COMM IS NOT BLANK
10. Create a Visual to Show ENAME & DEPTNO where DEPTNO =10 OR DEPTNO=20

### OR Operator

Cond 1	Cond 2	Result
True	True	True
True	False	True
False	True	True
False	False	False

11. Create a Visual to Show ENAME , SAL & DEPTNO Where SAL Between 3000 AND 5000

SAL Between 3000 AND 5000 **Equivalent to** SAL>=3000 AND SAL<=5000

### AND Operator

Cond 1	Cond 2	Result
True	True	True
True	False	False
False	True	False
False	False	False

12. Show Top 2 Departments based on Sum of Salary

## Applying Filters on Text Columns

1. Create a Card Visual to Show Sum Of Sales Where Region = "Central"
2. Create a Card Visual to Show Sum Of Sales Where State = "New York" Or "Texas"
3. Create a Card Visual to Show Sum Of Sales Where State = "New York" Or "Texas" OR State = "Washington"
4. Create a Card Visual to Show Sum Of Sales Where State <> "New York"
5. Show Customer Name wise Sales where Customer Name Starts with "S"
6. Show Customer Name wise Sales where Customer Name Starts with "Allen"
7. Show Customer Name wise Sales where Customer Name Starts with "R" OR Customer Name Starts with "S" OR Customer Name Starts with "T"
8. Show Customer Name wise Sales where Customer Name Does Not Starts with "S"
9. Show Customer Name wise Sales where Customer Name Contains "Allen"
10. Show Customer Name wise Sales where Customer Name Does Not Contains "Allen"
11. Show ENAME, JOB & DEPTNO where JOB is Blank
12. Show ENAME, JOB & DEPTNO where JOB IS NOT Blank
13. Show Top 5 Customer Names Based on Sales.

## Hierarchies and Drill-Down in Power BI

### Hierarchy

The ordering and grouping of Attributes or Dimensions called as Hierarchy.

Hierarchy is an ordered sequence of dimensions, which is very helpful for analysis of data.

Hierarchy is a relationship, where one of the items is "detail" of another one. Like Continent / country / State /city, Year / month / day.

### Examples

#### Time Hierarchy

Year → Quarter → Month → Day

#### Geographic Location Hierarchy

Continent → Country → State → City

#### Product Hierarchy

Category → Sub-Category → Product Name

Instead of creating the visualizations for each dimension individually we will create a Hierarchy for Related Dimensions (Category → Sub-Category → Product Name) and will use that hierarchy in the visualizations.

Power BI Hierarchies provides you the drill down action to the Power BI Visualizations.

When a visual has a Hierarchy, it enables the ability to drill down for additional relevant details.

One very common hierarchy is a date hierarchy, which is used to show data summarized by year, then all the values for the quarters, then the values for three months in each quarter and each month which at the lowest level includes all the values for dates.

There are other hierarchies which may also exist in data, such as Geographic Location Hierarchy, Product Hierarchy.

Geographic Location Hierarchy contains Continent which includes Countries which include states or provinces which include cities.

Product Hierarchy Contains Categories which include Sub-Categories which include Product Names.

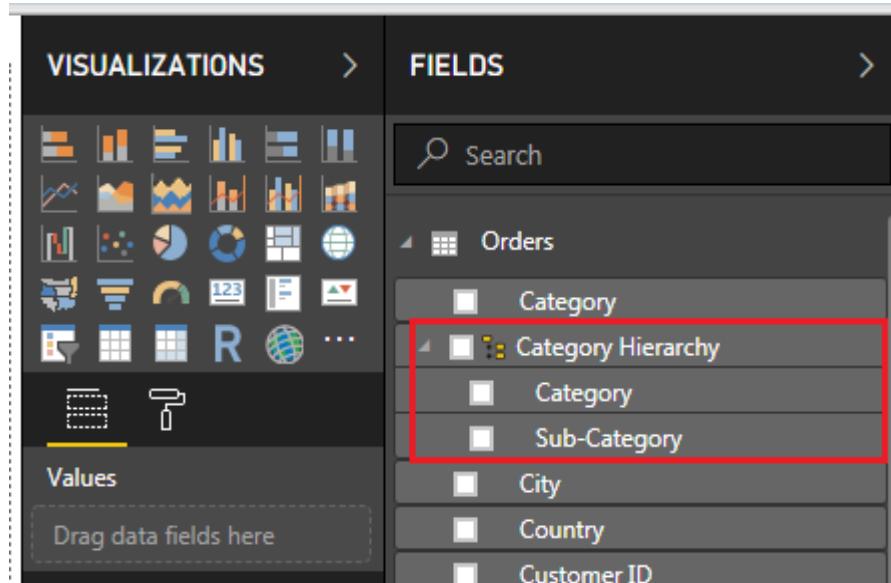
### Creating Hierarchy in Power BI

Before we build a hierarchy, we'll need to know the levels that comprise the hierarchy. In our example, the levels are Category → Sub-Category → Product Name

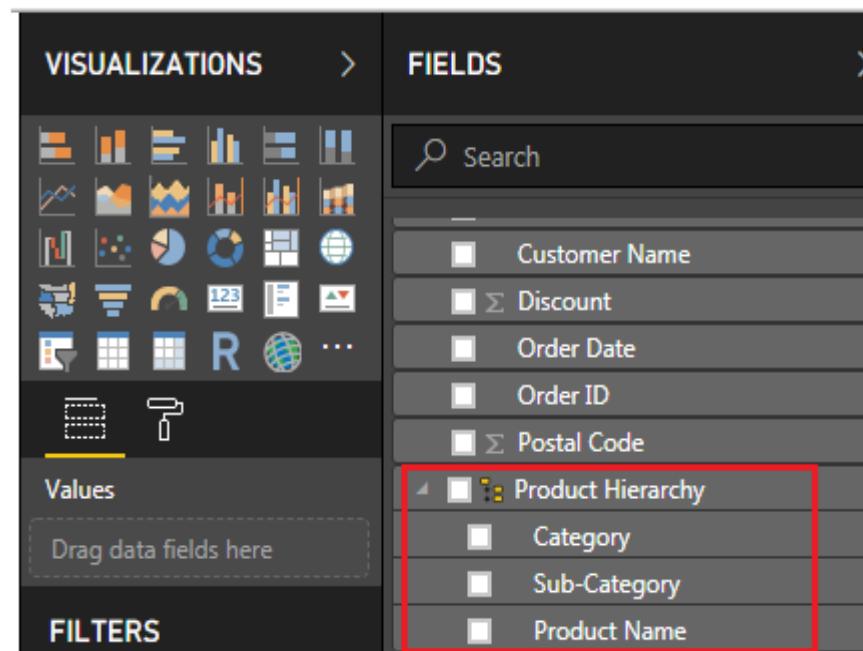
There is couple of ways to create a hierarchy in power bi.

## First approach to Create Hierarchy in Power BI

After we know the hierarchy levels, we'll use simple drag/drop techniques to create the hierarchy. Since Category is our first level in the hierarchy, let's look for Sub-Category and place it right on top of Category. Notice that there is a new item was created called "Category Hierarchy".

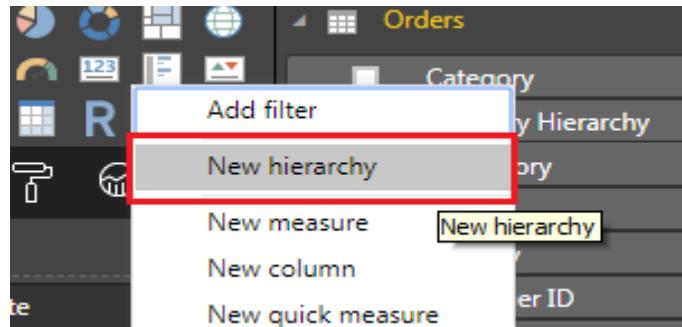


Finally, we'll drag and drop Product Name on the Category Hierarchy we just created and rename Category Hierarchy to Product Hierarchy by right-clicking Category Hierarchy and selecting "Rename".

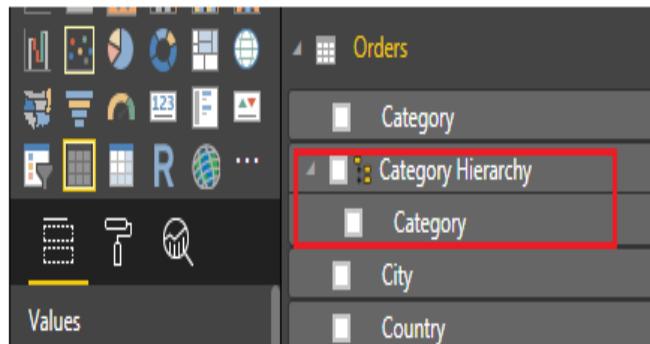


## Second Approach to Create Hierarchy in Power BI

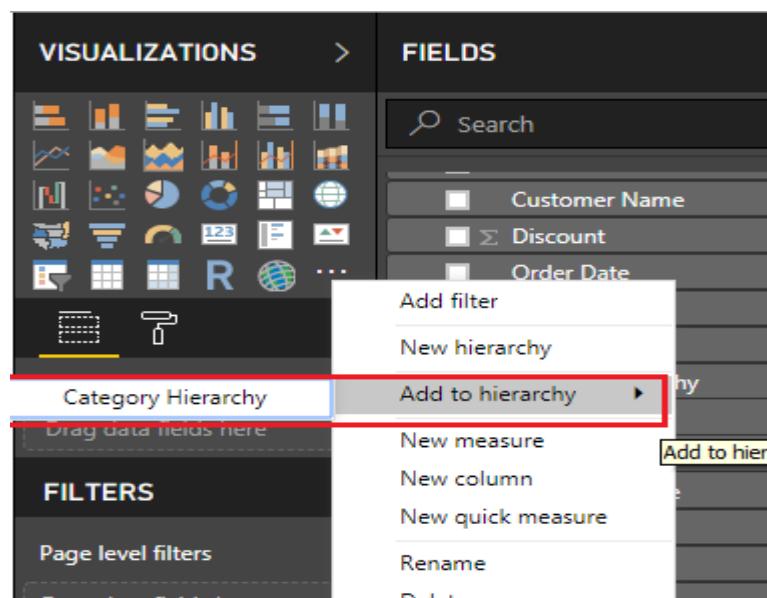
Right clicking on the ellipse next to any field let's say Category in a table, displays a context menu, and the second item on the context menu is New Hierarchy.



When you select this, a new hierarchy will be created with that field name suffixing with Hierarchy (Category Hierarchy).



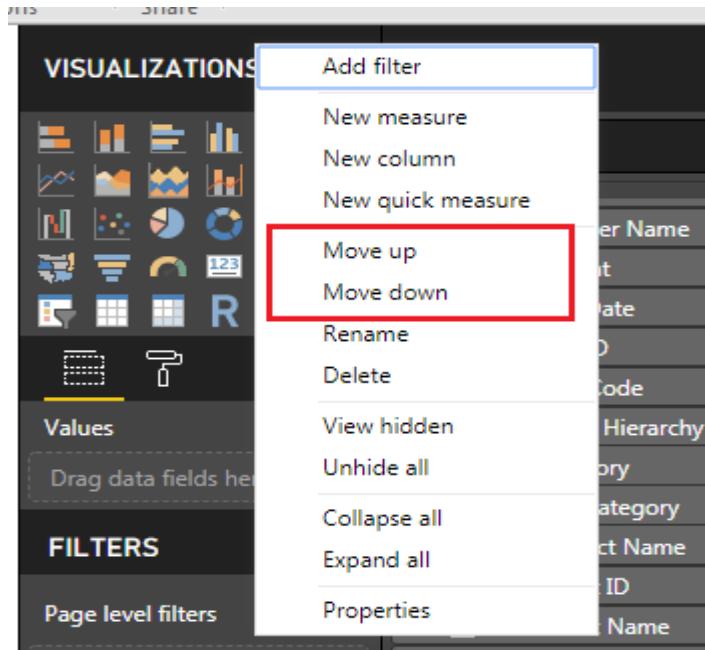
If you want to add other fields to the above hierarchy (Category Hierarchy), select the field and right click on it and select add to the hierarchy.



## Change Hierarchy Levels in Power BI

Please be careful while configuring the hierarchy levels. For instance, if you have the Sub-Category column above the Category, Then Level 1 will be Sub-Category, level 2 will be Category.

Please select the field that you want to change the position or level. Next, right-click on it and select the Move Up option (or move down) from the context menu. Alternatively, you drag and drop the field at the required position.

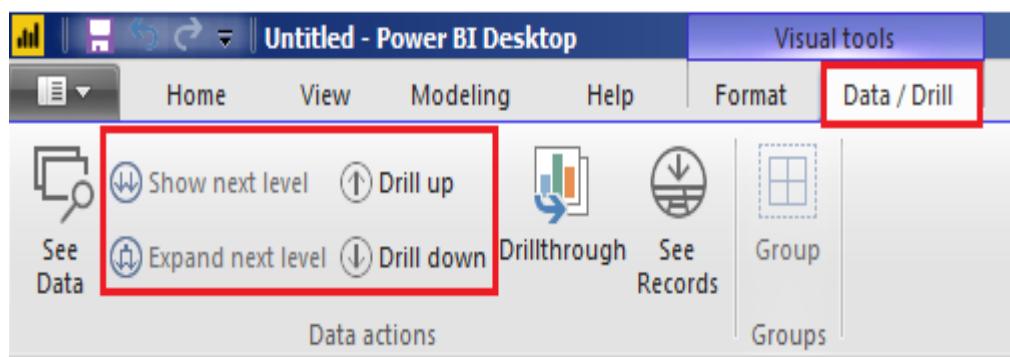


## Drill-Up and Drill-Down Reports in Power BI

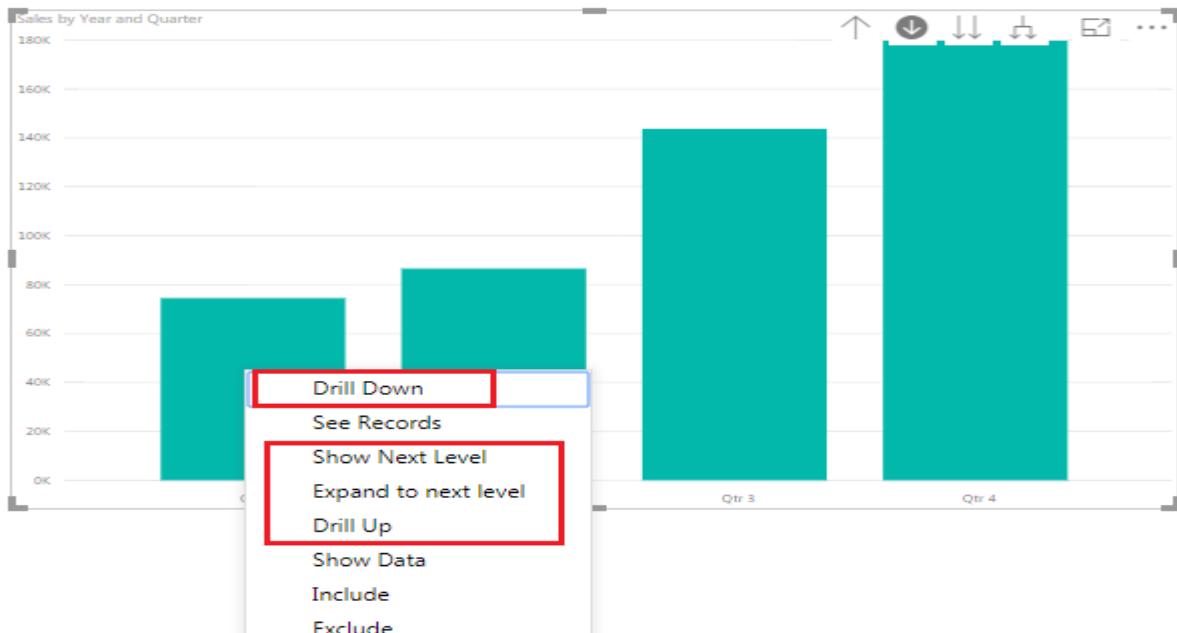
Power BI Drill-Up and Drill-Down Reports help you to drill down the reports to multi-levels. Using this you can see the data at each level.

You can perform Drill up, and Drill down using any of the three options. When you add a Hierarchy to the Visualization then only you can see these options.

1. In Ribbon from Home Tab Go to the Data / Drill tab to see these properties.



- If your data has multiple levels then these buttons  will be displayed in the visualization.
- Right-click on the report will show the context menu. You can use this menu to select the levels.

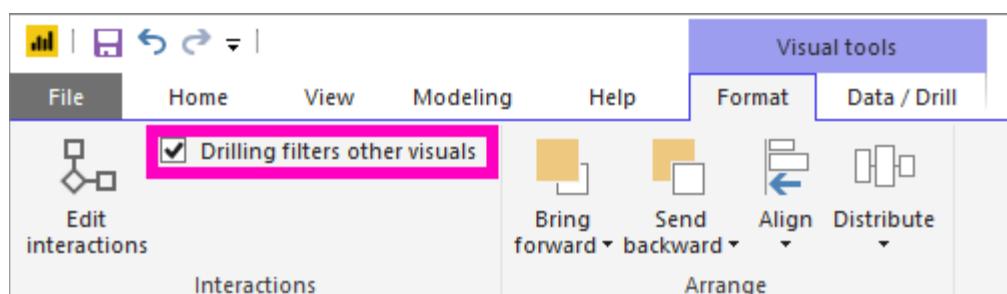


**NOTE:** There is a difference between Show Next Level, and Drill Down. Show Next level will show you the complete data in the next level but the Drill down option will show you the selected data.

### Drilling filters other visuals

As you work in Drill mode, you get to decide how drill down and expanding impacts the other visualizations on the page.

By default, drilling will filter other visualizations in a report. But this feature can be disabled in Power BI Desktop and Power BI service if you needed.



In Desktop, select the Format tab and select the checkbox for Drilling filters other visuals.

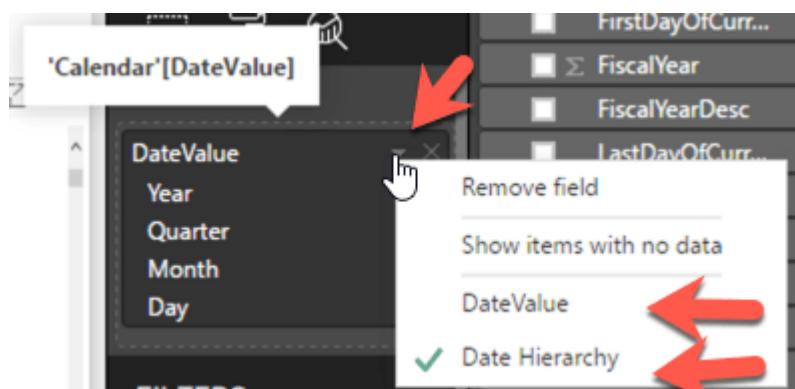
## Date Hierarchy in Power BI

As with most data analysis tasks, working with time can be the most challenging detail to get correct. Power BI has built-in functionality that will display a drillable hierarchy of year, quarter, month and days when a column, formatted as a date, is dropped onto visualization.

If you are creating a table and drag a field, formatted as a date, you do not get a single date but the hierarchy as pictured below. This creates a built-in way to drill down through the year on various graphs and tables. On visualization, this allows you to provide different views of your data over time. You can also remove a level from the list by selecting the 'X' next to the level depending on your needs.

The screenshot shows the Power BI interface. On the left, there is a table visual with columns labeled Year, Quarter, Month, and Day. The table contains data for February and March of 2014. A red arrow points from the table to the 'Values' section of the Fields pane. The 'Values' section is highlighted with a red border and lists 'Order Date', 'Year', 'Quarter', 'Month', and 'Day'. Below this is a 'FILTERS' section with 'Visual level filters' and a button 'Order Date - Day (All)'. To the right of the Fields pane is a list of fields: Customer Name, Discount, Order Date (which is checked), Order ID, Postal Code, Product Hierarchy (with Category, Sub-Category, Product Name, Product ID, Product Name, Profit, Quantity, Region, Row ID, Sales), Fiscal Year, Fiscal Year Desc, and Last Day of Current Year.

However, this behavior is the default. If you want to show the actual date field, you have to click on the down arrow next to the date field as shown below. This allows changing the selection from the Date Hierarchy to the field that contains your Date Values.



## Considerations

If adding a date field to visualization does not create a hierarchy, it may be that the "Date" field is not actually saved as a date. If you have owned the data set, open it in Data view in Power BI Desktop, select the column that contains the date, and in the Modeling tab change the Data Type to Date or Date/Time as shown in the picture below. If the report has been shared with you, contact the owner to request the change.

The screenshot shows the Power BI Desktop interface with the 'Modeling' tab selected in the ribbon. A red box highlights the 'Data type: Date/Time' dropdown menu, which is open, showing options like 'Format' and 'Auto'. Below the ribbon, a table is displayed with three columns: 'CountryRegionCode', 'CurrencyCode', and 'ModifiedDate'. The 'ModifiedDate' column is highlighted with a yellow background. Red boxes highlight the first row of the table and the first cell of the 'CountryRegionCode' column. The table data is as follows:

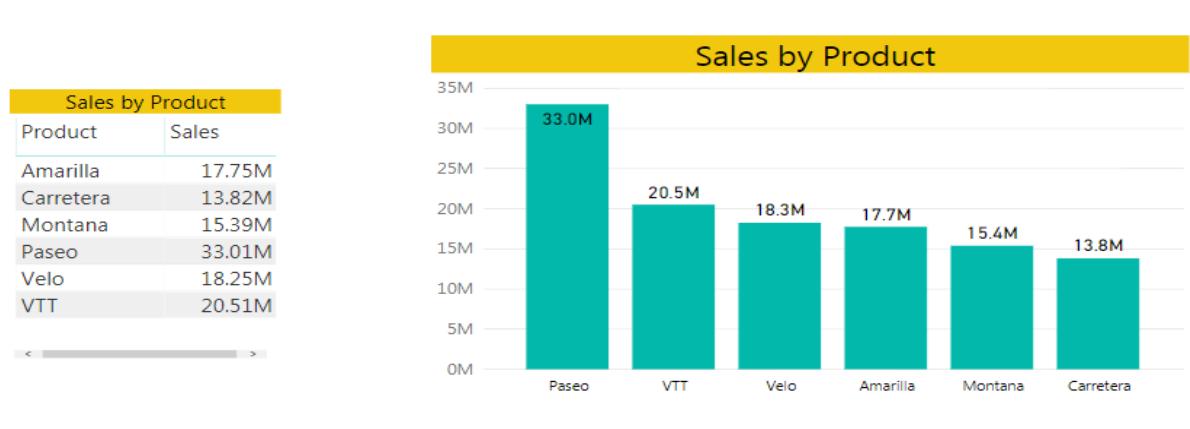
CountryRegionCode	CurrencyCode	ModifiedDate
AE	AED	2/8/2014 10:17:21 AM
AR	ARS	2/8/2014 10:17:21 AM
AT	ATS	2/8/2014 10:17:21 AM
AU	AUD	2/8/2014 10:17:21 AM
BB	BBD	2/8/2014 10:17:21 AM

## Visualizing Data

Visualizing the data means presenting the data in graphical or pictorial format format. Visualizing data is one of the core parts of Power BI - a basic building block as we defined it earlier in this course.

### Why Visualizations

- Creating visuals is the easiest way to find and share your insights with Clients.
- You can get the insights easily when you present in the form of visuals rather than text table. In the below figure you can easily identify the highest sales produced product in the visual rather than text table.



Once you have connected to the data sources and created a data model, the next step is to create visualizations. One of the things that set Power BI apart from other visualization tools is that you can create interactive visualizations—that is, visuals, that can interact with each other by cross-filtering the underlying data.

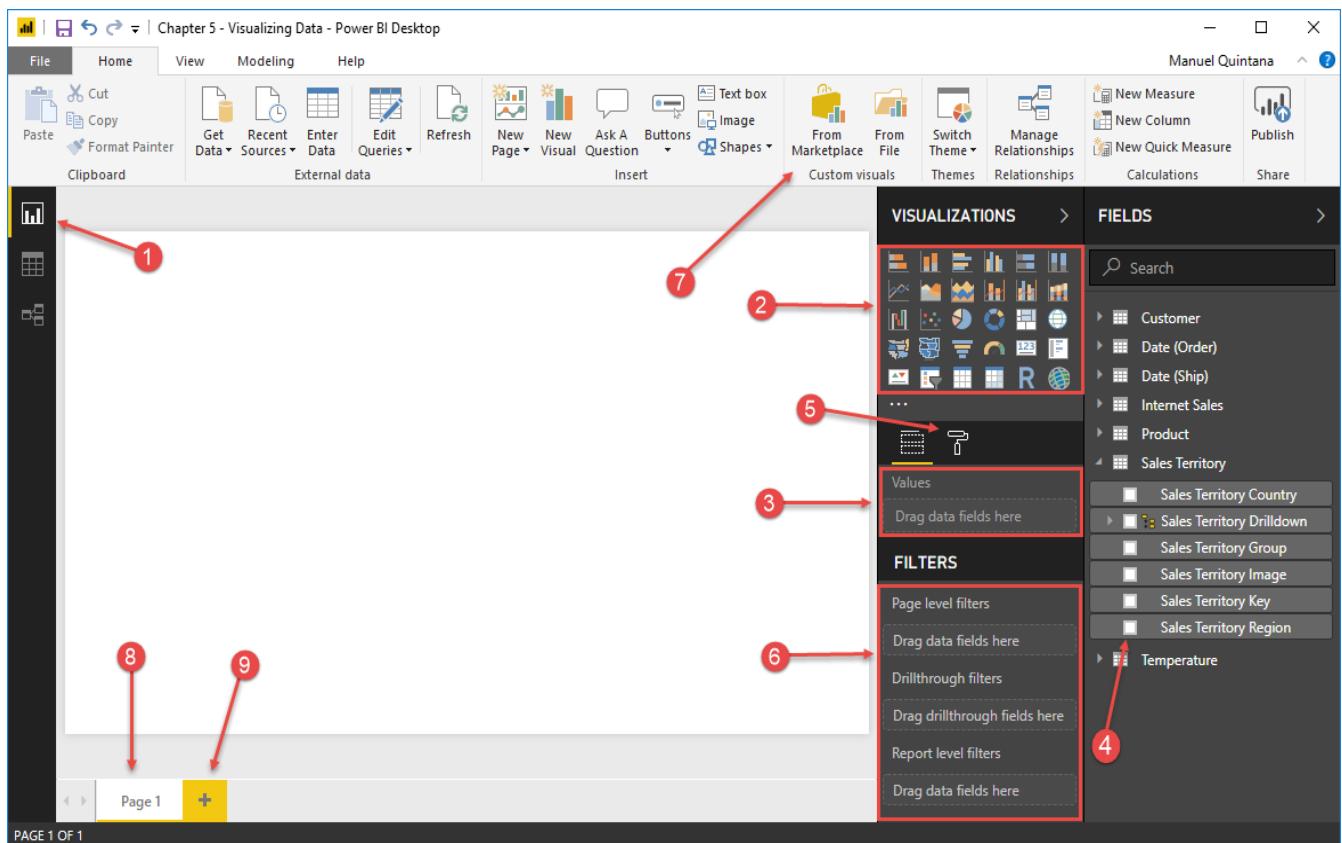
Up to this point, you have spent some time importing data and modeling it to your specifications. Now, we will begin to visualize the data in efficient and effective ways. The most common association with Power BI for consumers is the ability to create very impactful visualizations of data, and there are many options available to do this. Here, we will look at all the various options that are available to you within the Power BI Desktop application. Additionally, we will take a brief glimpse at the additional visualization options that are available through the Custom Visuals Marketplace. In this section we will discuss about below topics.

- Data visualization basics
- Visuals for Filtering
- Visualizing Categorical Data
- Visualizing Trend Data / Date and Time Data
- Visualizing KPI Data
- Visualizing Tabular Data
- Visualizing Geographical Data
- Leveraging Power BI Custom Visuals

There are Around 30 readily available visuals in the Power BI Desktop application. We will be exploring most of them and how they best work with certain types of Data Sets.

## Data Visualization Basics

As soon as you launch the Power BI desktop application and close out of the initial splash screen, you will find yourself in the Report View, which is where we will stay for the duration of this chapter. In the previous chapter, you explored the Relationship view as well as the Data view, but these areas are not necessary for the visualization work we will be doing. There are many items of interest in this initial Report view area that we need to discuss so that we can work efficiently.



### 1. Report view

This is the button that will place us in the Report canvas and allow us to create visuals.

### 2. Visuals area / Visualizations Pane

This is where we can choose which visual, we would like to use. Once custom visuals are added, they will appear here as well.

### 3. Field Area / Fields Well

This area will change depending on the visual but it is where we place the fields we will use within the selected visual.

### 4. Field pane

These are all the available fields we have to choose from to add to our visuals.

## 5. Format area

Here is where we can decide on many things specific to either the entire report page or the selected visual, such as text size, font style, titles, and so on.

## 6. Filters area

This is where we can apply filters of various scopes

- **Page-level filters:** Any filters applied here will affect every single visual on the selected page.
- **Drill through filters:** This option allows users to pass a filter value from a different report page to this one.
- **Report-level filters:** Filters applied here will affect every single visual for the entire Power BI report.
- **Visual-level filters:** This category will only appear when you have a visual selected, and the applied filters will only affect the selected visual.

## 7. Custom visuals

By selecting this button, you will have a menu appear that has access to all the custom visuals from the Microsoft store. You can then add whichever visual you would like to the Visuals area.

## 8. Report page

Here is where you can select which report page you would like to work with. Each page has a limited work area where we can use to create visuals, so it is common to have more than one page in a Power BI report.

## 9. Add Report page

By selecting this symbol, you can add a new report page to add more area in which we can add visuals.

Now that we have familiarized ourselves with the Report page features and layout, it's time to start visualizing!

## Visuals for filtering

Filtering the data that users will see within a Power BI report is the most effective way to answer very specific questions about that data, and there are many ways to accomplish this. One of Power BI's best features is its default capability to allow users to interact with a visual, which will then apply that as a filter to the rest of the visuals on that page, and this is known as interactive filtering or Visual Interactions. This behavior really puts the power into the user's hands, and they can decide how they want to filter the visuals. This now makes a report so much more robust because it can answer so many more questions about the data. Along with this functionality, we, report developers, can

add more explicit forms of filtering using the Slicer visual that is available to us in the visuals area. This allows us to choose a very specific field from our data, that we know our end users will want to manipulate to see that data in various different states. So now, let's dive in and get a better understanding on these two filtering options, as they will most definitely be elements we will see in our finished reports.

## Slicer Visualization

Slicers are one of the most powerful types of visualizations, particularly as part of a busy report. A slicer is an on-canvas visual filter in Power BI Desktop that lets anyone looking at a report segment the data by a particular value, such as by year or by geographical location.

Slicer Visualization in Power BI is used to filter the results of visuals on your report page. It is used as a user level filter. Users will interact with report using Slicers. And with slicers, you can easily adjust the filter that's applied by interacting with the slicer itself. You can also specify options for how your slicer appears, and how you interact with it.

### When to use a Slicer

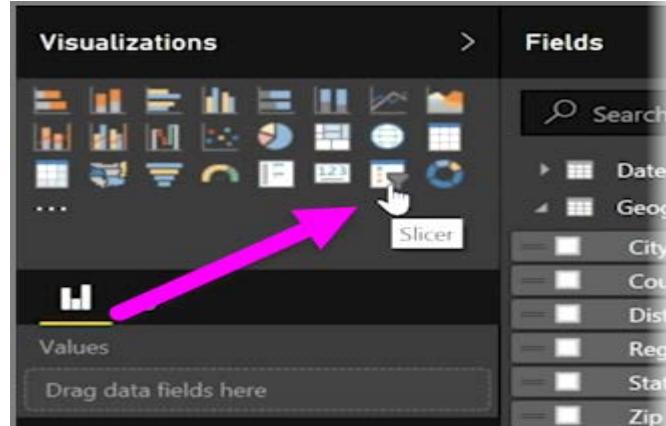
There could be scenarios where we have to filter data dynamically based on some values in such a way that when a particular value is selected, then all the data displayed in the report should get refreshed and aligned with respect to that value. For example, assume a report displaying sales across the globe of a Multinational Company. Imagine the sales viewer wants to view Sales of only United States. In this scenario, we can add filters under the Country. Therefore, when reader clicks particular Country, sales belonging to that country alone are shown. This can be achieved using Slicer.

A slicer is a list of values from which one, multiple or all values can be selected to filter the data shown in report.

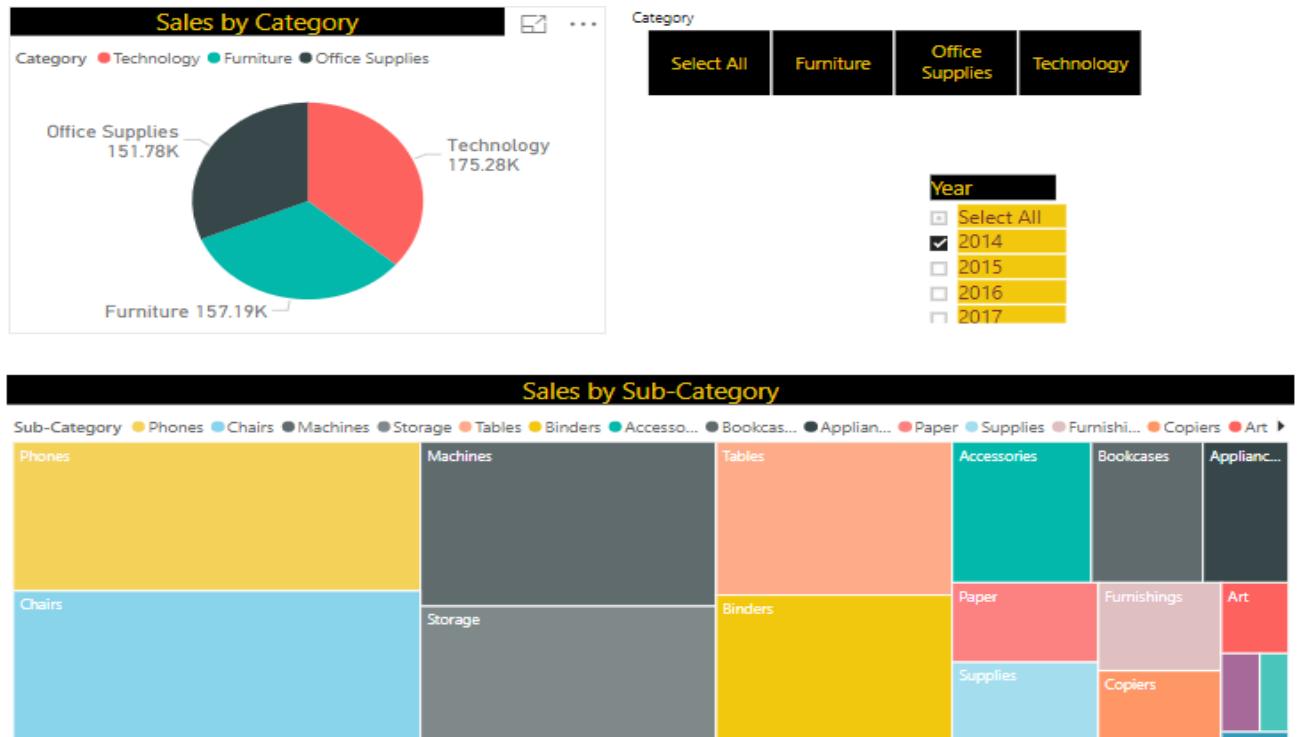
When you select an item in the slicer, only that item corresponding information will be shown and the rest of the data gets filtered out. For instance, take your entire sales data. You can set up slicers to look at sales by Country, Category, Subcategory and Product.

### Create and format slicers

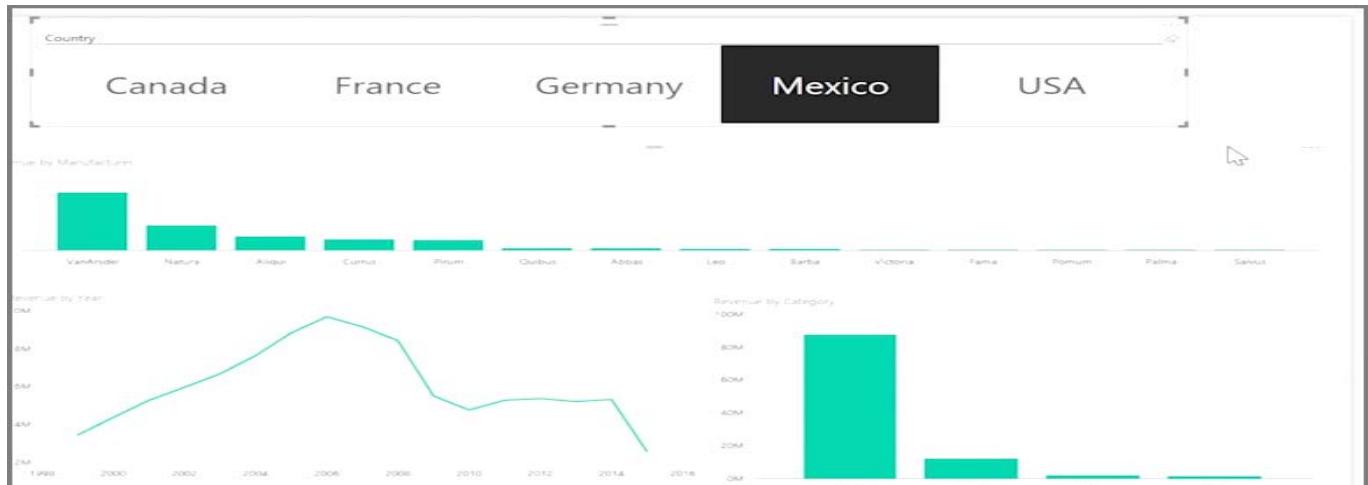
To add a slicer to your report, select Slicer from the Visualizations pane.



Drag the field by which you want to slice and drop it top of the slicer placeholder. The visualization turns into a list of elements with checkboxes. These elements are your filters, select the box next to one to segment and all other visualizations on the same report page are filtered, or sliced, by your selection.



There are a few different options available to format your slicer. You can set it to accept multiple inputs at once, or toggle Single Select mode to use one at a time. You can also add a Select All option to your slicer elements, which is helpful when you have a particularly long list. Change the orientation of your slicer from the vertical default to horizontal, and it becomes a selection bar rather than a checklist.



A slicer can be shown from one of various types below based on data type of the column used in slicer.

- List
- Dropdown
- Between
- Before
- After
- Relative
- Less than or equal to
- Greater than or equal to

### When to use a slicer

Slicers are a great choice when you want to

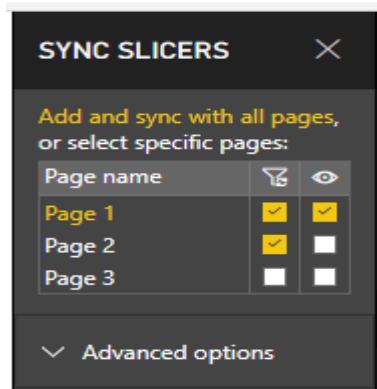
- Display commonly-used or important filters on the report canvas for easier access.
- Make it easier to see the current filtered state without having to open a drop-down list.
- Filter by columns that are unneeded and hidden in the data tables.
- Create more focused reports by putting slicers next to important visuals.

### Power BI slicers have the following limitations

- Slicers cannot be pinned to a dashboard.
- Drilldown is not supported for slicers.

### Synchronize slicers across report pages

In Power BI Desktop you can synchronize slicers across multiple report pages. To synchronize slicers, in the View Tab in the ribbon, select Sync slicers. When you synchronize slicers, the Sync Slicers pane appears, as shown in the following image.



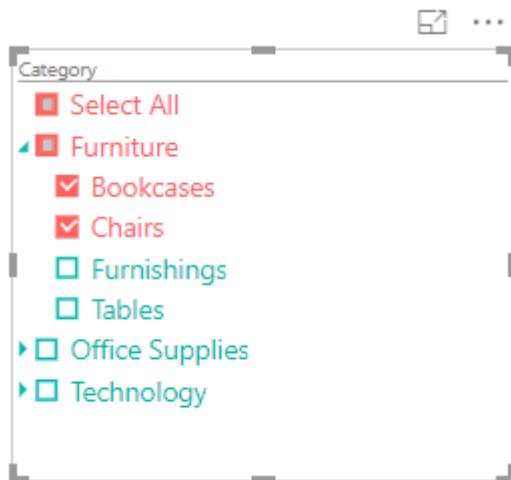
## Hierarchy Slicer - A Custom Visualization in Power BI

The Hierarchy Slicer visual is a great way to filter your data within a hierarchical view. The hierarchy slicer for Power BI provides the opportunity to simply select multiple members of different levels of a hierarchy as selection. The slicer can be used with an existing hierarchy or a manually created hierarchy.

### Advantages of Hierarchy Slicer

- Hierarchy Slicer lets you expand and dig into each level of your hierarchy data.
- Hierarchy Slicer allows you to display a measure value at the lowest level.
- Hierarchy Slicer replaces the idea using multiple regular slicers to represent what the Hierarchy Slicer does.

In this Hierarchy Slicer we're showing the ability to filter down a product hierarchy.



## Visualizing Categorical Data

The following visuals are best for displaying data values across categories. In these visuals, we will be displaying Bars, Columns, and other visual elements, which will be proportional to the data value. All of the visuals allow for interactive filtering and the use of drilldowns. We will focus on how to understand and configure the following visuals.

- Pie and Donut Charts
- Treemap Visual
- Bar and Column Charts
- Scatter Charts

### Pie and Donut charts

Both the Pie chart and Donut chart are meant to visualize a **particular section to the whole**, rather than comparing individual values to each other. The only difference between the two is that the Donut chart has a hole in the middle or the Center is blank and allows space for a label or icon. Both of these visuals can be very effective in allowing interactive filtering, but if there are too many categories it can become difficult to read and interpret. Pie Charts are very useful to visualize the High level data. If we have many categories where we need to see a particular section to the whole, then go with Treemaps.

#### Both charts have four field wells

**Legend** → You may use one or more categorical columns.

**Details** → You may use a categorical column.

**Values** → You may use one or more numerical fields here. If you use Details, you are limited to one field only.

**Tooltips** → You may use one or more fields here.

### Treemap

Treemaps display hierarchical data as a set of nested rectangles. Each level of the hierarchy is represented by a colored rectangle (often called a "branch") containing other rectangles ("leaves"). The space inside each rectangle is allocated based on the quantitative value being measured, with the rectangles arranged in size from top left (largest) to bottom right (smallest).

Treemap charts can be thought of as rectangular pie charts because they also show the relationship of the parts to the whole. You can nest rectangles to further divide the whole. There are five field wells in Treemap.

**Group** → You may use one or more categorical columns.

**Details** → You may use a categorical column.

**Values** → You may use one or more numerical fields here. If you use Details, you are limited to one field only.

**Tooltips** → You may use one or more fields here.

### When to use a treemap

Treemaps are a great choice

- To display large amounts of hierarchical data.
- When a bar chart can't effectively handle the large number of values.
- To show the proportions between each part and the whole.
- To show the pattern of the distribution of the measure across each level of categories in the hierarchy.
- To show attributes using size and color coding.
- To spot patterns, outliers, most-important contributors, and exceptions.

### Power BI has six variations of bar charts

It is best to use bar charts when you are **comparing values across categories**. Power BI Bar Chart is useful for the data comparison. Both the Bar and Column charts are very similar in setup and how they visual data. The only difference here will be the orientation. The Bar chart uses rectangular bars horizontally where the length of the bar is proportional to the data, while the Column chart displays the bars vertically, but both are used to compare two or more values. Both of these visualizations have three different formats - Stacked, Clustered, and 100% Stacked.

- Stacked bar chart
- Stacked column chart
- Clustered bar chart
- Clustered column chart
- 100% Stacked bar chart
- 100% Stacked column chart

**Stacked** → Arranged one on top of another.

**Clustered** → Positioned close together or one beside another

### All six charts share the same five field wells

**Axis** → You can use one or more categorical column.

**Legend** → One categorical column can be used.

**Value** → You can use one or more numerical fields. If you use a legend or color saturation, you can only put one field into this field well.

**Color saturation** → one numerical field can be used.

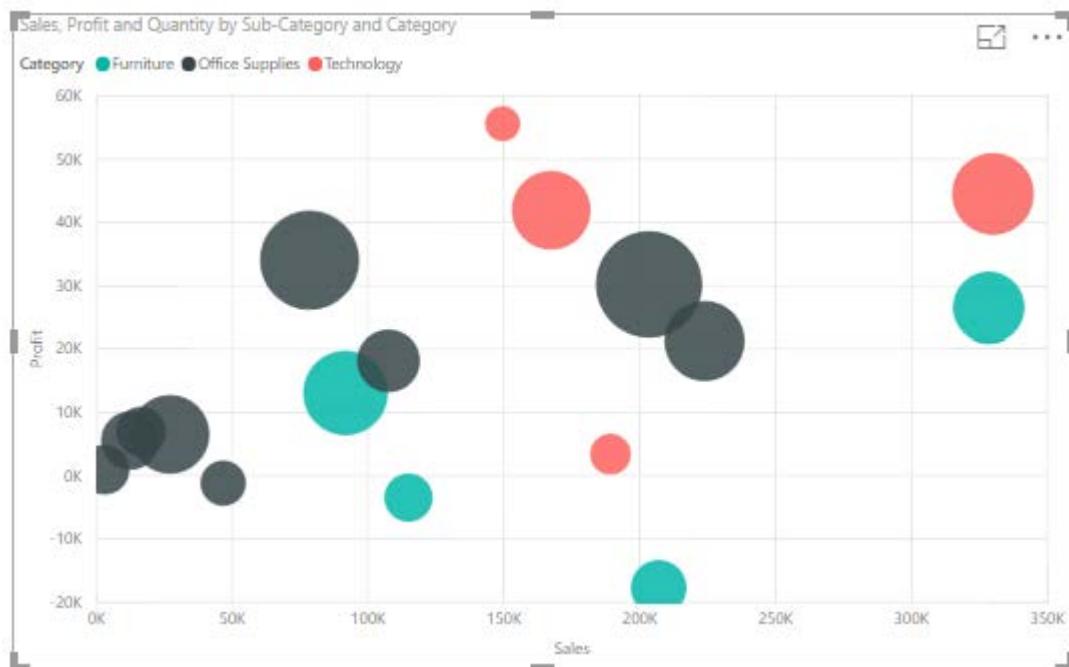
**Tooltips** → You can use one or more fields here.

There are some situations where the Bar chart will better display data, and the same thing can be said of the Column chart. The biggest limitation for the Column chart would be the limited space on the x-axis where the category would go. So, if you have a lot of data labels or if they are very long, you may find that the Bar chart is the better option. An example where you might choose the Column chart over the Bar chart is if your data set contains negative values. In a Bar chart, the negative values will show on the left side while in a Column chart they will display on the bottom. Users generally associate negative values with a downward direction.

## Scatter Chart / Bubble Chart

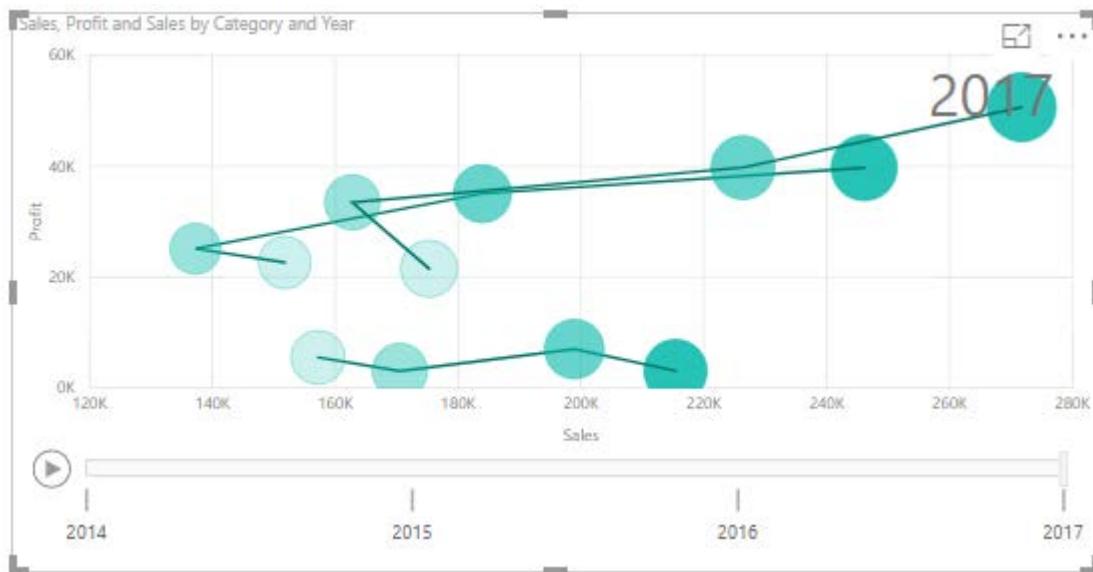
A scatter chart always has two value axes to show one set of numerical data along a horizontal axis and another set of numerical values along a vertical axis. The chart displays points at the intersection of an x and y numerical value, combining these values into single data points. These data points may be distributed evenly or unevenly across the horizontal axis, depending on the data.

A bubble chart replaces the data points with bubbles, with the bubble size representing an additional dimension of the data.



Column or Bar chart can be easily used for showing a single measure's insight across a category. Mixed charts such as Line and Column chart can be used for showing two measure and comparing their values across a set of categories. However, there are some charts that can be used to show values of three measures, such as Scatter Chart. Scatter chart not only shows values of three measure across different categories, it also has a special Play axis that helps you to tell the story behind the data.

Scatter chart is a built-in chart in Power BI that you can show up to three measure with a categorization in it. Three measures can be visualized in position of X axis, Y axis, and size of bubbles for scatter chart. You can also set up a date field in play axis, and then scatter chart will animate how measure values are compared to each other in each point of a time.



## Visualizing Trend Data

When we use the term Trend Data, we are talking about displaying and comparing values over time. Power BI gives us many options in this category, each with their own focus. The idea for each of the visuals, though, is to draw attention to the total value across a length of time.

- Line and Area Charts (Line chart, Area chart, Stacked area chart)
- Combo Charts (Line and Stacked column chart, Line and Clustered column chart)
- Ribbon Charts
- Waterfall and Funnel Charts

### Line and Area Charts

Power BI has a line chart and two area charts, which are similar to the line chart, but have a shaded area under the lines

- Line chart
- Area chart
- Stacked area chart

The Line chart is the most basic of our options when it comes to looking at data over time. Line charts are best used when you are comparing values across time. The Area chart and Stacked Area chart are based on the line chart; the difference is that the area between the axis and the line is filled in with colors to show volume. Because of this, we will focus on the Line chart for our example. Since we have a very nice Date hierarchy, we will use this alongside a couple of measures to see trending.

#### All three charts have the following field wells

Axis → One or more categorical columns.

Legend → You can place one categorical column.

Values → One or more numerical fields can be used; if you use a legend, you can only use one field in this well.

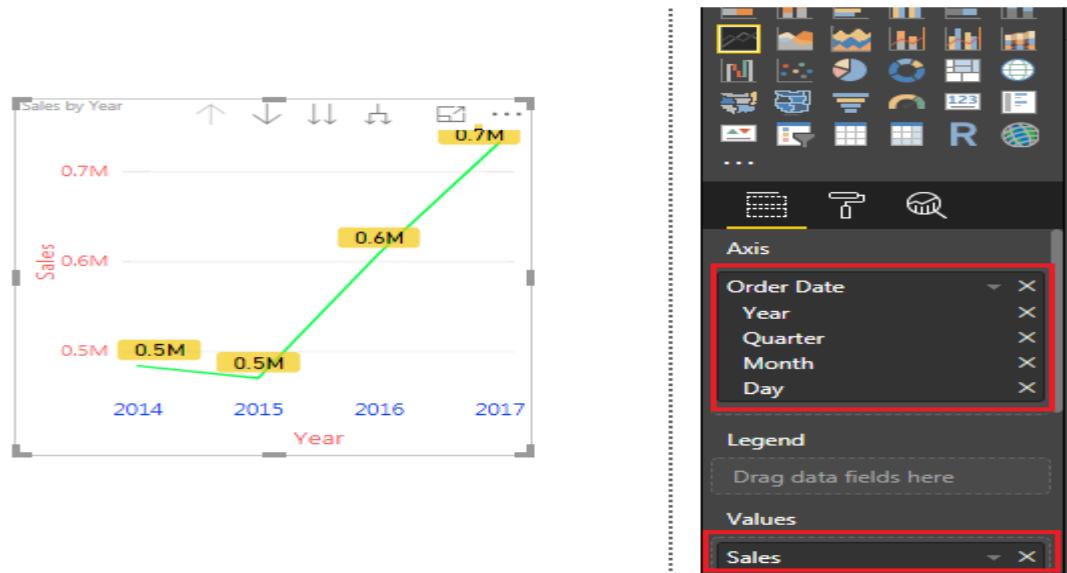
Tooltips → One or more fields can be used.

#### Single Line Chart

Put Date/Time value to the Axis field well, this is the X-axis of the line chart.

Put data field that you want to show (here Sales) in the Value field well.

The line chart is basically done. Now it is up to you to customize it as per your need.

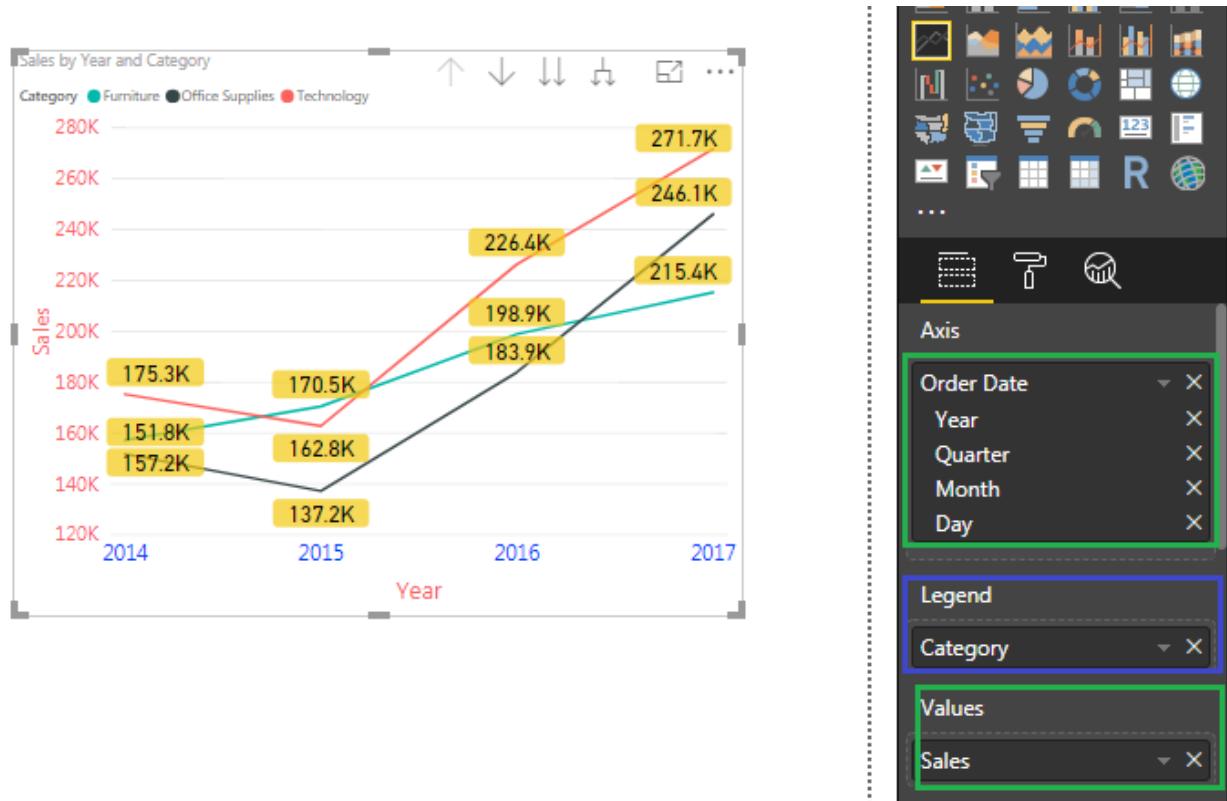


## Multiple Line Chart

Put Date/Time value to the Axis field well, this is the X-axis of the line chart.

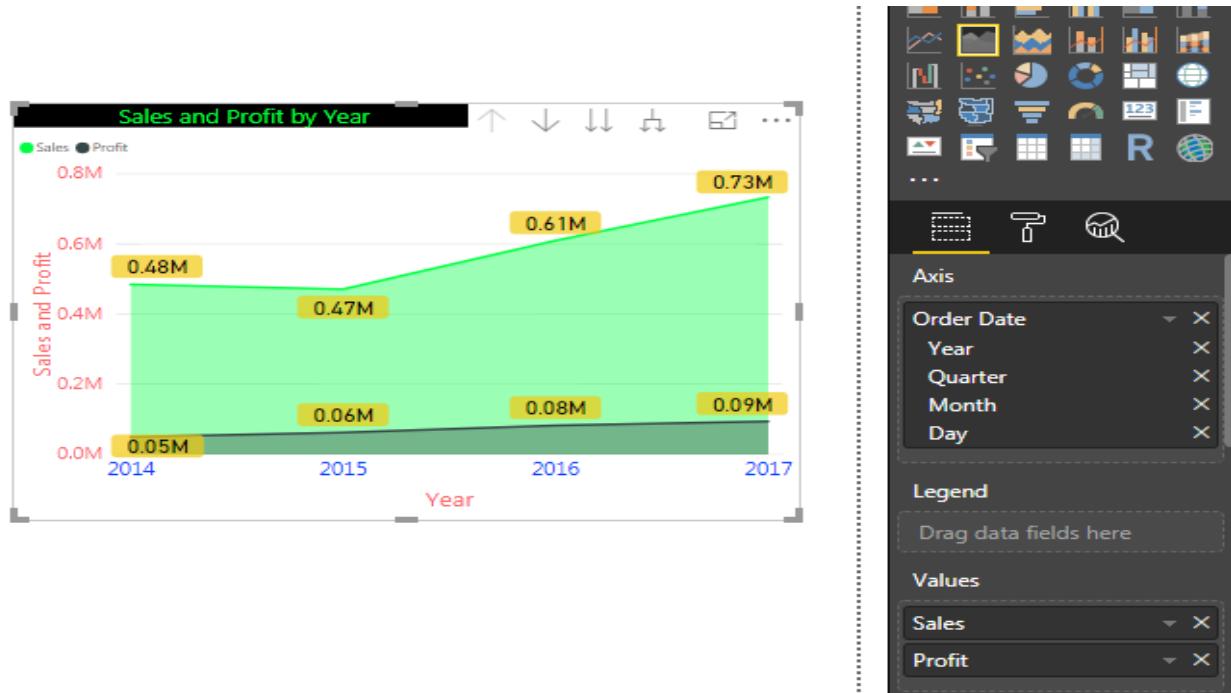
Put data field that you want to show (here Sales) in the Value field well.

To further breakdown the sales by Category, put the required field in the Legend field (here Category). Now it is up to you to customize it as per your need.



## Area Chart

Area charts are very similar to line charts. The basic area chart is based on the line chart. The area between axis and line is filled with colors to indicate volume. They both show quantitative data with a time period on the x-axis.



## Stacked Area Chart

A stacked area chart is used to compare each category of the data against the whole.



## Combo charts

There are two combo charts in Power BI

- Line and stacked column chart
- Line and clustered column chart

As the name states, Combo charts combine the Line chart and Column chart together in one visual. Users can choose to have either the Stack Column format or the Clustered Column format. By combining these two visuals together, we can make a very quick comparison of the data. The main benefit of this type of chart is that we can have one or two Y axes. What this means is that we 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 Order Quantity and Profit; let's use the two for our example.

### Let's look at, setting up the visual

Both charts have five field wells

**Shared axis** → One or more categorical columns.

**Column series** → A legend well in which you can use a categorical column.

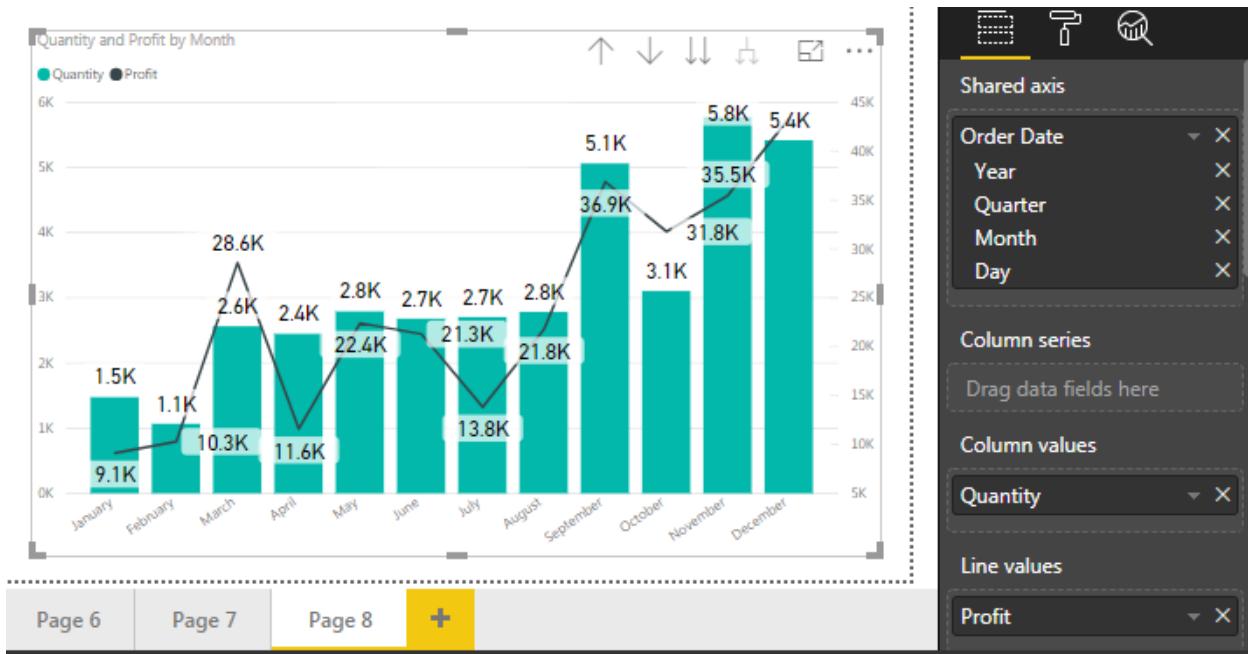
**Column values** → You may use one or more numerical fields in this well; if you have a column series, you can only use one field.

**Line values** → One or more numerical fields may be used in this well.

**Tooltips** → One or more fields may be used in this well.

For this example, we will be using the Line and Stacked Column Chart visual.

- ✓ For the Shared Axis area, let's select the Date field (Order Date).
- ✓ We will then select the Quantity field to populate the Column Values section.
- ✓ Then select the Profit measure field to populate the Line Values section.
- ✓ In this example, you can see that we have two Y axes; the left one relates to the Quantity while the right one corresponds with our Profit measure.
- ✓ By default, the column values appear on the left, and the line values appear on the right.
- ✓ Go ahead and expand the hierarchy two level; this will give us more data points to see the trending between the two measures, as seen in the below Figure. From this visual, it's fairly easy to validate that when we sell more items we make more profit.



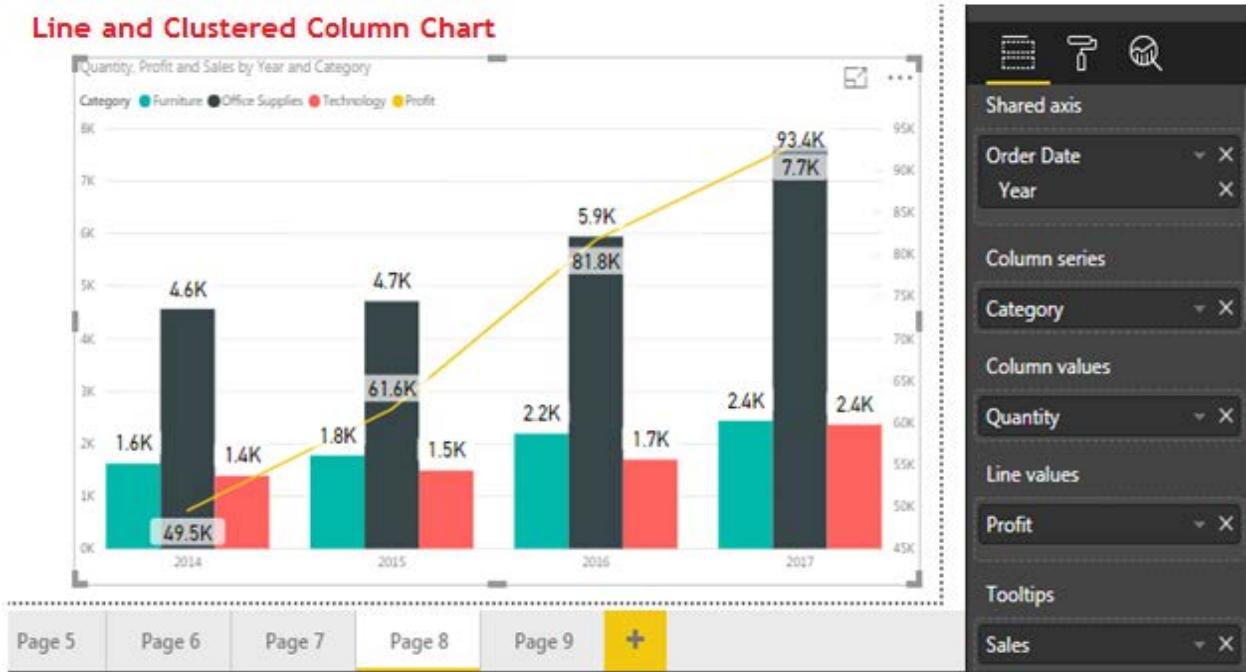
## When to use a Combo Chart

Combo charts are a great choice

- When you have a line chart and a column chart with the same X axis.
- To compare multiple measures with different value ranges.
- To illustrate the correlation between two measures in one visualization.
- To check whether one measure meet the target which is defined by another measure
- To conserve canvas space.

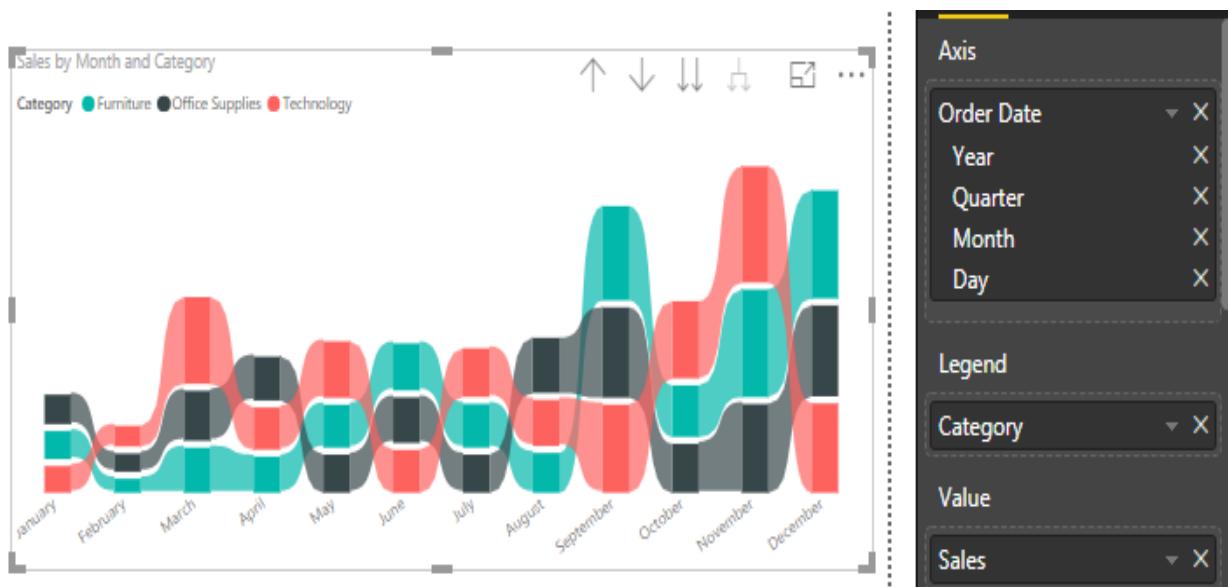


## Line and Clustered Column Chart



## Ribbon chart

The Ribbon chart is similar to a column chart, but it has ribbons between the bars to highlight changes in the relative ranking of categorical items. The item with the highest value will be displayed on top. You can use ribbon charts in Power BI to visualize data, and quickly determine which category of data has the highest rank (largest value). Ribbon charts are effective at showing rank change, with the highest range (value) always displayed on top for each time period.



## The chart has four field wells

Axis → One or more categorical columns.

Legend → You may use one categorical column.

Value → You can use one or more numerical fields in this well; if you use a legend, you can use one field only.

Tooltips → You may use one or more fields.

## Waterfall Chart

A waterfall chart shows a running total as values are added or subtracted. It's useful for understanding how an initial value (for example, net income) is affected by a series of positive and negative changes.



The columns are color coded so you can quickly tell increases and decreases. The initial and the final value columns often start on the horizontal axis, while the intermediate values are floating columns. Because of this "look", waterfall charts are also called bridge charts.



**Example:** Drag Order Date to Category field Well, Sales field to Y Axis field well and Category field to Breakdown field well as shown in the above visual. You can see Year wise sales and contribution of each category for decay or growth of year wise sales.

## When to use a waterfall chart

Waterfall charts are a great choice

- When you have changes for the measure across time series or different categories
- To audit the major changes contributing to the total value
- To plot your company's annual profit by showing various sources of revenue and arrive at the total profit (or loss).
- To illustrate the beginning and the ending headcount for your company in a year
- To visualize how much money, you make and spend each month, and the running balance for your account.

## The Funnel Chart

Funnel chart, which looks similar to a bar chart with bars center-aligned. The Funnel Chart allows users to see the percentage difference between values. Normally, the highest value is at the top and the lowest is at the bottom, which gives the look of a funnel. Each stage of the funnel with tells the percentage difference between itself and the previous stage, as well as compared to the highest stage. With this type of design, it makes sense that the Funnel Chart is very effective when visualizing a linear process with at least three or four stages. Our data set does not have a process with multiple stages, but we can still create something that gives us value.

Category	Value
Site Users	1020
Enquiries	800
Demo Attended	200
Registered	150
Placed	100



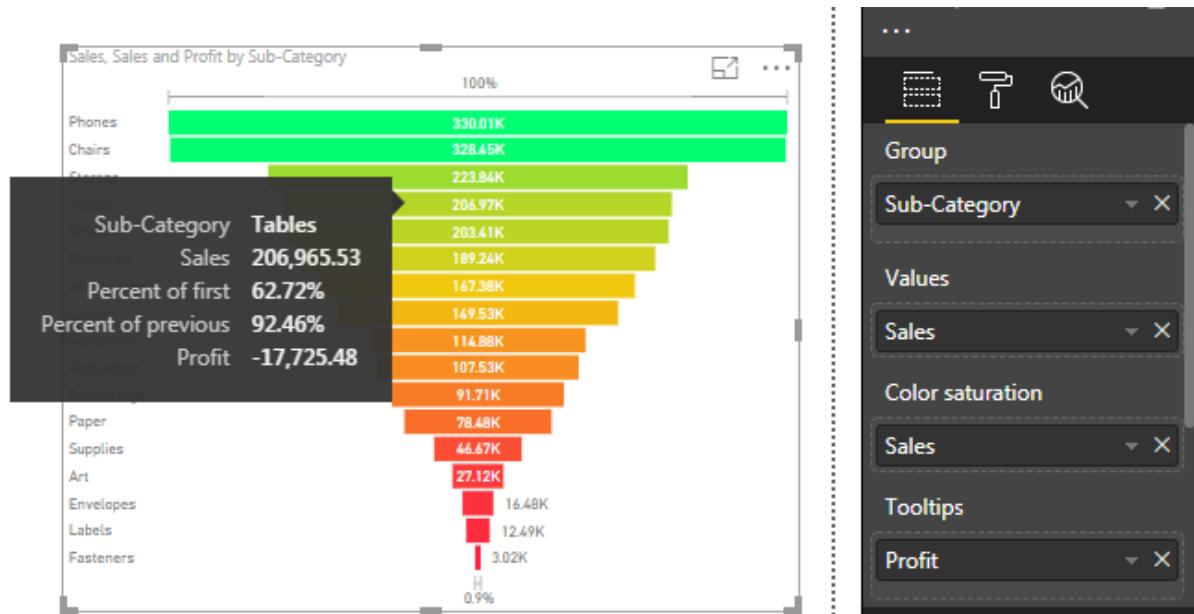
## The visual has four field wells

- **Group** → You can use one or more categorical columns here.
- **Values** → You can only use a numerical field here.
- **Color saturation** → A numerical field may be used.
- **Tooltips** → One or more fields can be used here.

A funnel chart can be a proper choice for showing values by stages, for instance, by lead generation stages. The visual can also be used for revealing bottlenecks in a process or to track workflow.

In our example drag the Sub-category field to Group and Sales to Values.

The way we have set up this visual allows us to very easily identify which Sub-category make the most Sales and which make the least, but this is something we can achieve with many other visuals. What gives the Funnel Chart an edge is when we hover over one of the sections within the funnel and note the items that appear within the Tooltip. You will see when we hover over the section for Tables Subcategory that the tooltip lets us know how it compares to the section directly above it, as well as how it compares to the highest section, which is represented by the Phones.



## Visualizing KPI Data

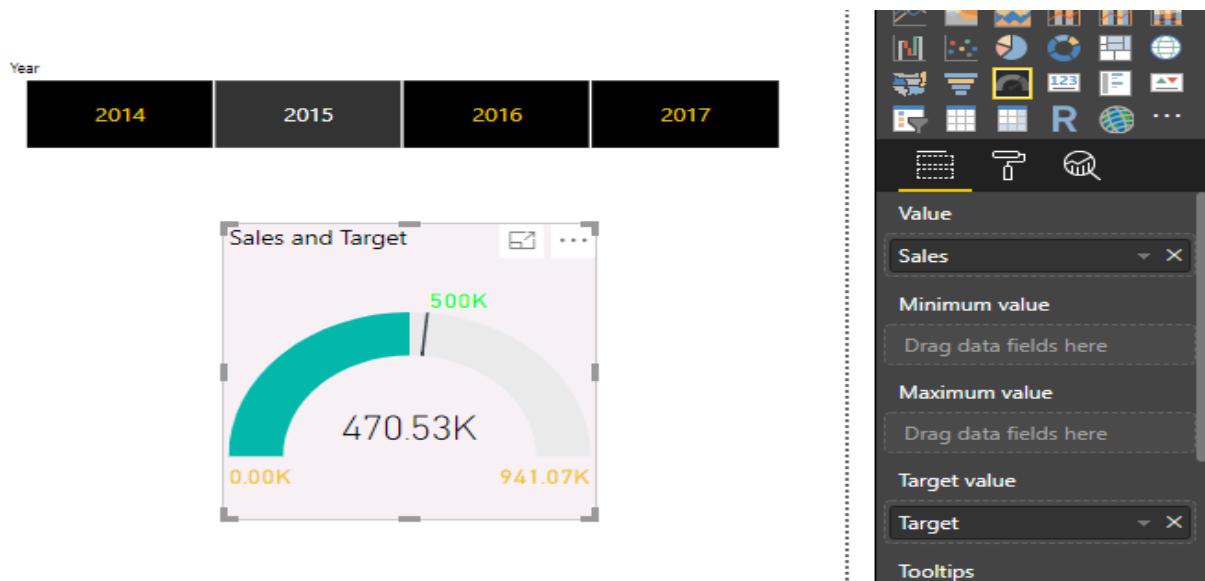
KPI stands for Key Performance Indicator. It is a measurable value that demonstrates how well a company is achieving a certain objective. With Power BI, we have a couple of options to measure the progress being made towards a goal for operational processes. The strength of a KPI visual lies in its simplicity. It displays a single value and its progress toward a specific goal.

Key Performance Indicator is a measure for business to understand how they perform in specific area that is important and many times critical for their business. For example, they would like to see how the current year to date sales is going against what they estimated? Is it higher or lower? What is the trend? Is it going upward or downward?

### The Gauge visual

The Gauge visual displays a single value within a circular arc and its progress towards a goal or target value that we specify. The Target Value is represented by a line within the arc. With our current data set, we do not have a measure that we can use to illustrate an accurate business goal, so we will have to create it. Before we set up this visual, we will need to create a new calculated measure.

Create a Measure with Name **Target** and assign a value of 500000



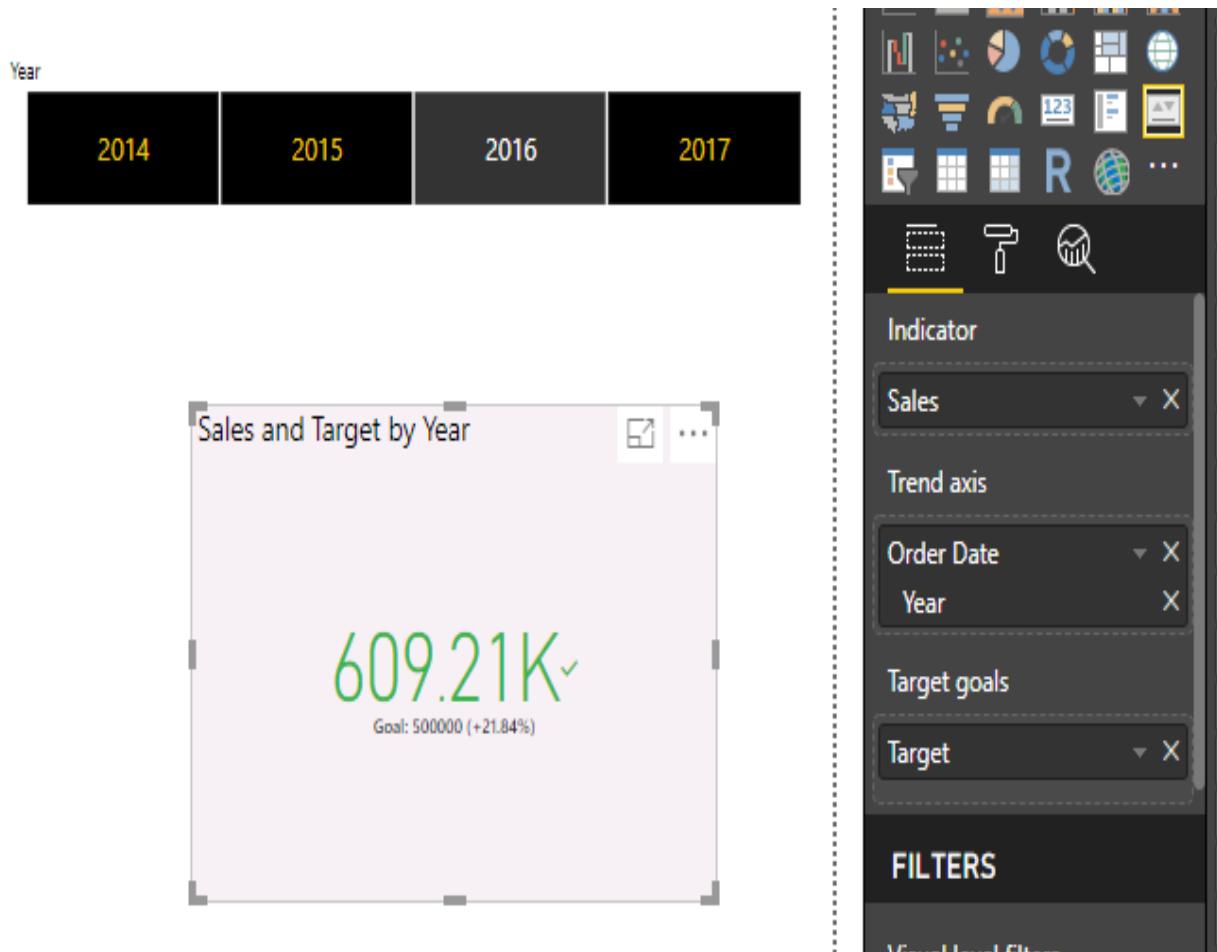
### When to use a radial gauge

Radial gauges are a great choice to

- Show progress toward a goal.
- Represent a percentile measure, like a KPI.
- Show the health of a single measure.
- Display information that can be quickly scanned and understood.

## The KPI visual

A Key Performance Indicator (KPI) is a visual cue that communicates the amount of progress made toward a measurable goal. Where the Gauge visual uses the circular arc to show the current progress, the KPI visual takes a more explicit approach and just shows the value in plain text, along with the goal. The only real visual elements that are in play with this visual occur when the indicator value is lower than the goal and the text is shown in red, and when it has surpassed the goal and the text is in green. This is definitely one of the more direct visuals and perfectly exemplifies what we want for a KPI.



## When to use a KPI

KPIs are a great choice

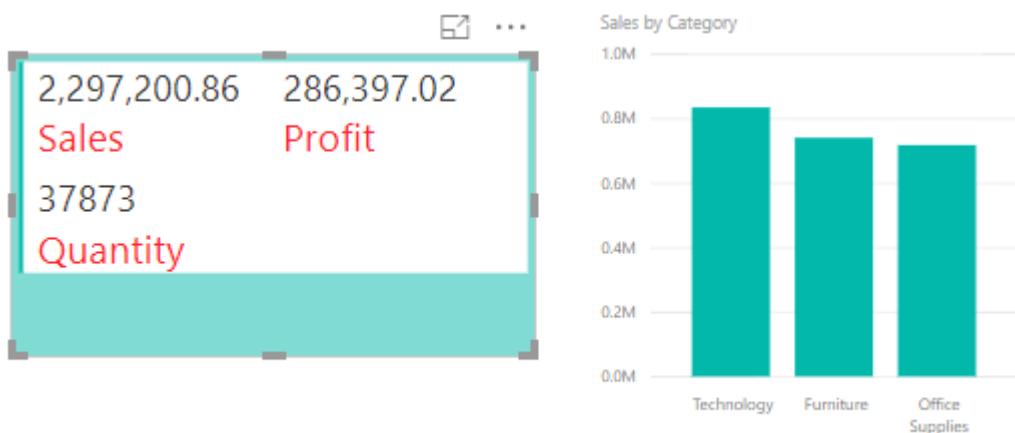
- To measure progress (what am I ahead or behind on?)
- To measure distance to a goal (how far ahead or behind am I?)

## Card Visualization

Sometimes a single number is the most important thing you want to track in your Power BI dashboard, such as total sales, market share year over year, or total opportunities. You can create a big number tile in a Power BI report as shown below using Card Visualization.



Sometimes instead of going and creating multiple Card Visualizations for multiple values that occupies more space in report page we will create a Multi Row Card and will place all the values in that Multi row card as shown below.



## Visualizing Geographical Data

One of the most exciting ways to visualize data in Power BI is through the various map options that we have. All the maps serve the same purpose to illustrate data in relation to locations around the world, but there are some small differences between each of them. Power BI integrates with Bing Maps to provide default map coordinates (a process called geocoding) so you can create maps. Together they use algorithms to identify the correct location, but sometimes it's a best guess. If Power BI tries can't create the map visualization on its own, it enlists the help of Bing Maps. All of the maps, except the Shape Map, have the option to provide the latitude and longitude coordinates, which will be the best way to ensure the appropriate location is being displayed. The reason for this is because the information that we provide the visual will be sent to Bing Maps to verify the positioning on the map. If we do not provide enough detail, then Bing may not return the desired results. For example, if we were to provide the map visual with a field that contains only the city name, which could result in some confusion because there may be multiple cities in the US with that name. In these scenarios, we will either want to supply some sort of geo-hierarchy to give better definition, or we can create new columns with more detailed information. Power BI also has a built-in feature when dealing with geographic data that allows users to help identify the type of data that is being provided, this is called Data Category.

Let's look at, setting up the example of Data Category

- Within the Fields Pane, expand the Geography table.
- The first field that we will categorize will be the City field. Highlight this field and then navigate to the Modeling ribbon. Once here, you will see the Data Category option.
- Inside of the drop-down menu, we will select the City option. Upon accomplishing this, you will see that there is now a globe icon next to the City field.
- Repeat the steps above for the StateName field, but choose the State or Province option for the data category dropdown.
- The final field that we need to perform these steps for is Country NameField; Select the Country/Region option from the dropdown.
- Now that we have given a better description of our geographical data, we can proceed with using the various map visuals. One thing of note is that using any of these visuals does require internet access because we are going to send data to Bing Maps.

## Maps in Power BI

Power BI has several options when it comes to visualizing geospatial data. In this section, we are going to review two map visuals, Map and Filled Map. You can use the following field wells.

**Location**→ One or more categorical columns can be used.

**Legend**→ You may use one categorical field here.

**Latitude**→ You may use one field here.

**Longitude**→ You may use one field here.

**Size**→ You may use one numerical field (Map only).

**Color saturation**→ You may use a numerical field here unless you use a Legend.

**Tooltips**→ You may use one or more fields here.

## What is sent to Bing Maps?

Power BI service and Power BI Desktop send Bing the geo data it needs to create the map visualization. This may include the data in the Location, Latitude, and Longitude buckets.

## The Map / Bubble Mapvisual

The first visual we will use to illustrate geographical data is simply called the Map visual. This visual is also referred to as the Bubble Map because it plots the points of data with circles that can be set to change in size based off a supplied measure. If you are using a Legend, your bubbles will be turned into pie charts. If your Legend field well is empty, you can also use a numerical field for color saturation. Better to pass latitude and longitude values to the visuals as we may find same city name some times. If we do not have such detailed data of Latitude and Longitude, so we will need supply the necessary information through the Location section, which will be sent to Bing Maps.

If you use geographical hierarchies in the Location field well, you will be able to drill down into specific areas on your map.

When using a geo-hierarchy with a map, enabling the Drill Mode, which is signified by the down arrow in the upper right, can make this visual even more enjoyable. Remember this for any visual where we have a hierarchy selected; you should explore the different views it gives you.

## The Filled Map visual

Unlike the traditional Map visual, that uses a bubble to indicate locations, the Filled Map visual uses shading in proportion across a geography or region to display the geographic data.

## Best Practices with Maps

### Use latitude and longitude fields (if they exist)

In Power BI, if the dataset you are using has fields for longitude and latitude → use them! Power BI has special buckets to help make the map data unambiguous. Just drag the field that contains your latitude data into the Visualizations > Latitude area. And do the same for your longitude data. When you do this, you also need to fill the Location field when creating your visualizations. Otherwise, the data is aggregated by default, so for example, the latitude and longitude would be paired at the state level, not the city level.

### Use geo-hierarchies so you can drill down to different "levels" of location

When your dataset already has different levels of location data, you and your colleagues can use Power BI to create *geo-hierarchies*. To do this, drag more than one field into the **Location** bucket. Used together in this way, the fields become a geo-hierarchy. In the example below we have added geo fields for: Country/Region, State, and City. In Power BI you and your colleagues can drill up and down using this geo-hierarchy.

When drilling with geo-hierarchies, it is important to know how each drill button works and what gets sent to Bing Maps.

- The drill button on the far right, called Drill Mode , allows you to select a map Location and drill down into that specific location one level at a time. For example, if you turn Drill Down on and click North America, you move down in the hierarchy to the next level -- states in North America. For geo-coding, Power BI sends Bing Maps country and state data for North America only.
- On the left there are 2 other drill options. The first option, , drills to the next level of the hierarchy for all locations at once. For example, if you are currently looking at countries and then use this option to move to the next level, states, Power BI displays state data for all countries. For geo-coding, Power BI sends Bing Maps state data (no country data) for all locations. This option is useful if each level of your hierarchy is unrelated to the level above it.
- The second option, , is similar to Drill Down, except that you don't need to click on the map. It expands down to the next level of the hierarchy remembering the current level's context. For example, if you are currently looking at countries and select this icon, you move down in the hierarchy to the next level → states. For geo-coding, Power BI sends data for each state and its corresponding country to help Bing Maps geocode more accurately. In most maps, you will use either this option or the Drill Down option on the far right, so you can send Bing as much information as possible to get accurate location information.

## Visualizing Tabular Data

We will see that there are many options within Power BI to visually represent data, but sometimes our users may want to see and compare detailed data and exact values. In these scenarios, by using the Table or Matrix visual ends up being our best choice. When leveraging either of these two visuals, it is important to take advantage of the Format area to ensure that users can easily interpret the detailed data that is being presented. One of the best ways to bring attention to values of importance with these visuals is by using Conditional Formatting. We will explore this option, as well as take advantage of the hierarchies to allow for drill downs within the visuals.

### The Table Visual

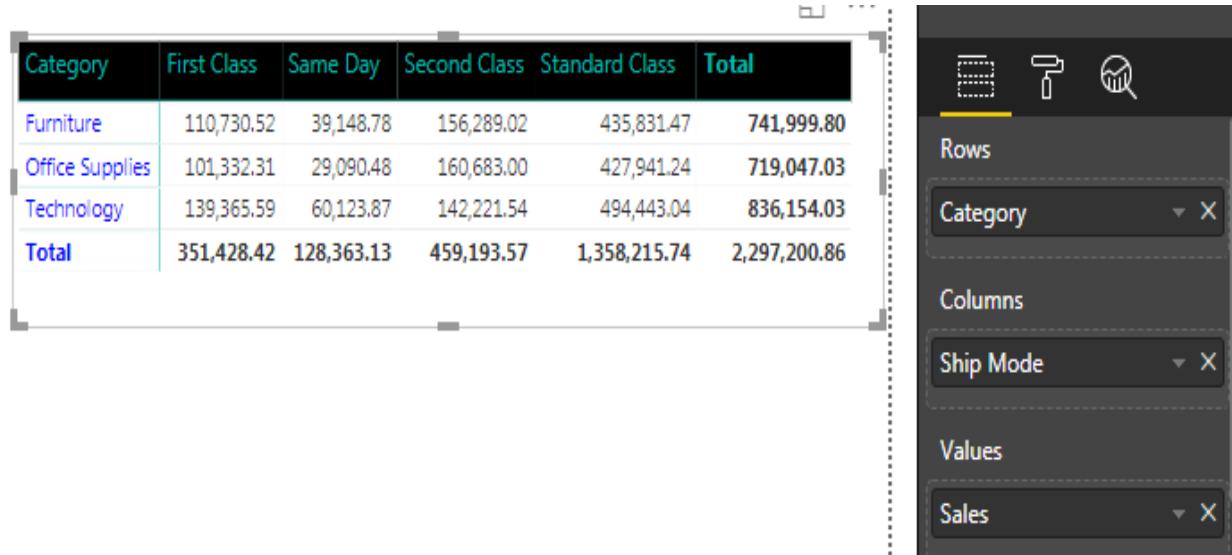
The table visual is perfect for looking at many values (measures) for a category. To really make the table shine, we will also want to take advantage of the Conditional Formatting option that is available to us.

In the below table visual we use multiple dimensions and measures also we formatted the Profit column conditionally in such a way negative profits in Red color and Positive Profits in Green Color.

Category	Sub-Category	Product Name	Quantity	Sales	Profit
Furniture	Bookcases	Atlantic Metals Mobile 2-Shelf Bookcases, Custom Colors	3	400.03	-113.26
Furniture	Bookcases	Atlantic Metals Mobile 3-Shelf Bookcases, Custom Colors	35	7,539.71	780.33
Furniture	Bookcases	Atlantic Metals Mobile 4-Shelf Bookcases, Custom Colors	27	5,184.08	-126.44
Furniture	Bookcases	Atlantic Metals Mobile 5-Shelf Bookcases, Custom Colors	26	5,492.89	15.05
Furniture	Bookcases	Bestar Classic Bookcase	31	1,897.81	-612.94
Furniture	Bookcases	Bush Andora Bookcase, Maple/Graphite Gray Finish	34	3,317.72	135.59
Furniture	Bookcases	Bush Birmingham Collection Bookcase, Dark Cherry	9	825.17	-117.88
Furniture	Bookcases	Bush Cubix Collection Bookcases, Fully Assembled	4	729.23	48.62
Total			37873	2,297,200.86	286,397.02

## Matrix Visualization

A Matrix is similar to a table, but it has different category headers on the columns and rows. As with tables, numerical information will be automatically totaled along the bottom and right side of the matrix.



## Create Power BI visuals using R Script

### Install R

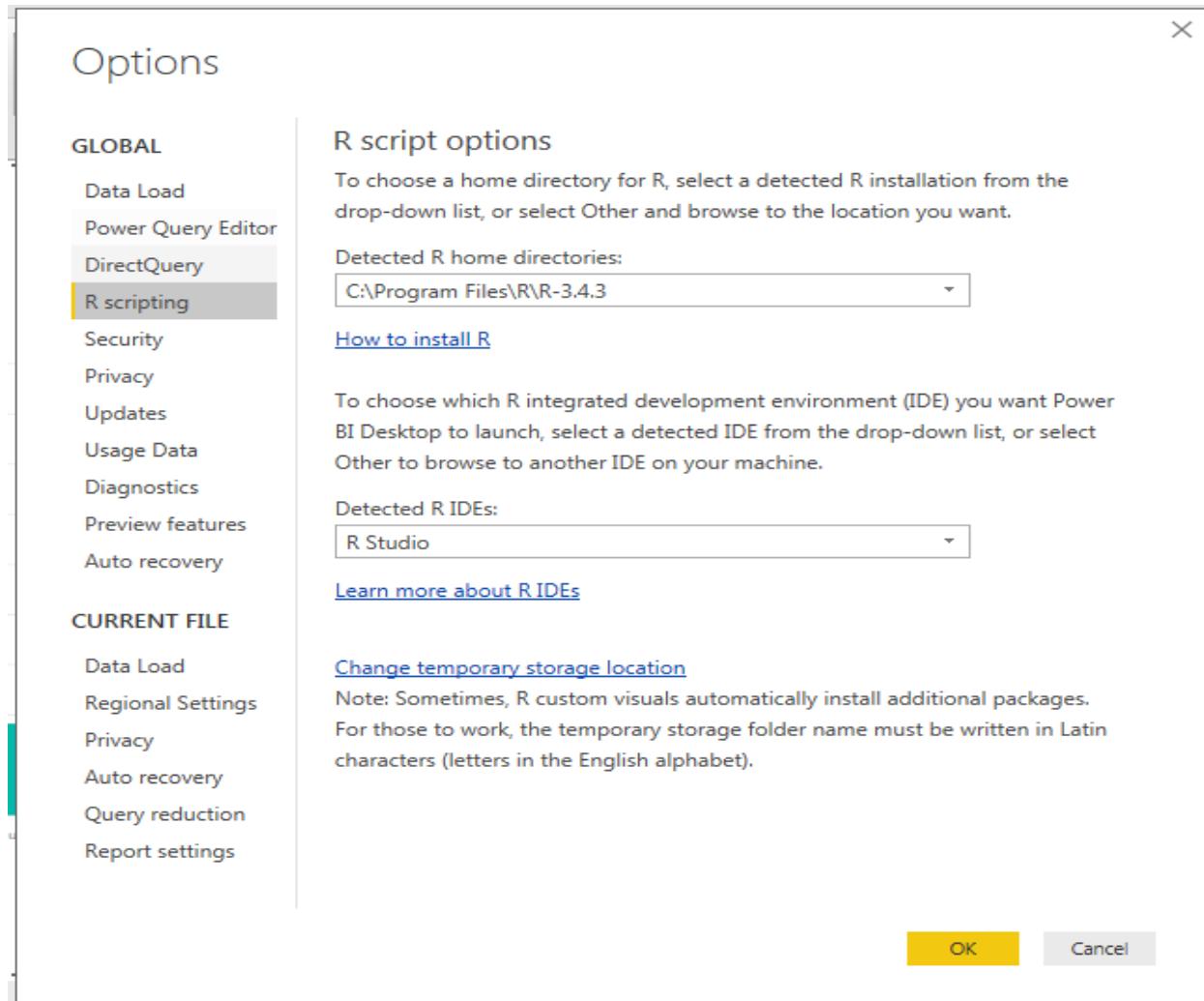
Power BI Desktop does not include, deploy, or install the R engine. To run R scripts in Power BI Desktop, you must separately install R on your local computer. You can download and install R for free from the below links.

<https://mran.revolutionanalytics.com/download>

<https://cran.r-project.org/bin/windows/base/>

### Enable R visuals

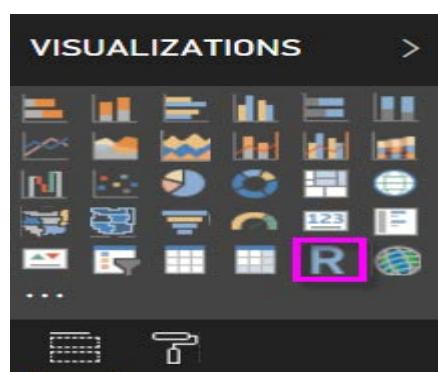
To enable R visuals, select File → Options and settings → Options and in the Options page that appears, make sure your local R installation is specified in the R Scripting section of the Options window, as shown in the following image. In the following image, the path local installation of R is C:\Program Files\R\R-3.4.3 and that path is explicitly provided in the text box. Make sure the path it displays properly reflects the local R installation you want Power BI Desktop to use.



Once you specify your R installation, you're ready to begin creating R visuals.

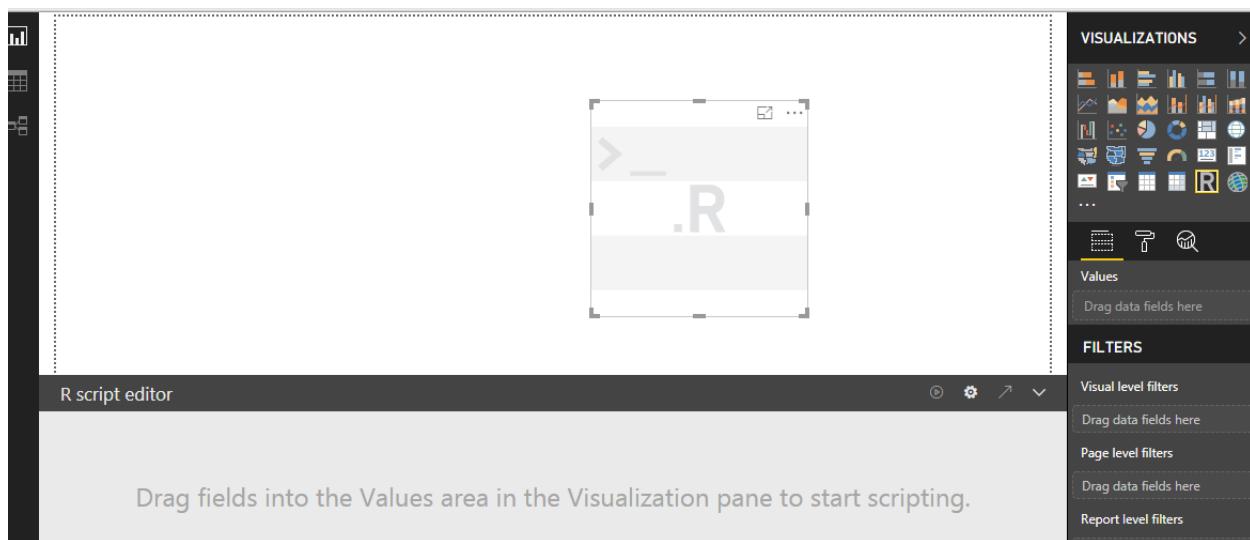
### Create R visuals in Power BI Desktop

Select the R Visual icon in the Visualization pane, as shown in the following image, to add an R visual.



When you add an R visual to a report, Power BI Desktop does the following

- A placeholder R visual image appears on the report canvas.
- The R script editor appears along the bottom of the center pane.



Next, add fields you want to consume in your R script to the Values section in the Fields well, just as you would with any other Power BI Desktop visual.

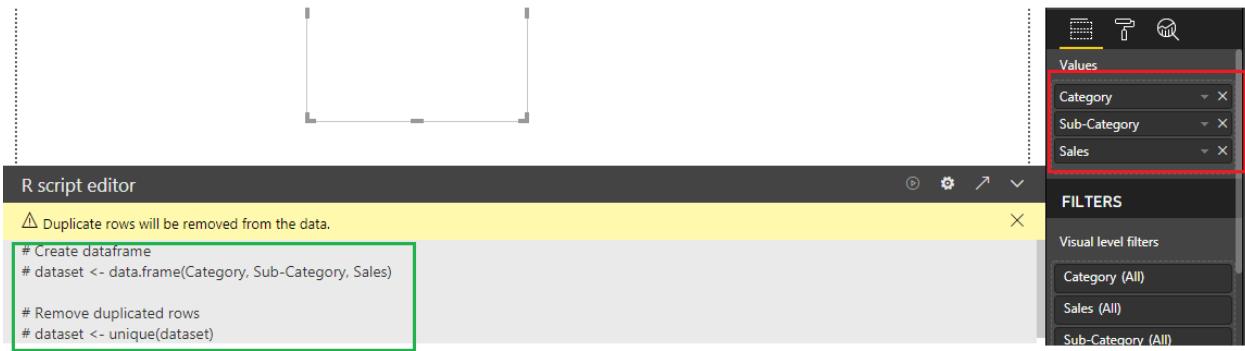
**Note:** The default aggregation type for R visuals is **do not summarize**.

Now you can use the data you selected to create a plot.

As you select fields, the R script editor generates supporting R script binding code based on your selections in the gray section along the top of the editor pane.

In the example shown in the following image, three fields were selected: Category, Sub-Category, and Sales. As a result of those selections, the R script editor generated the following binding code:

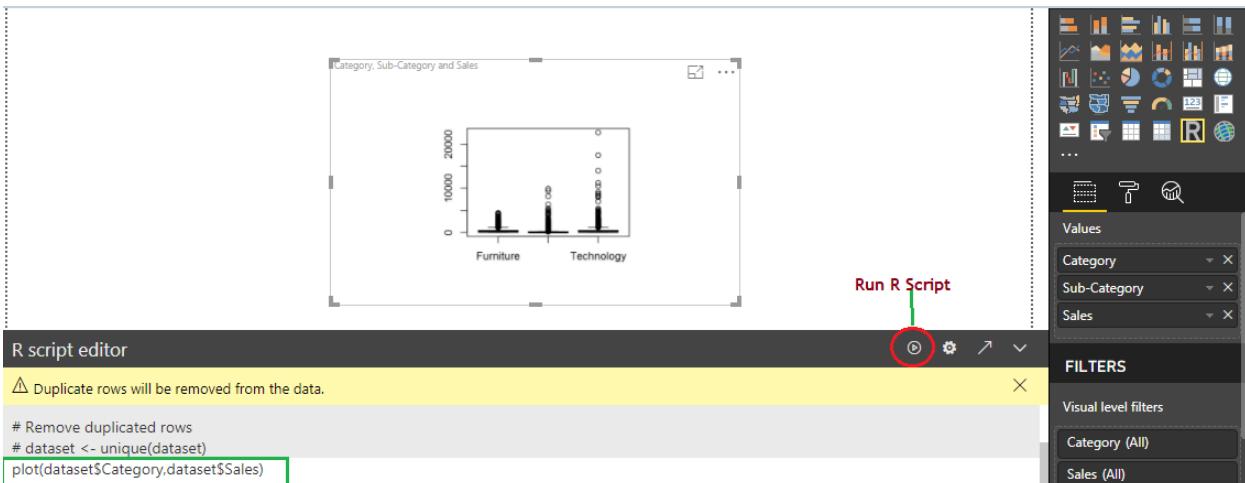
- A dataframe called dataset was created
- That dataframe is comprised of the different fields selected by the user
- The default aggregation is do not summarize
- Similar to table visuals, fields are grouped and duplicate rows only appear once



The generated dataframe is called a dataset, and you can access selected columns by their respective names. For example, access the Category field by writing `dataset$Category` in your R script. For fields with spaces or special characters, use single quotes (`dataset$'Sub-Category'`).

With the dataframe automatically generated by the fields you selected, you're ready to write an R script those results in plotting to the R default device. When the script is complete, select Run from the R script editor title bar (Run is on the right side of the title bar).

```
plot(dataset$State,dataset$Sales)
```



When you select Run, Power BI Desktop identifies the plot and presents it on the canvas. Since the process is executed on your local R installation, make sure the required packages are installed.

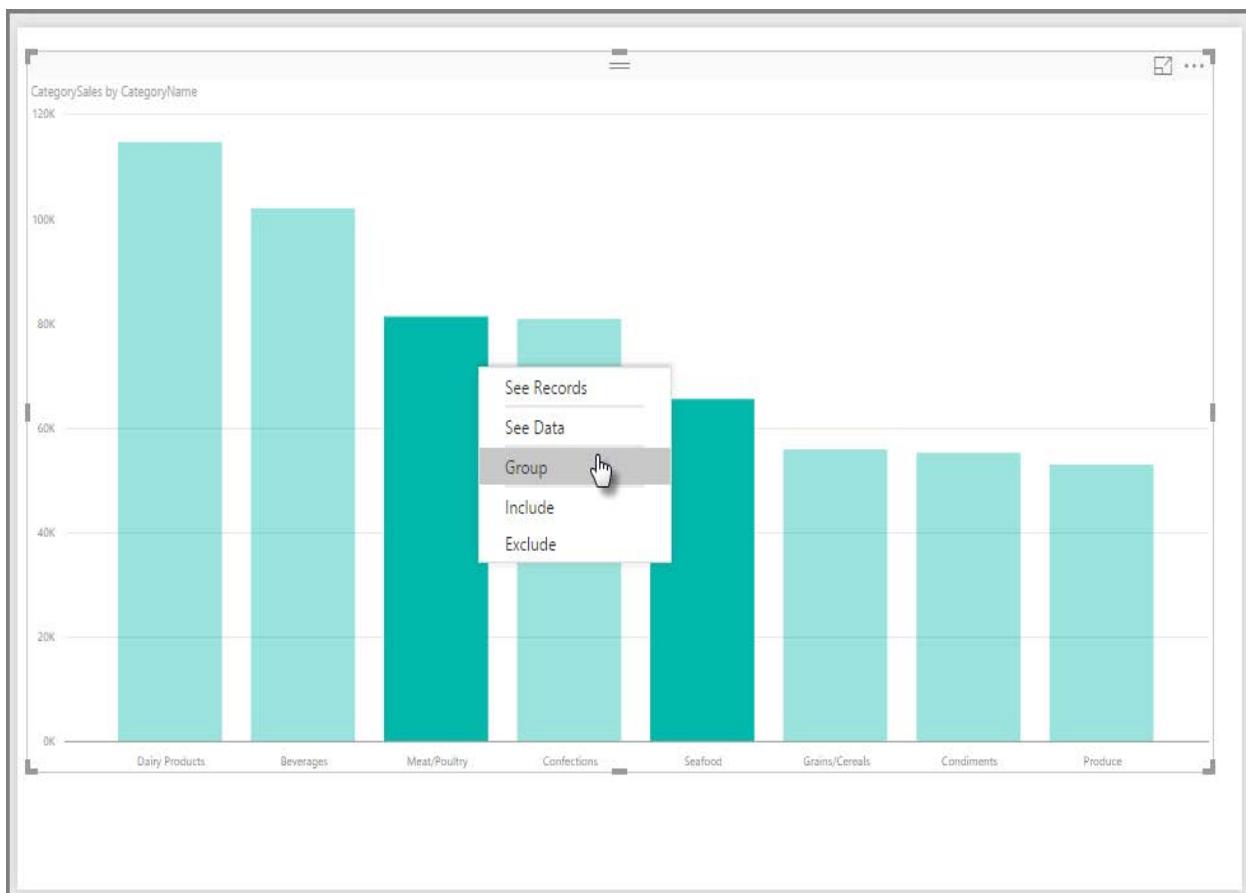
## Grouping and Binning

When Power BI Desktop creates visuals, it aggregates your data into chunks (or groups) based on values found in the underlying data. Often that's fine, but there may be times when you want to refine how those chunks are presented. For example, you might want to place three categories of products in one larger category (one group). Or, you might want to see sales figures put into bin-sizes.

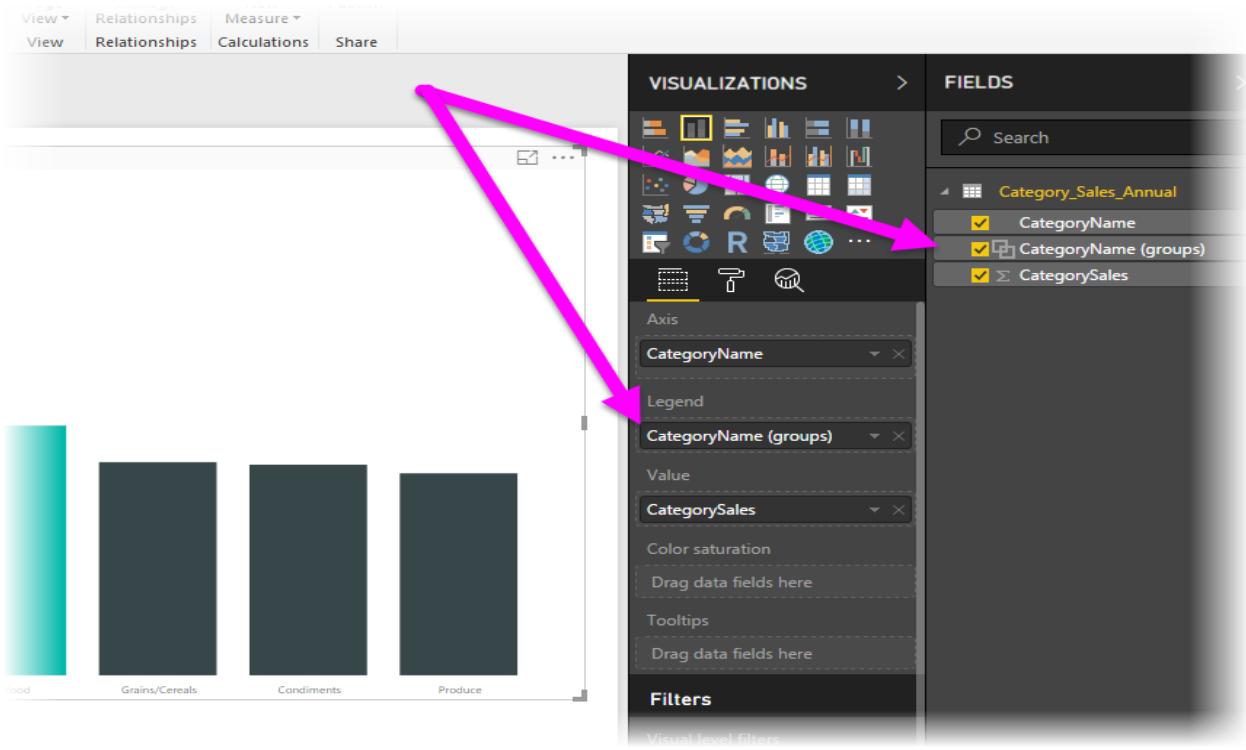
In Power BI Desktop, you can Group data points to help you more clearly view, analyze, and explore data and trends in your visuals. You can also define the Bin size, often called binning, to put values into equally sized groups that better enable you to visualize data in ways that are meaningful.

### Using grouping

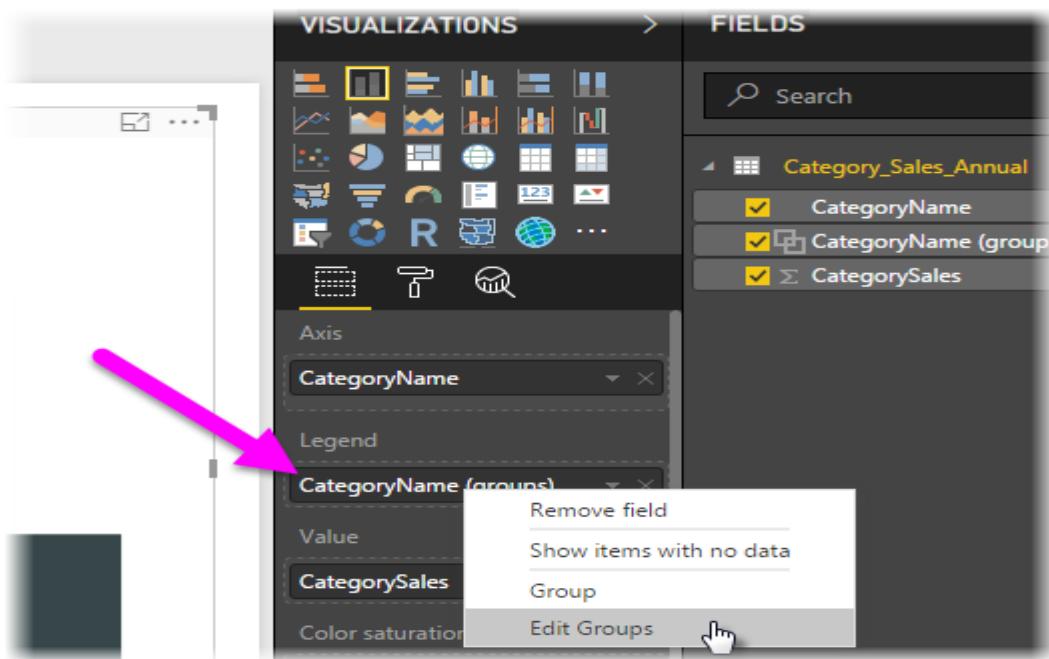
To use grouping, select two or more elements on a visual by using Ctrl+click to multi-select elements. Then, right-click one of the multi-select elements and select Group from the menu that appears.



Once created, the group is added to the Legend bucket for the visual and it also appears in the Fields list.



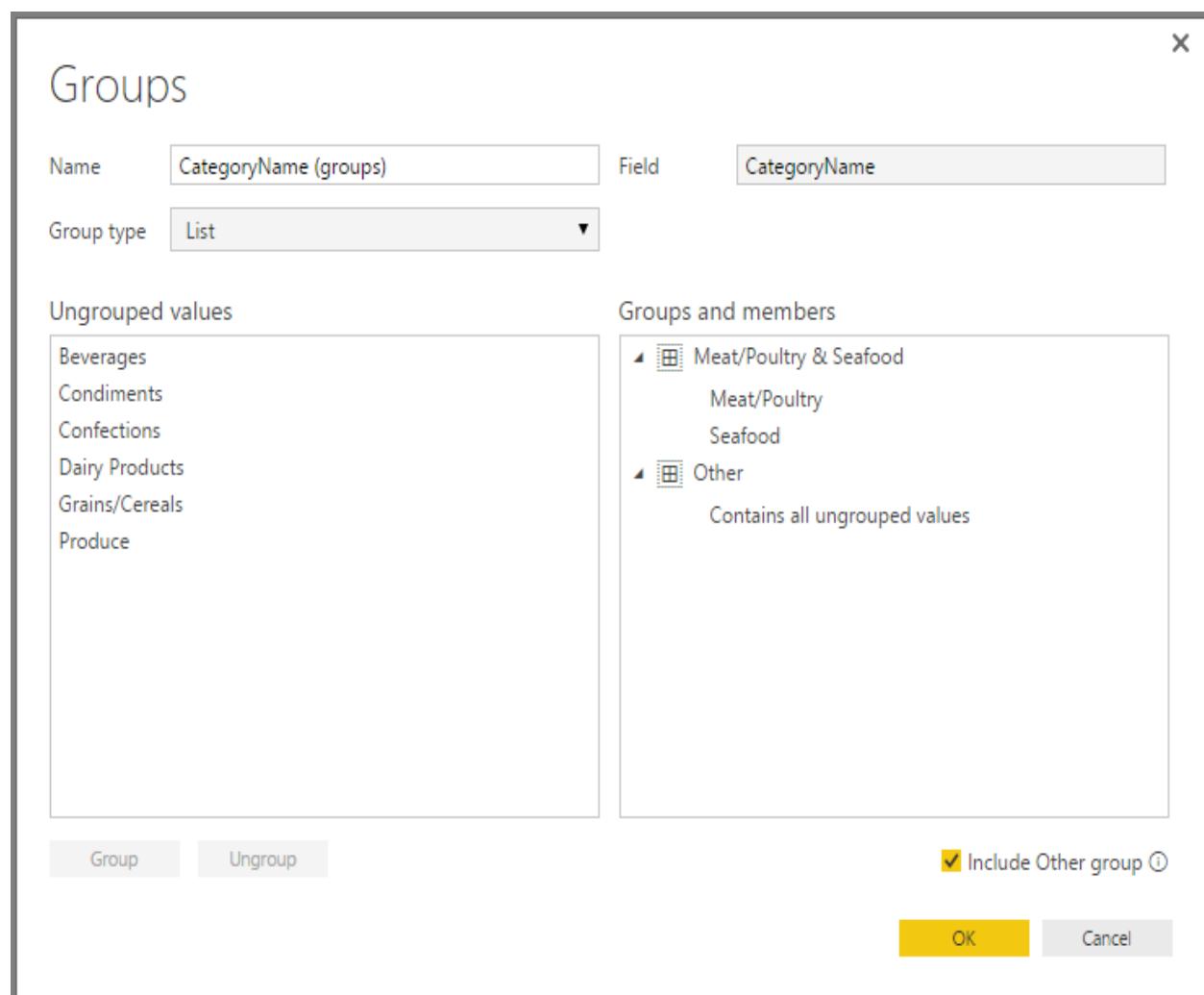
Once you have a group, you can easily edit the members of that group by right-clicking the field from the Legend bucket, or from the Fields list, and selecting Edit Groups.



In the Groups window that appears, you can create new groups or modify existing groups. You can also rename any group by double-clicking on the group title in the Groups and members box and typing a new name.

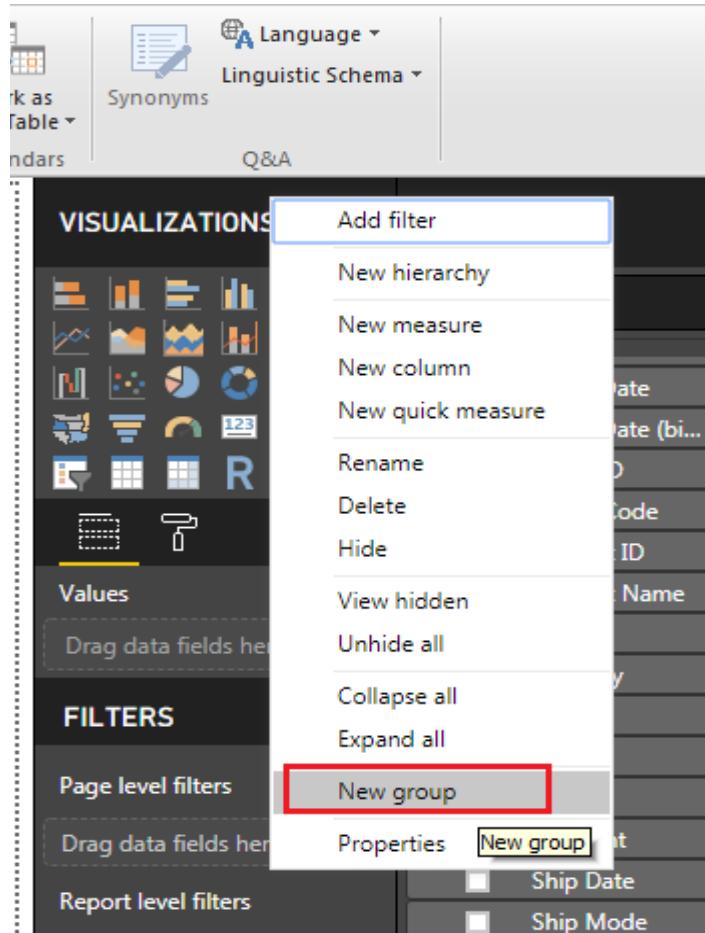
There are all sorts of things you can do with groups like you can add items from the ungrouped values list into a new group or into one of the existing groups. To create a new group, select two or more items (using Ctrl+click) from the ungrouped values box, and then click the Group button below that box.

You can add an ungrouped value into an existing group, just select the ungrouped value, then select the existing group to which you want to add it, and click the Group button. To remove an item from a group, select it from the Groups and members box and then click Ungroup. You can also select whether ungrouped categories should be placed into the Other group, or should remain ungrouped.

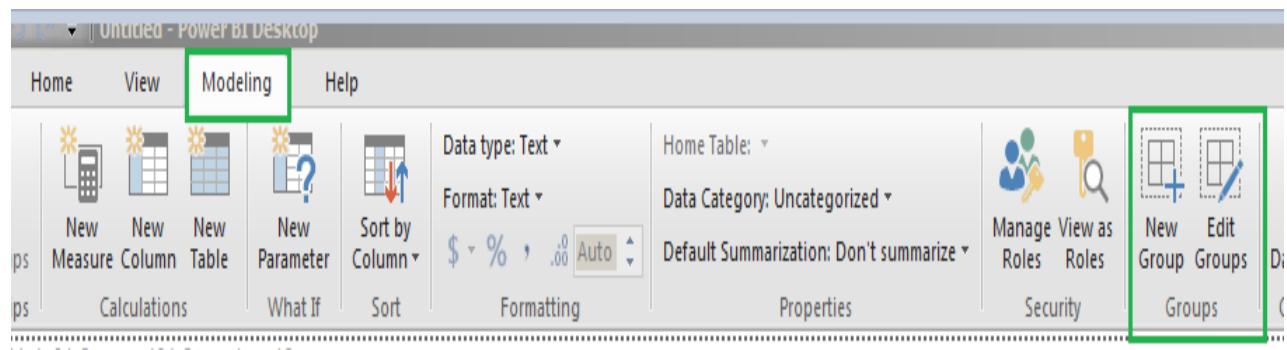


## Note

You can create groups for any field in the Fields well, without having to multi-select from an existing visual. Just right-click the field and select New Group from the menu that appears.



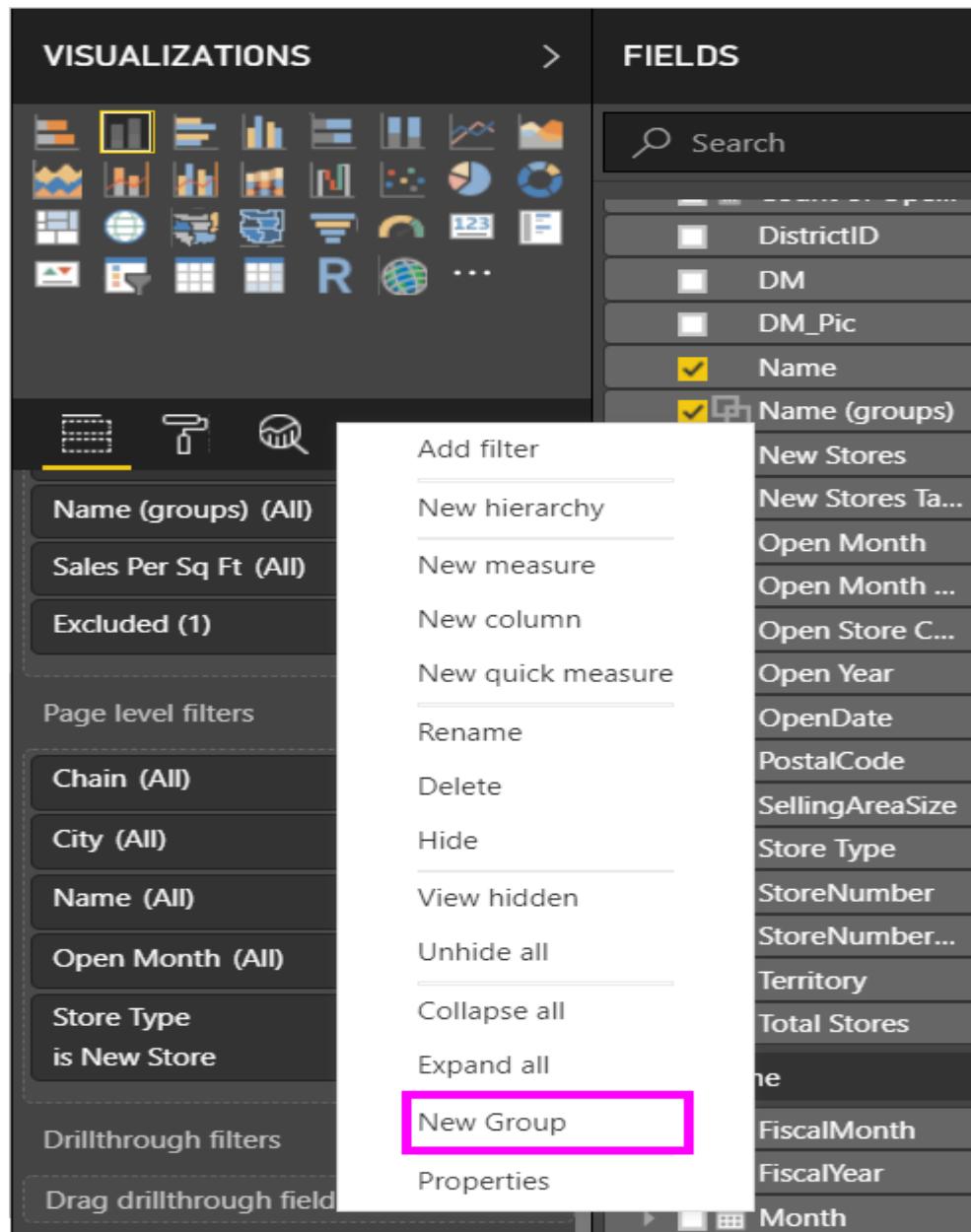
You can also create New Group and Edit Existing Groups from the Ribbon under Modeling Tab as shown below.



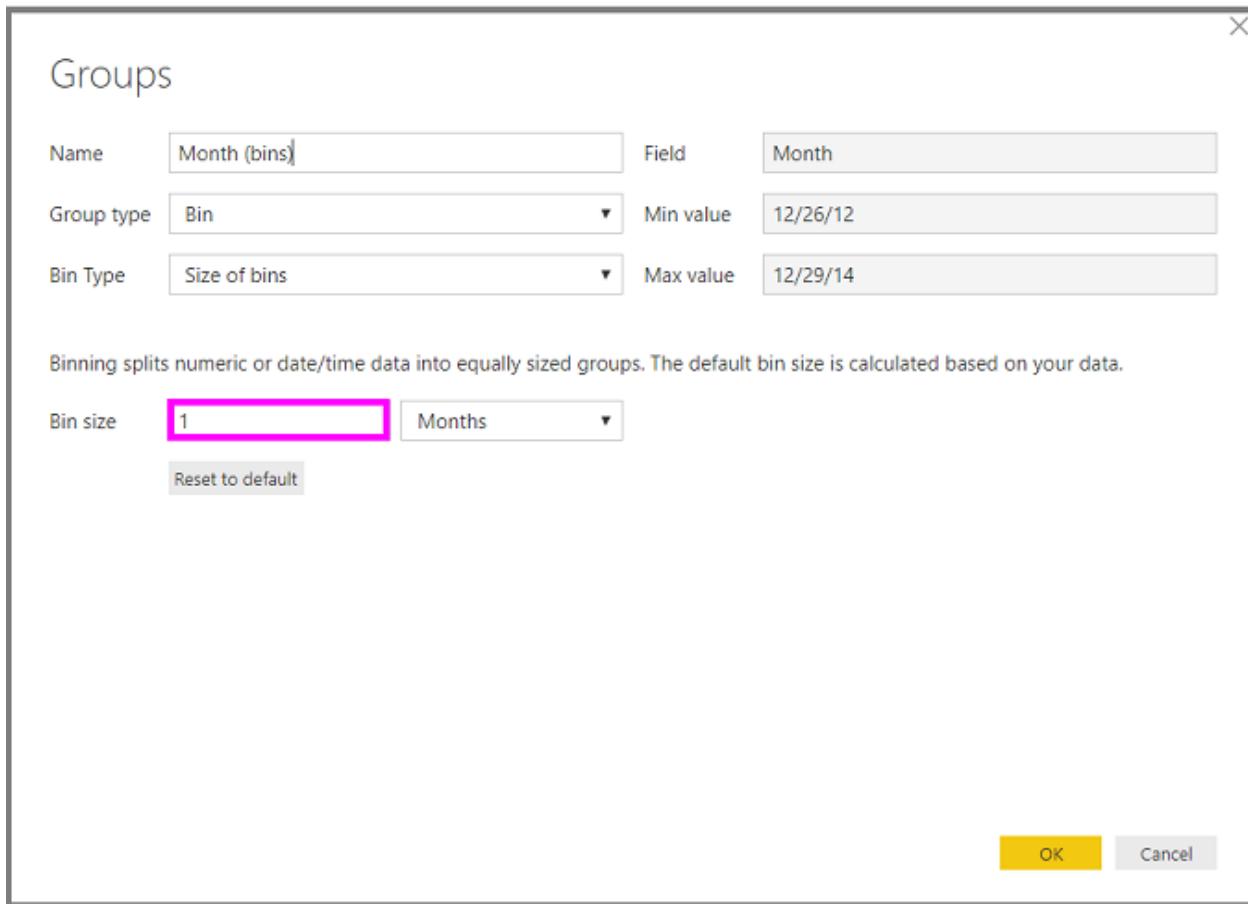
## Using binning

You can set the bin size for numerical and time fields in Power BI Desktop. You can use binning to right-size the data that Power BI Desktop displays.

To apply a bin size, right-click a Field and select New Group.



From the Groups window, set the Bin size to the size you want.



When you select OK, you'll notice that a new field appears in the Fields pane with (bins) appended. You can then drag that field onto the canvas to use the bin size in a visual.

The screenshot shows the Power BI Fields pane. On the left, there are sections for 'Page level filters' (Chain (All), City (All), Name (All), Open Month (All), Store Type is New Store) and 'Drillthrough filters' (Drag drillthrough fields here). On the right, the 'Fields' pane lists various fields under categories like 'OpenDate', 'PostalCode', 'SellingAreaSize', 'Store Type', 'StoreNumber', 'StoreNumber...', 'Territory', 'Total Stores', and 'Time'. Under 'Time', 'FiscalMonth' is listed. A tooltip 'Time'[Month (bins)] appears over the 'Month (bins)' field, which is highlighted with a pink box. Other fields in the 'Time' category include 'Year', 'Month', and 'Period'.

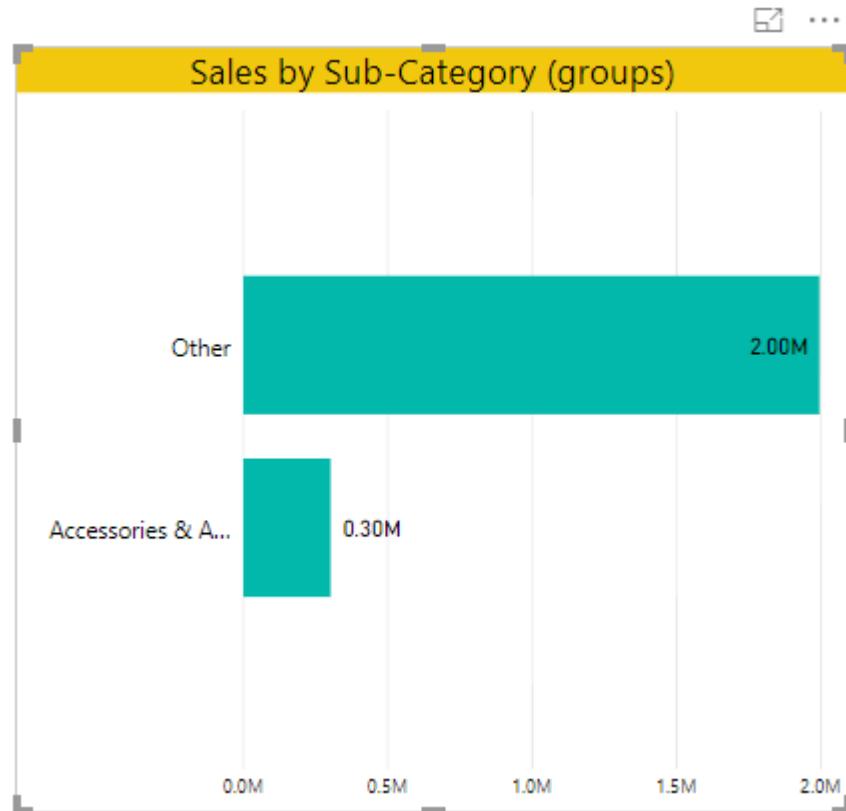
## Creating Bins on Number Column

We have two columns Order Date and Ship Date, from those columns get the number of days taken for delivering the order using the below DAX formula by creating a New Column. Give the column Name as Delivered Days. In his column we will create the Bins to see how it works.

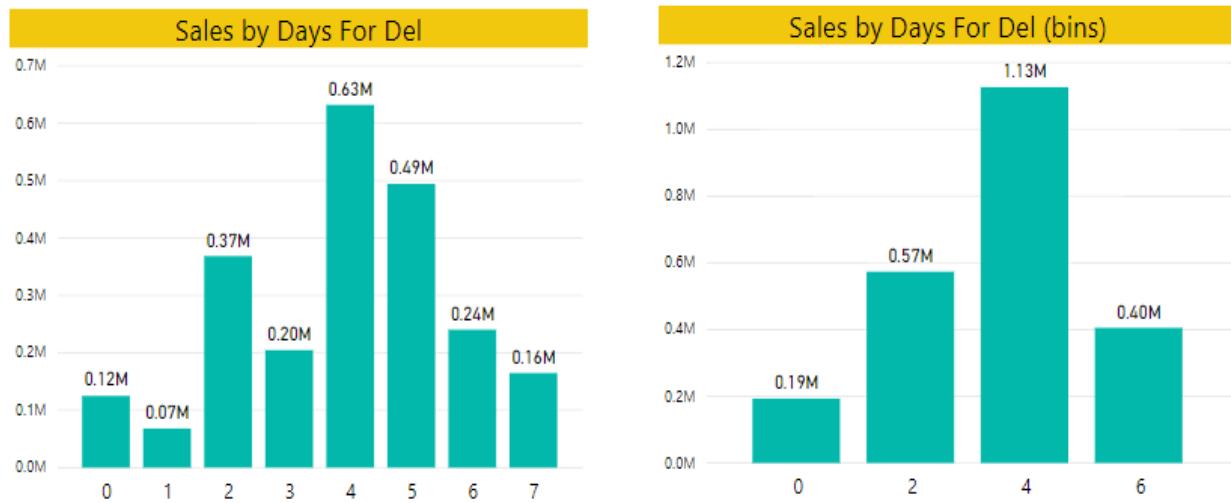
DATEDIFF(Orders[Order Date], Orders[Ship Date], DAY)

## Practice

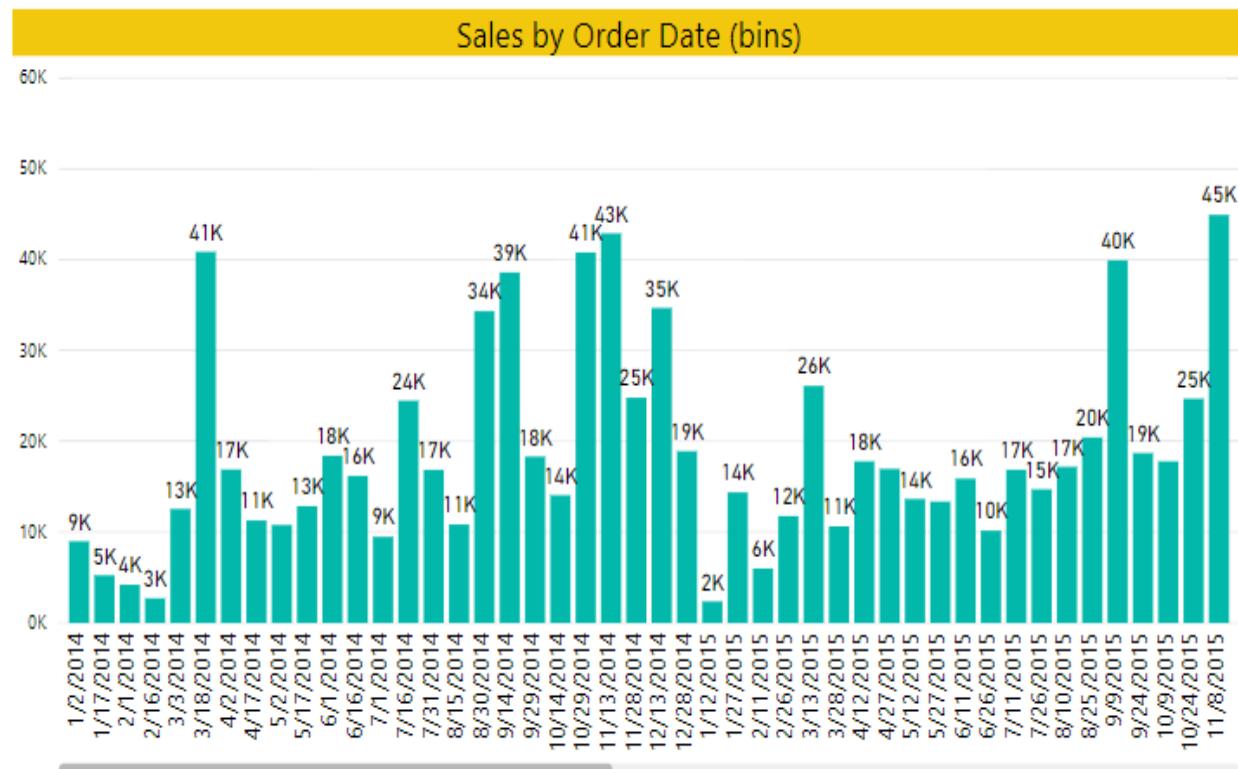
Create a visualization which show Sales by Sub-Category by Grouping Accessories, Appliances and Art into one group and all other Sub-Categories into another Group.



Create a Visualization which shows sales by "No of Days for Delivery" by creating bin on "No of Days for Delivery" field with bin size 2.



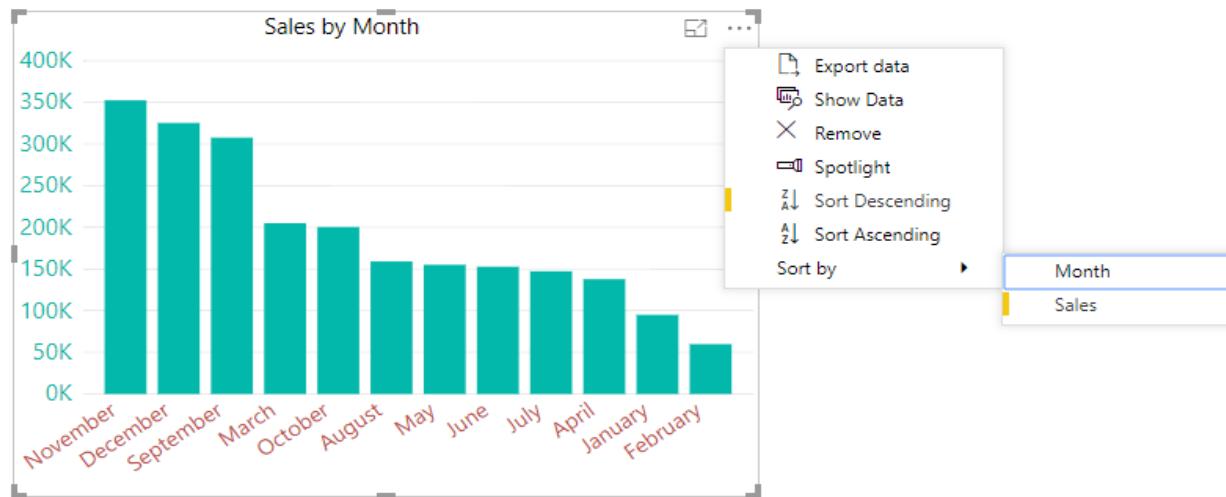
Create a visualization which show each 15 days sales as shown below.



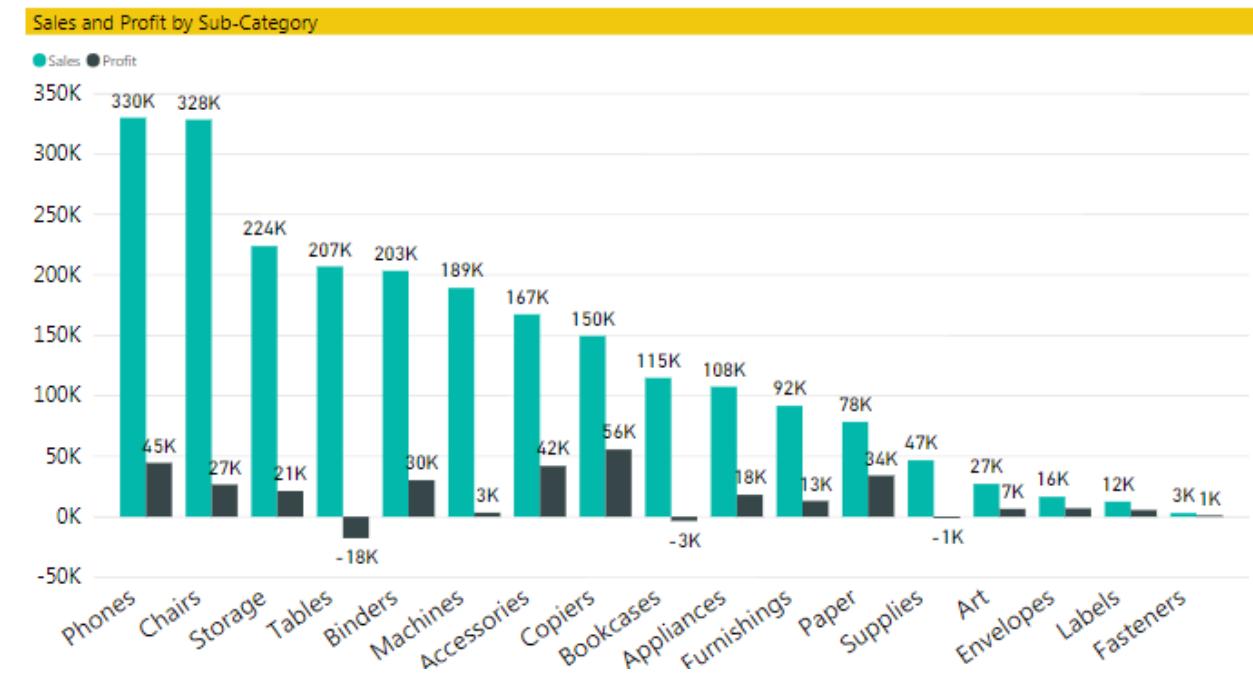
## Sorting Data in Visuals in Power BI Desktop

In Power BI Desktop and the Power BI service, you can change how a visual look by sorting it by different data fields. By changing how you sort a visual, you can highlight the information you want to convey, and ensure the visual reflects that trend (or emphasis).

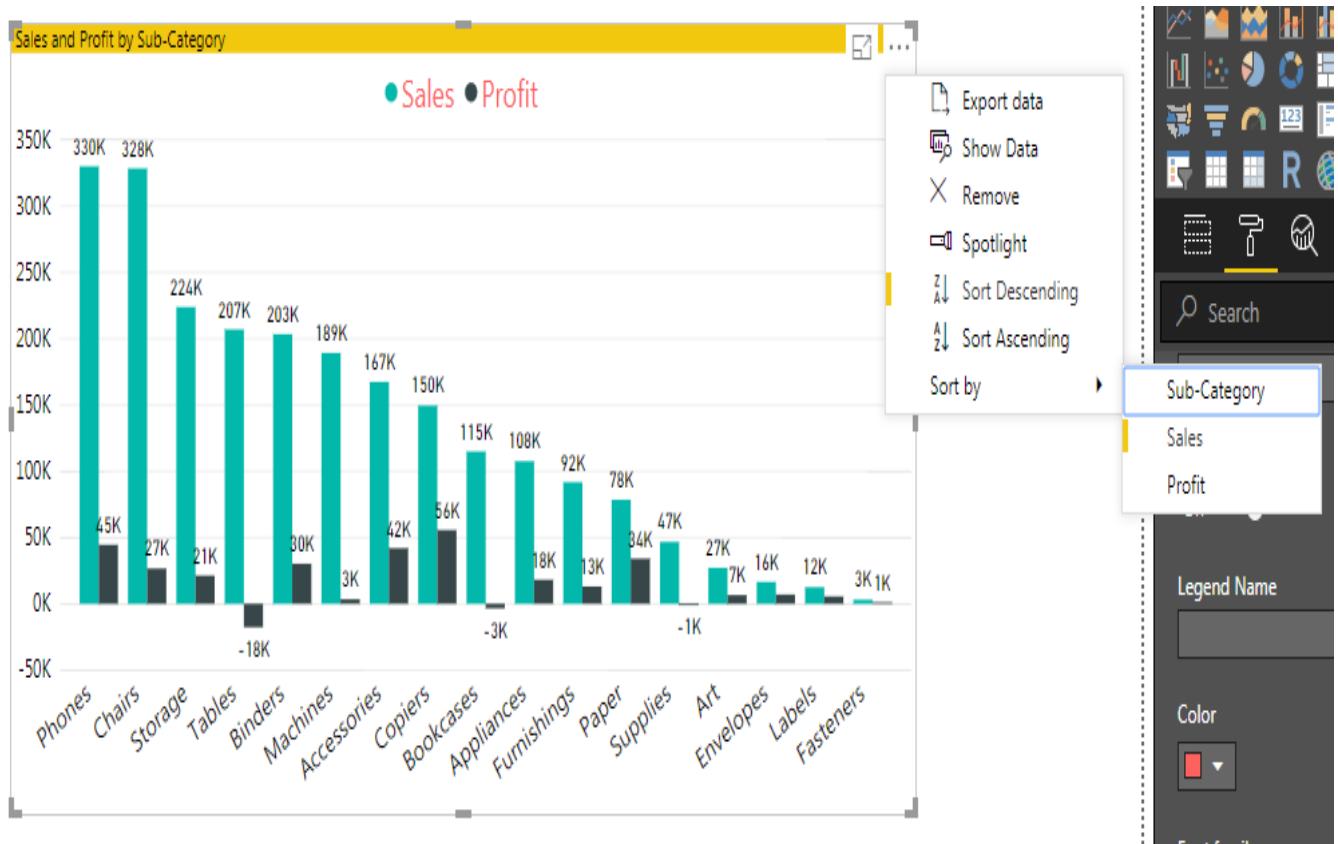
Whether you're using numeric data (such as sales figures) or text data (such as state names), you can sort your visualizations as desired, and make them look like you want them to. Power BI provides lots of flexibility for sorting, and quick menus for you to use. On any visual, select the ellipses menu (...) and then select the field by which you want to sort, as shown in the following image.



The below visualization shows Sales and Profit by Sub-Category. Here's the visualization as it looks before we do any further sorting.



The visual is currently sorted by Sales - we can tell that by matching the color of the descending bars to the legend, but there's a better way to determine the current sort column, the ellipses menu (...) in the upper right corner of the visual. When we select the ellipses, we see the following.

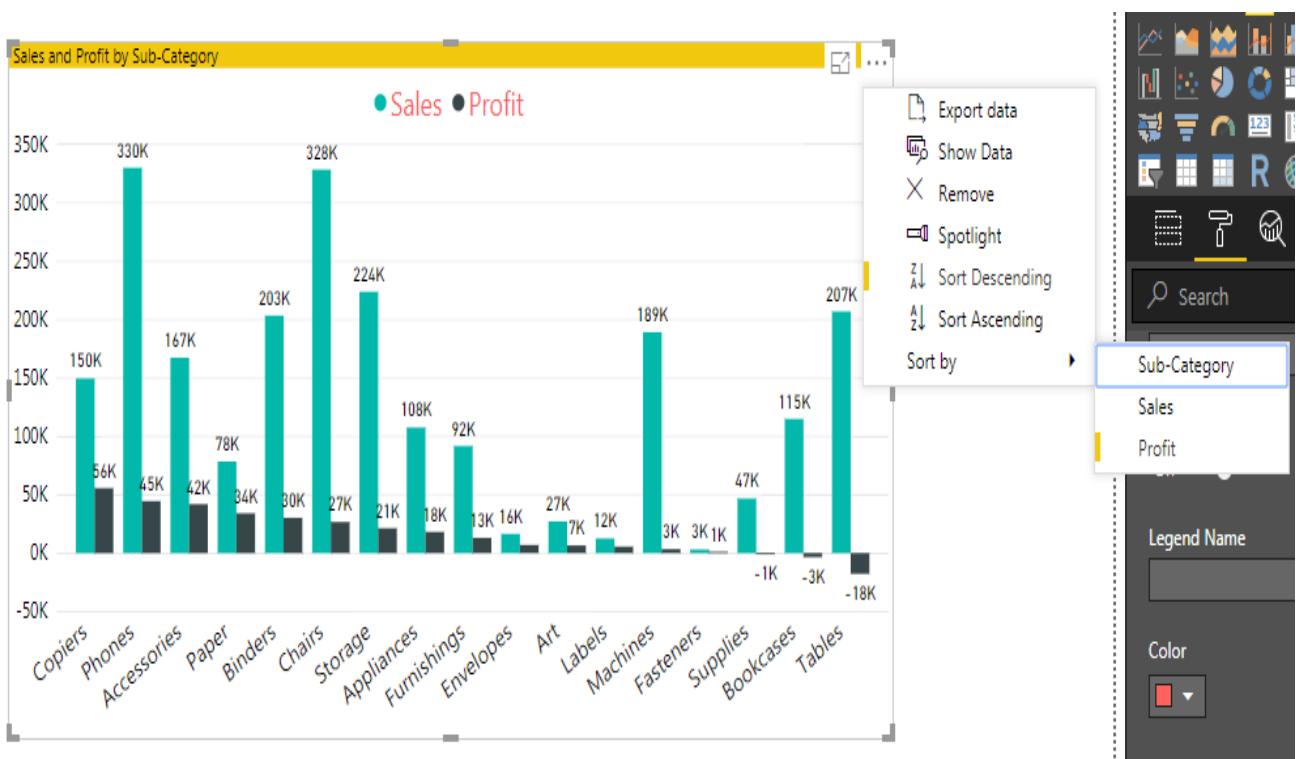


- The current sorting field is Sales, indicated by the facts that **Sort by Sales** has a yellow bar.
- The current sorting direction is largest to smallest, as shown by the little icon Z/A and a down arrow, indicated by the facts that **Z/A Sort Descending** has a yellow bar.

### Changing or selecting which column to use for sorting

In the above Image you noticed the yellow bar beside **Sort by Sales** in the More Options menu, which indicates that the visual is sorted by the Sales column. Sorting by another column is easy - just select the ellipses to show the ellipsis menu, and then select another column.

In the following image, we selected Profit as the column by which we want to sort. Here's what it looks like after we select **Sort by Profit**.



Notice how the visual has changed. The values now are ordered from highest Profit value, in this visual "Copiers", down to "Tables" which has the lowest and Negative Value of Profit.

But what if we want to sort ascending, instead of descending?

#### Selecting the sort order - smallest to largest, largest to smallest

When "Z/A Sort Descending", it means the visual is being sorted by the selected column in order of greatest value to smallest value. Want to change that? No problem - just select "A/Z Sort Ascending", it means the visual is being sorted by the selected column in order of smallest to greatest value.

#### NOTE

Not all visuals can be sorted. For example, the following visuals cannot be sorted: Treemap, Map, Filled Map, Scatter, Gauge, Card, Multi Row Card, and Waterfall.

## Sort using the Sort by Column button

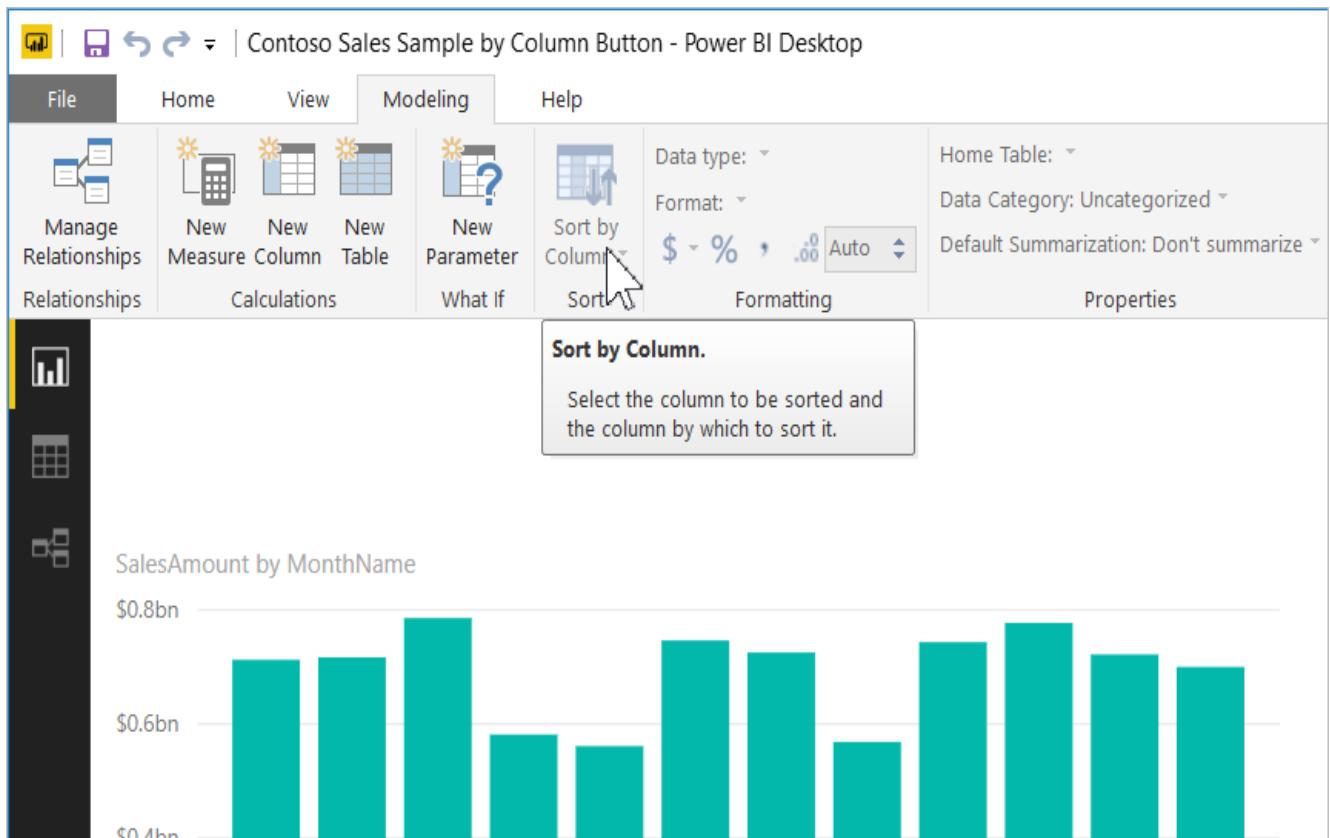
You can even sort the Visualization using the column that is not used in the Visualization using Sort by Column option in The Ribbon under Modeling Tab.

Use the below DAX formulas to create Two New Columns for Getting Month\_Name and Month\_Number from "Order Date" Column. Or else add those two columns in Power Query.

```
MonthName = FORMAT (Orders [Order Date], "MMMM")
```

```
MonthNo = MONTH (Orders [Order Date])
```

There's another way to sort your data, and that's by using the Sort by Column button in the Modeling ribbon.



This approach to sorting requires that you select a column from the Fields pane, and then select the Sort by Column button to choose how (by which column) you want to sort your visual. You have to select the column (field) you want to sort from the Fields pane in order to enable the Sort by Column button - otherwise the button is inactive.

Let's look at a common example: you have data from each month of the year, and you want to sort it based on chronological order. The following steps show you how.

First, notice that when the visual is selected but no column is selected in the Fields pane, the Sort by Column button is inactive (grayed out).

SalesAmount by MonthName

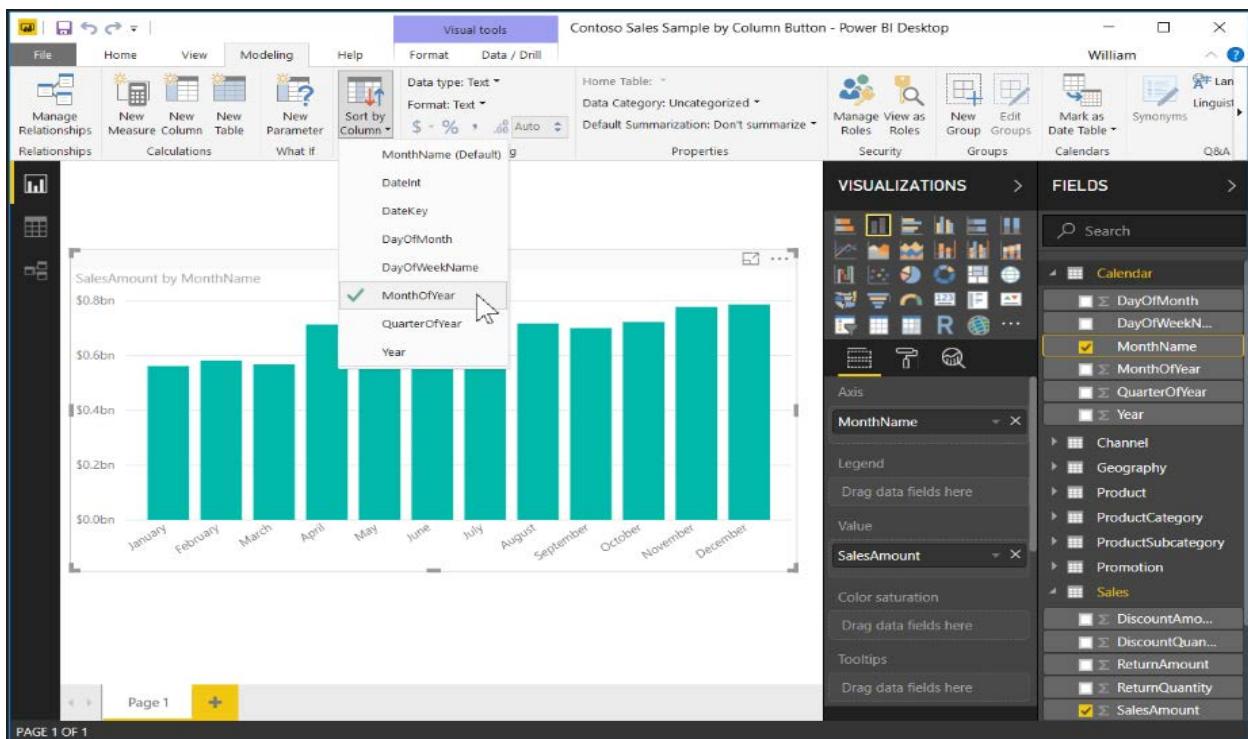
MonthName	SalesAmount
September	\$0.68bn
October	\$0.68bn
November	\$0.75bn
May	\$0.72bn
March	\$0.55bn
June	\$0.68bn
July	\$0.72bn
January	\$0.55bn
February	\$0.58bn
December	\$0.78bn
August	\$0.68bn
April	\$0.68bn

When we select the column by which we want to sort, in the Fields pane, the Sort by Column button becomes active.

SalesAmount by MonthName

MonthName	SalesAmount
April	\$0.68bn
August	\$0.68bn
December	\$0.75bn
February	\$0.55bn
January	\$0.55bn
July	\$0.72bn
June	\$0.68bn
March	\$0.55bn
May	\$0.72bn
November	\$0.78bn
October	\$0.68bn
September	\$0.68bn

Now, with the visual selected, we can select MonthOfYear, instead of the default (MonthName), and the visual now sorts in the order we want: by the month of the year.



And that's it. Remember that you must select a column in the Fields pane for the Sort by Column button to become active.

## Create tooltips based on report pages in Power BI Desktop

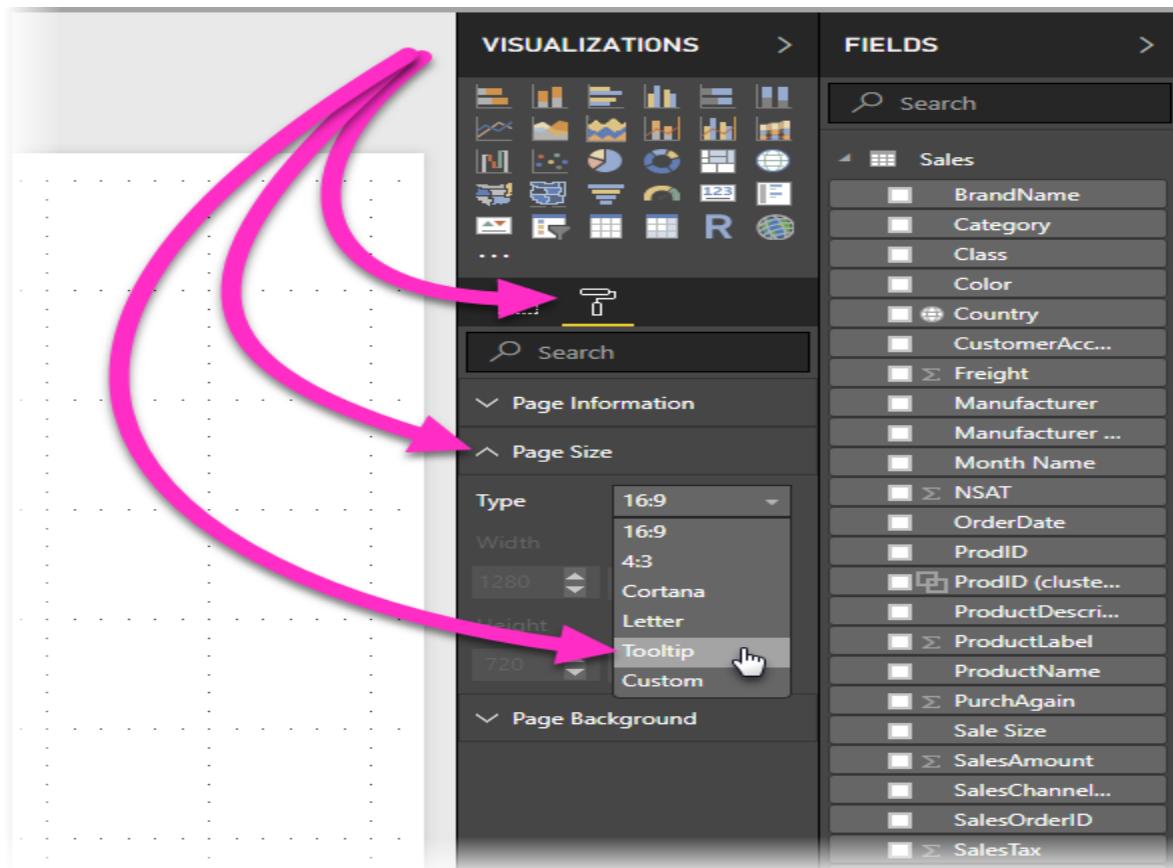
You can create visually rich report tooltips that appear when you hover over visuals, based on report pages you create in Power BI Desktop. By creating a report page that serves as your tooltip, your custom tooltips can include visuals, images, and any other collection of items you create in the report page.

You can create as many tooltip pages as you want. Each tooltip page can be associated with one or more fields in your report, so that when you hover over a visual that includes the selected field, the tooltip you created on your tooltip page appears when you hover over the visual, filtered by the data point over which your mouse is hovering.

### Create a report tooltip page

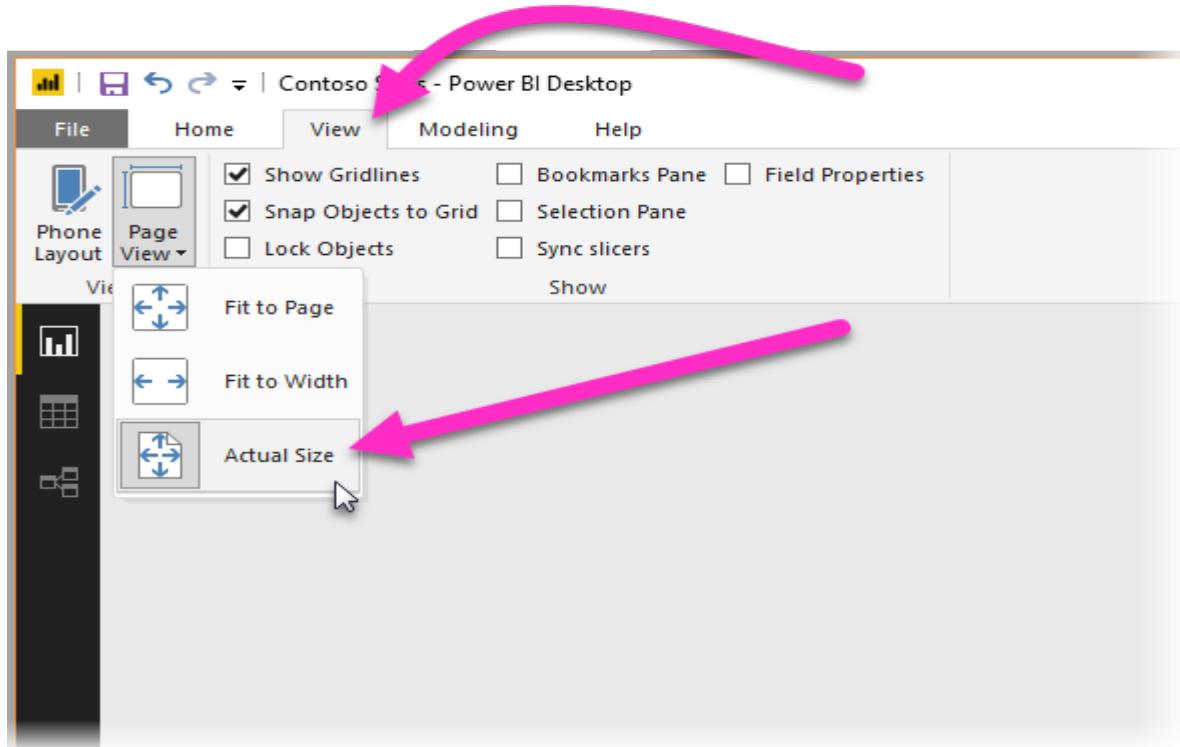
Create a new report page by clicking the + button, found along the bottom of the Power BI Desktop canvas, in the page tabs area. The button is located beside the last page in the report.

Your tooltip can be any size, but keep in mind that tooltips hover over the report canvas, so you might want to keep them reasonably small. In the Format pane in the Page Size card, you can see a new page size template called Tooltip. This provides a report page canvas size that's ready for your tooltip.



By default, Power BI Desktop fits your report canvas to the available space on the page. Often that's good, but not in the case of tooltips. To get a better sense and view of what your tooltip will look like when you're done, you can change the Page View to actual size.

To do that, select the View tab from the ribbon. From there, select Page View → Actual Size, as shown in the following image. Also name the report page (Tooltip1 or Year wise sales) so its purpose is clear.



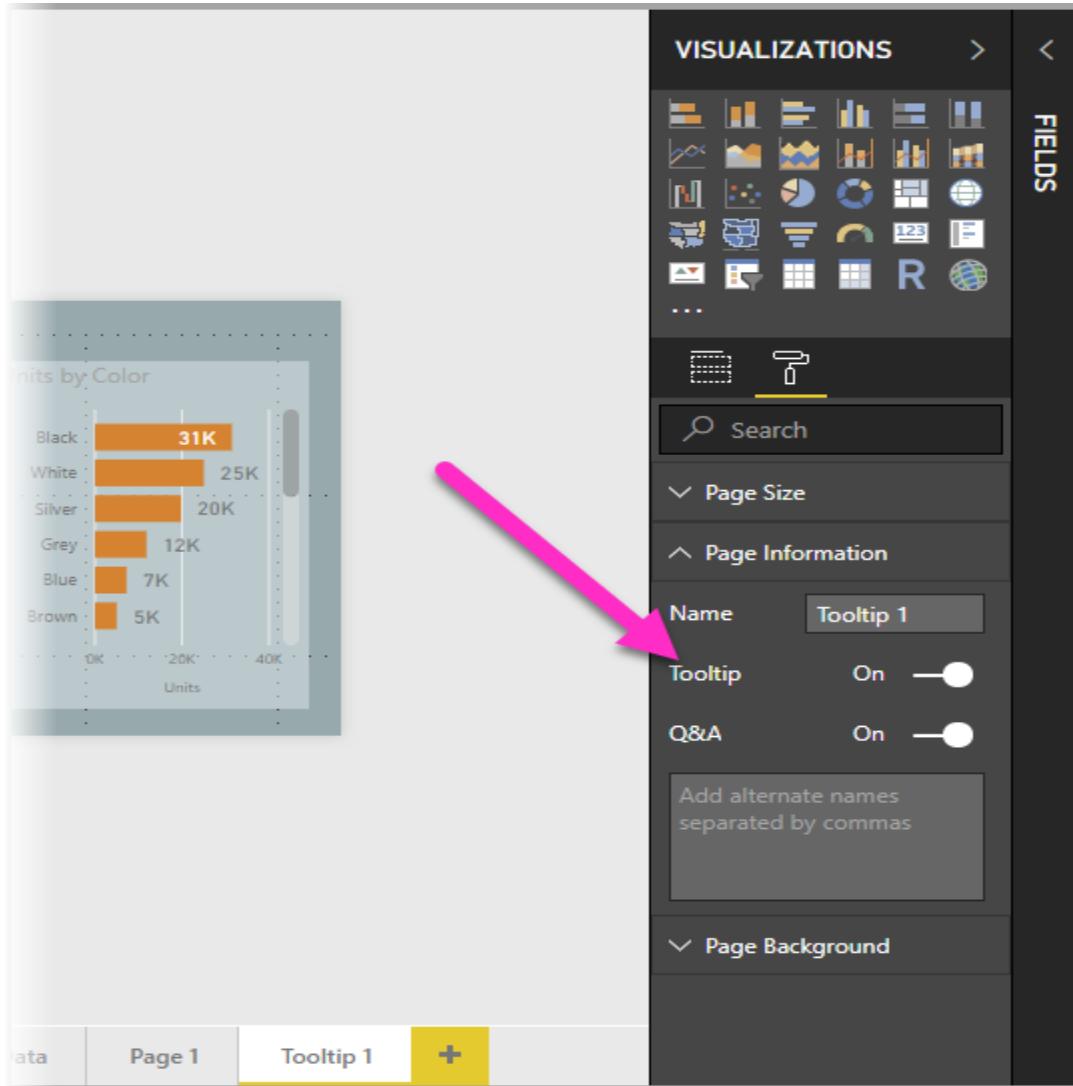
From there, you can create whatever visuals you would like to show up in your tooltip as shown below.



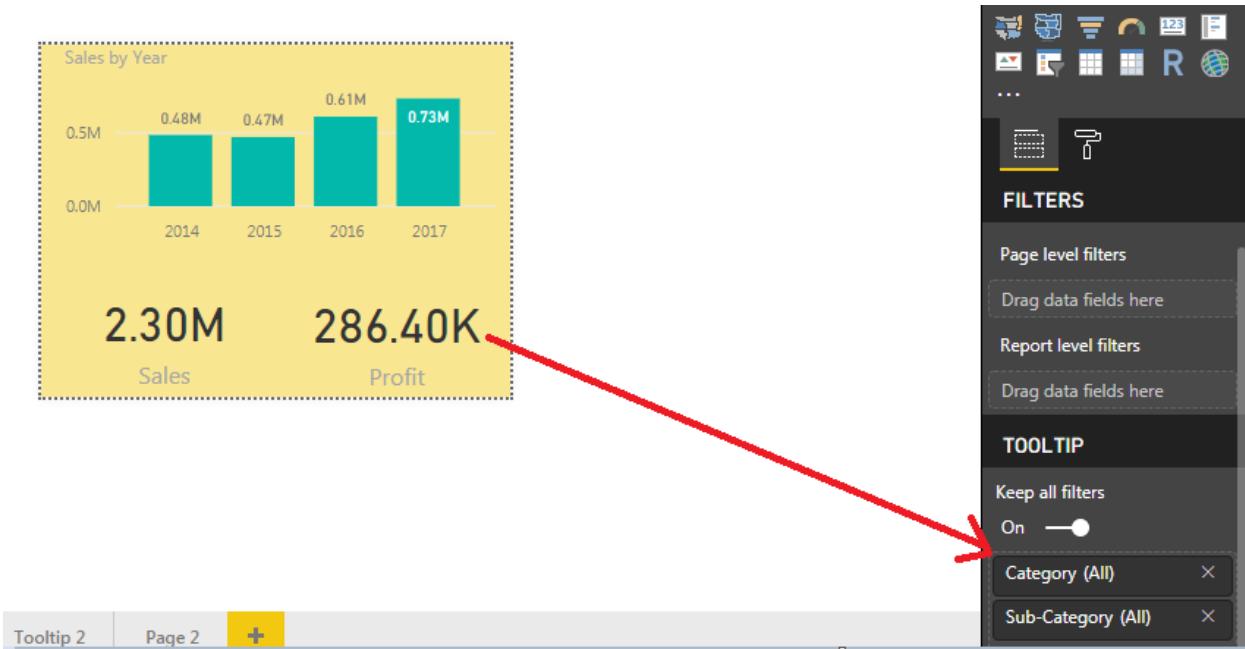
## Configure your tooltip report page

Once you have the tooltip report page created, you need to configure the page in order for Power BI Desktop to register it as a tooltip, and to ensure it appears in over the right visuals.

To begin with, you need to turn the Tooltip slider to ON, in the Page Information card, to make the page a tooltip.



Once that slider is set to ON, you specify the fields for which you want the report tooltip to appear. For visuals in the report that include the field you specify, the tooltip will appear. You specify which field or fields apply by dragging them into the Tooltip fields bucket, found in the Fields section of the Visualizations pane. In the following image, the Category and Sub-Category fields has been dragged into the Tooltips fields bucket. You can include both categorical and numerical fields in the Tooltips fields bucket, including measures.



Once completed, the tooltip report page you created will be used as a tooltip in visuals in the report that use any fields you placed into the Tooltips fields bucket, replacing the default Power BI tooltip.

### Manually setting a report tooltip

In addition to creating a tooltip that automatically appears when hovering over a visual that contains the specified field, you can manually set a tooltip.

Any visual that supports report tooltips now has a Tooltip card in its Formatting pane.

To set a tooltip manually, select the visual for which you want to specify the manual tooltip, then in the Visualizations pane, select the Format section and expand the Tooltip card.

Then, in the Page dropdown, select the tooltip page you want to use for the selected visual. Note, only report pages that are specified as Tooltip pages show up in the dialog box.

### Reverting to default tooltips

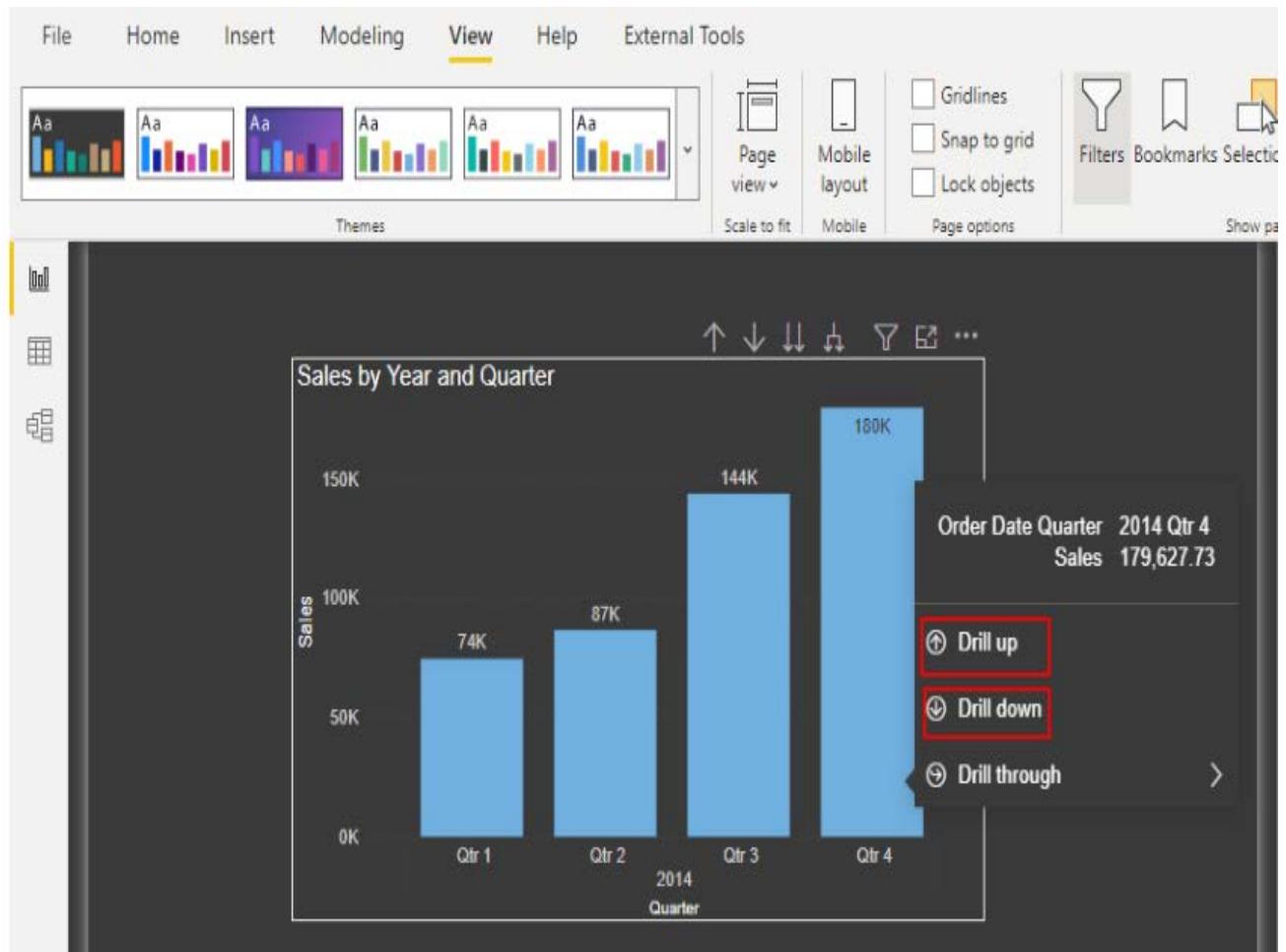
If you create a manual tooltip for a visual but decide you want the default tooltip instead, you can always return to the default tooltip that Power BI provides. To do so, when a visual is selected and the Tooltip card is expanded, just select Auto from the Page dropdown to go back to the default.

## Modern Visual Tooltips

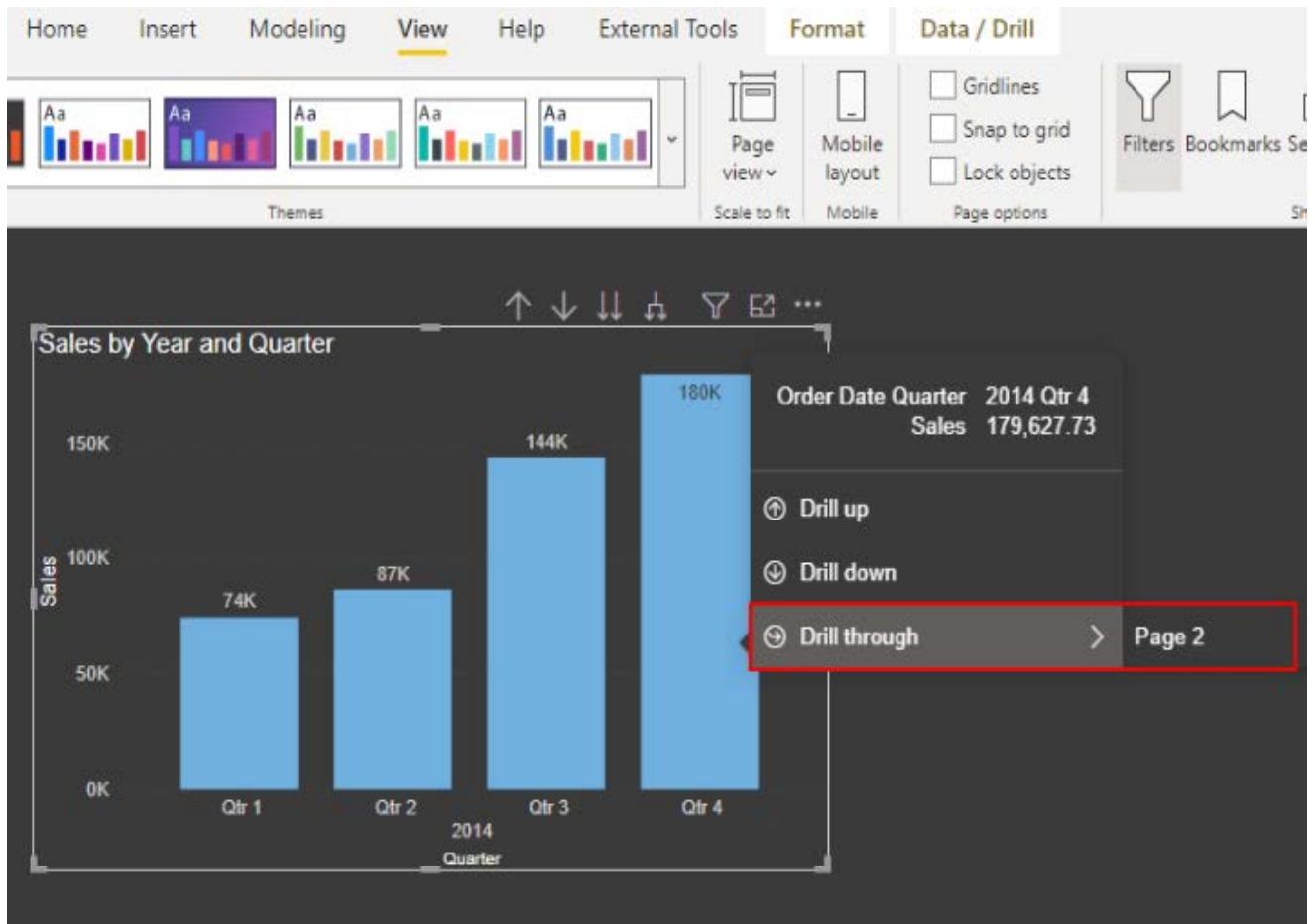
Modern Visual Tooltips includes Data Point Drill Actions (Drill Down and Drill Up, DrillThrough) and updated styling to match your Report Theme.

When you enable these new tooltips

- ✓ You can Drill Down and Drill Up the visual without using the Drill Actions in the header for the visual.



- ✓ You can DrillThrough on a Data Point without having to use the right-click menu.



## Bookmarks, Selection Pane & Buttons

### Bookmarks

Bookmark saves the state of the Page, exactly as it is at the time of creating it.

That means you can then select the Bookmark and see the Page with the state that you have saved it.

Once you have a list of bookmarks, you can use those in several ways like organizing presentations of your report, creating report navigation, creating an Index page in report and more.

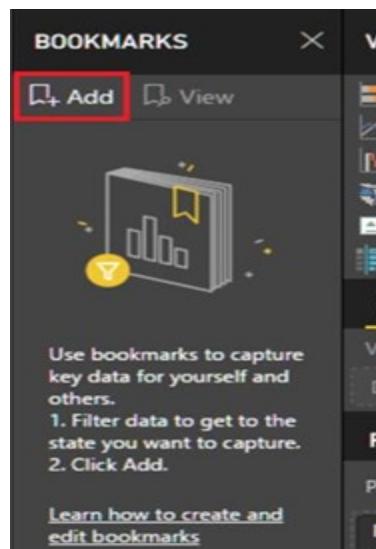
When you add a bookmark to your report, it will include the following

- The current page
- Filters
- Slicers
- Sort order
- Drill location
- Visibility
- Any of the “focus” modes

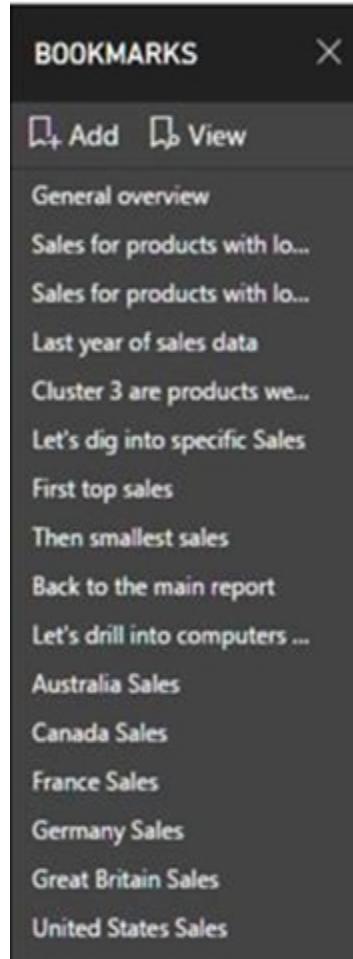
You can Open Bookmarks Pane by opening the Show panes, which can be found under the View tab.



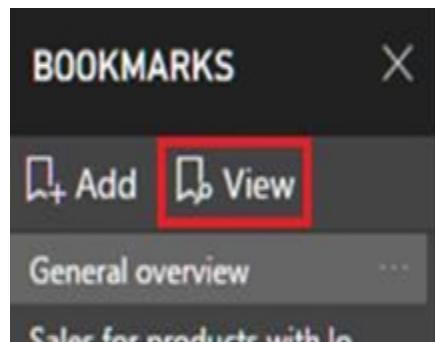
Once you Open Bookmarks Pane we can the add Bookmarks. Set up your report to look how you want, with all the filters properly set, and click the Add button under Bookmarks.



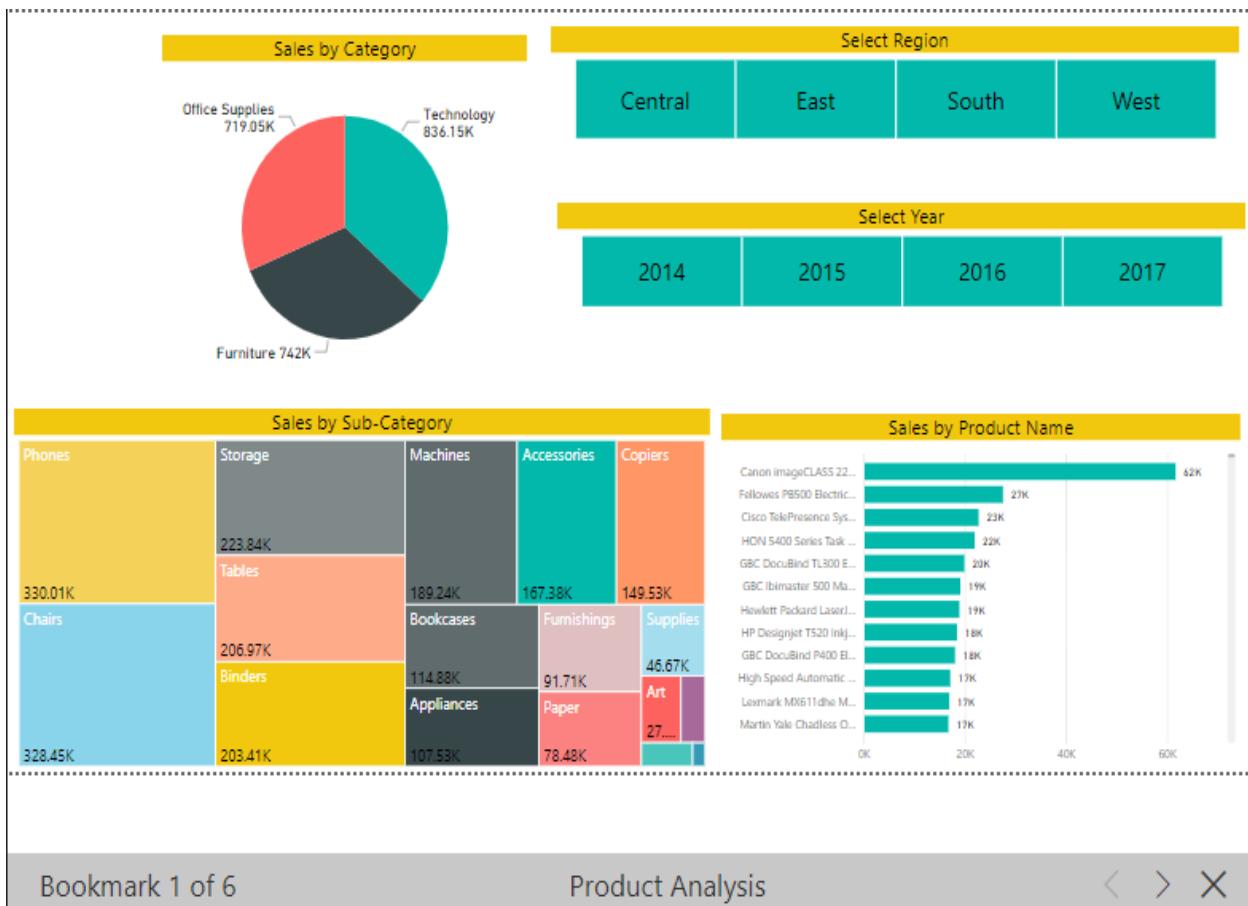
This will add a bookmark to pane, which you can rename and go back to whenever you want. You can continue adding as many bookmarks as you want.



If you want to use your bookmarks as a story or in a presentation, you can use the View option to enter into a view mode for bookmarks.



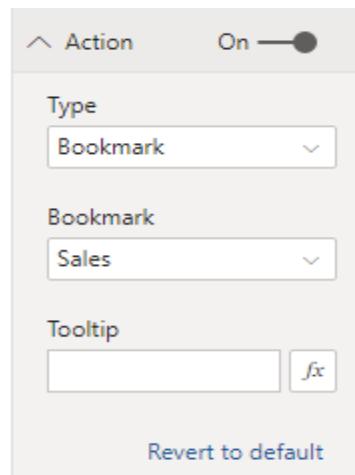
When in this mode, there is a title bar for each bookmark that includes the bookmark name and navigation arrows. At this point, you can close the bookmarking pane if you want.



### Bookmark links for buttons, shapes and images

You can link buttons, shapes and images to bookmarks. This is a great feature for creating custom navigation in your reports or creating buttons that perform different actions across your report.

You'll find the option in the formatting pane under the Action. You can pick either Back or Bookmark as your Action type. If you pick bookmark, you'll have a dropdown to pick one of the bookmarks you currently have in your report.



Just like the back button, you can activate bookmark buttons by pressing “ctrl+click” while in editing mode or just clicking in reading mode.

## Selection Pane

You can hide or unhide visuals in a Power BI report through the selection pane. There is an eye icon besides every visual, which gives you the control over visibility of that visual on the report page.

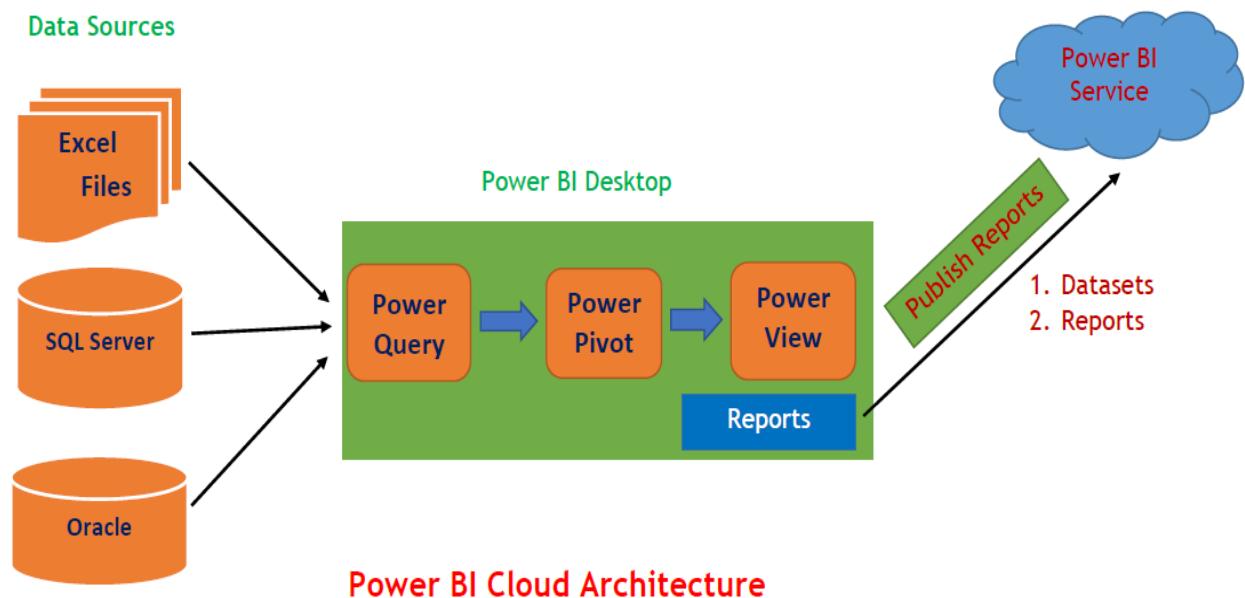
## Buttons

Buttons are action objects in Power BI. You can create a button (or even an image or shape), and then set the action of that to be; back, Q&A, or bookmark. In this post, we are only talking about the bookmark action feature.

### Changing the Chart Type Dynamically

## Power BI Service Introduction

- ✓ Power BI Service is a Cloud Based Business Intelligence Application or Software from Microsoft Corporation.
- ✓ Power BI Service is Constructed upon and secured by the Windows Azure Cloud platform.



## Power BI Service Usage

Still now we created reports using the Power BI Desktop client. Now, it's time to share those reports with your team, company, or customers using Power BI Service.

- ✓ Power BI Desktop Solution can be published to Power BI Service for Admin related activities.
- ✓ The Reports Developed in Power BI Desktop is published to Power BI Service for Dashboard development.
- ✓ You can build Reports directly by selecting Data Sources on Power BI Service as well. From Power BI Desktop you can connect Live to Power BI datasets that are published to Power BI Service to create New Reports.
- ✓ The Datasets prepared in Power BI Desktop published to Power BI Service and can be scheduled to refresh, if you need any additional Reports with this Datasets you can generate and can be used in Dashboards.
- ✓ With Power BI Service you can manage the Reports and Dashboards; you can share the Reports and Dashboards with the end users or Business users.
- ✓ Reports or Dashboards shared from Power BI Service can be accessed anywhere from the world.
- ✓ You can view the Reports or Dashboards that are shared with you with the help of web browsers or Mobile Application (Power BI Mobile).

## Creating Power BI Service Account

Before we move forward, make sure you get an account at Power BI (<https://powerbi.microsoft.com/>) using the following steps.

Open the above link → Select START FREE



The image shows a yellow banner with the text "Business intelligence like never before". Below it, a subtext reads: "Go from data to insights in minutes. Any data, any way, anywhere. And all in one view." A red-bordered "START FREE >" button is centered below the subtext.

Scroll Down; go to Cloud collaboration and sharing → TRY FREE. Sign Up with your work email address.

### POWER BI

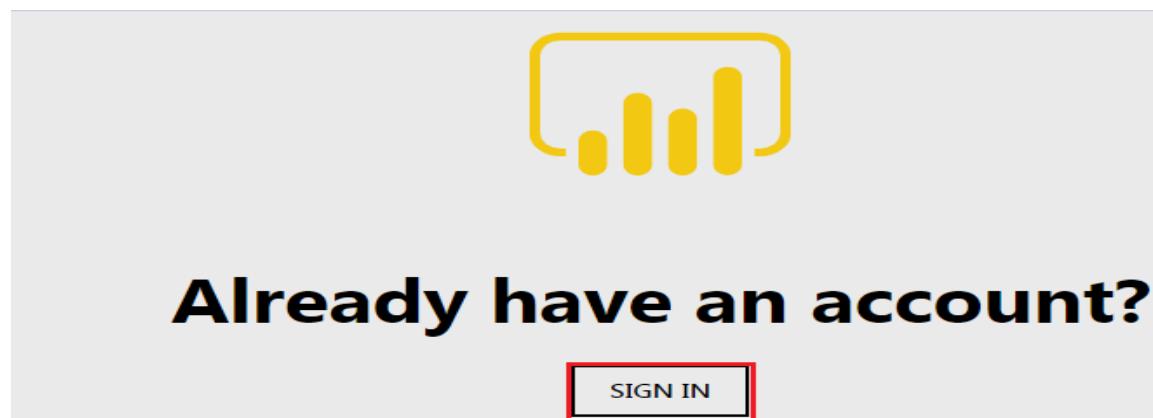
## Cloud collaboration and sharing

Use Power BI Pro to share and distribute reports with others, without any complicated setup. Get started now with a free 60-day trial of Power BI Pro.

TRY FREE >

SIGN IN to Power BI Service Account

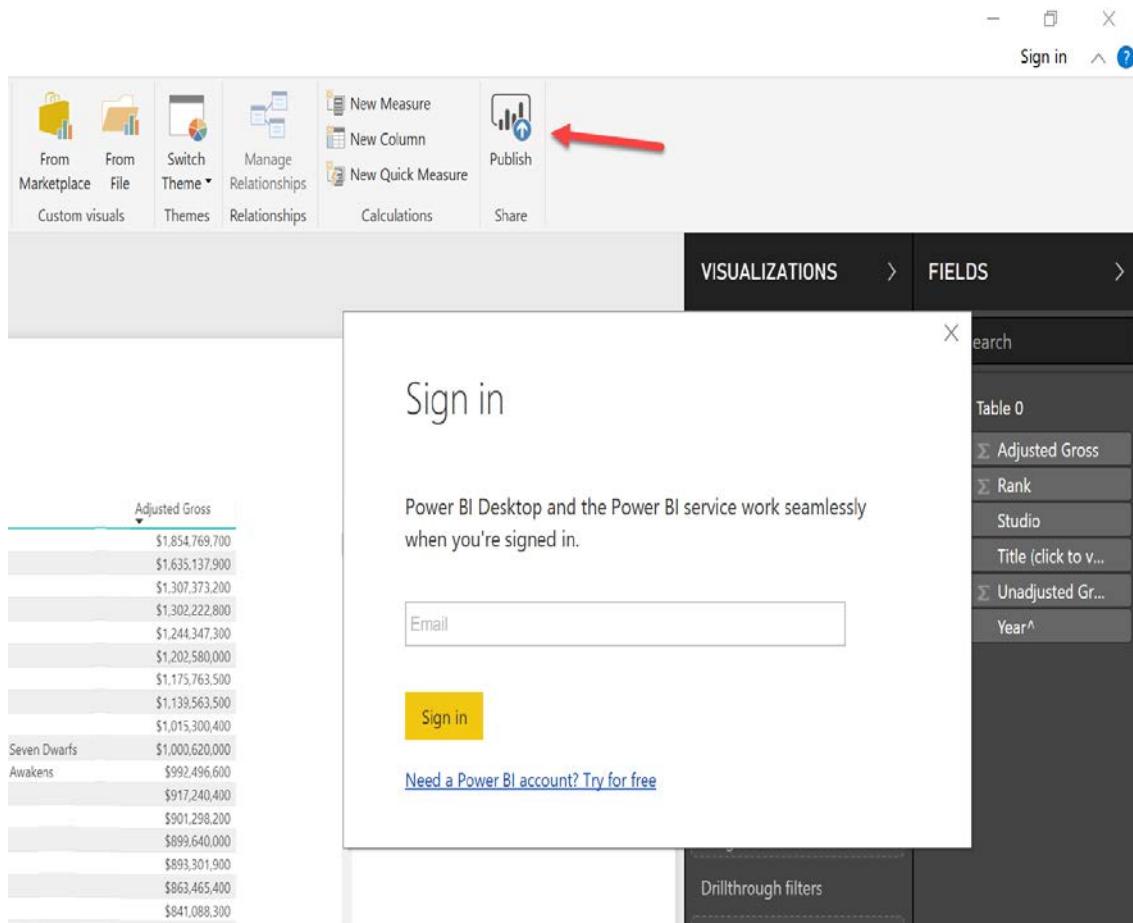
Open Power BI Service using (<https://app.powerbi.com/>) and SIGN IN



The image shows the Power BI sign-in page. It features a yellow bar chart icon. Below the icon, the text "Already have an account?" is displayed in large, bold, black font. A red-bordered "SIGN IN" button is located at the bottom of the page.

## Deploying / Publishing Reports to the Power BI service

There are numerous ways to publish a report to the Power BI Service, but the easiest way is by using the Power BI desktop. To do this, you'll need to simply click the Publish button in the desktop application, as shown in the following screenshot. If you have not previously signed in with your Power BI Service account, you will be prompted to create one or sign in with an existing account.



You'll then be asked which workspace you want to deploy to. A **Workspace** is an area in the Power BI service that is much like a folder, where you can bundle your reports, datasets, and dashboards. You can also assign security to the workspace and not have to worry about securing each item. Most importantly, it allows for team development of a Power BI solution, where you can have multiple authors on a solution.

At this point, select the **My Workspace** item, which will send the report and its data to your personal workspace. The report will then deploy to the Power BI service. The amount of time this takes will depend on how large your dataset is.

If you open the report, Power BI Service will launch and show you the same report that you were viewing in the desktop in a web browser. You'll also be able to immediately see the report in the Power BI mobile app from your Android or iPhone.

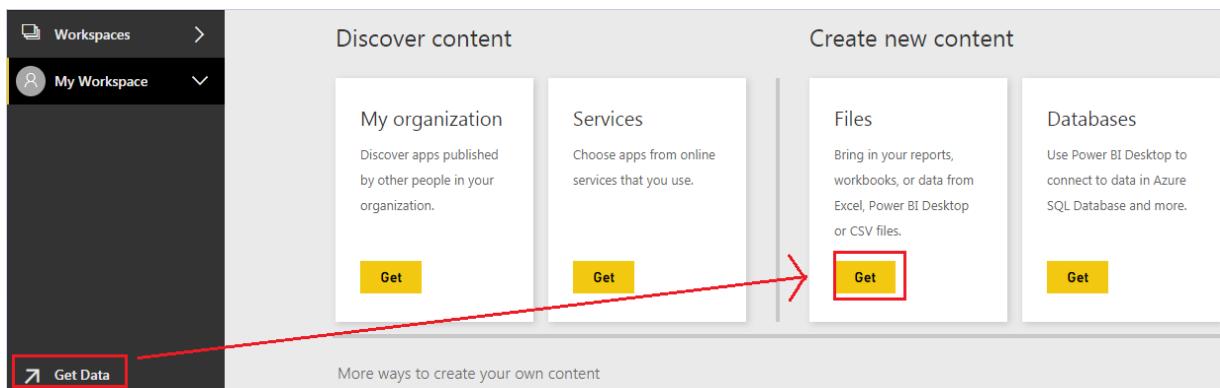
## Other way to publish data to PBI Service / Get the Report to PBI Service

Logon to Power BI Service using below link

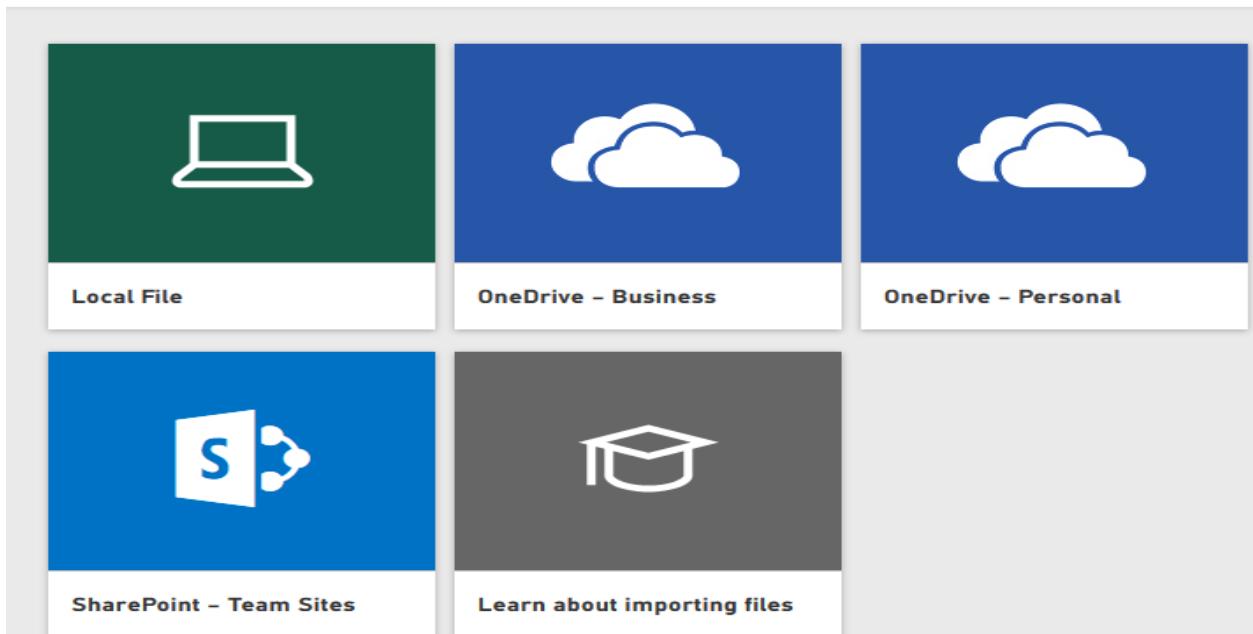
<https://app.powerbi.com>

Open Workspace where you need the Report to be published or imported. At this point, select the My Workspace item, which will send the report and its data to your personal workspace.

Select Get Data → Under Create new content and Under Files Select Get



Select Local Files from the below selection.



Browse for the .Pbix file → Press Open.

Once you press open the PBIX file will come to selected Workspace.

Furthermore, if the workspace you are publishing to already contains a dataset with the same name, you will be asked if you want to replace it. By default, the report will be published to My Workspace.

Once you are done, you can go to Power BI service and see the report. Now you can see the Dataset used for that report Development in the Datasets Section and Report you Published in Reports Section along with an Empty Dashboard in Dashboards Section.

## My Workspace Tabs / APP WORKSPACES TAB

Power BI Service My Workspace / APP WORKSPACES has Four Key Areas you can interact with



### DATASETS

This is the data that you have built in the Power BI desktop. When you click the datasets, you can also build new reports from those datasets in Power BI Service. You can also connect to this Dataset Live from Power BI Desktop if you have access to that workspace to build the New Reports using Power BI Desktop using this dataset.

### WORKBOOKS

You can upload Excel workbooks into this area. These Excel workbooks can be used as a dataset or can form pieces of the workbook that can be pinned to a dashboard.

### REPORTS

This refers to what you have built in the Power BI desktop. These reports can be explored, modified, or downloaded in this section.

### DASHBOARDS

You can pin the best elements from multiple reports into a unified set of dashboards. These dashboards are the first thing most of your casual users will interact with.

## Working with Datasets

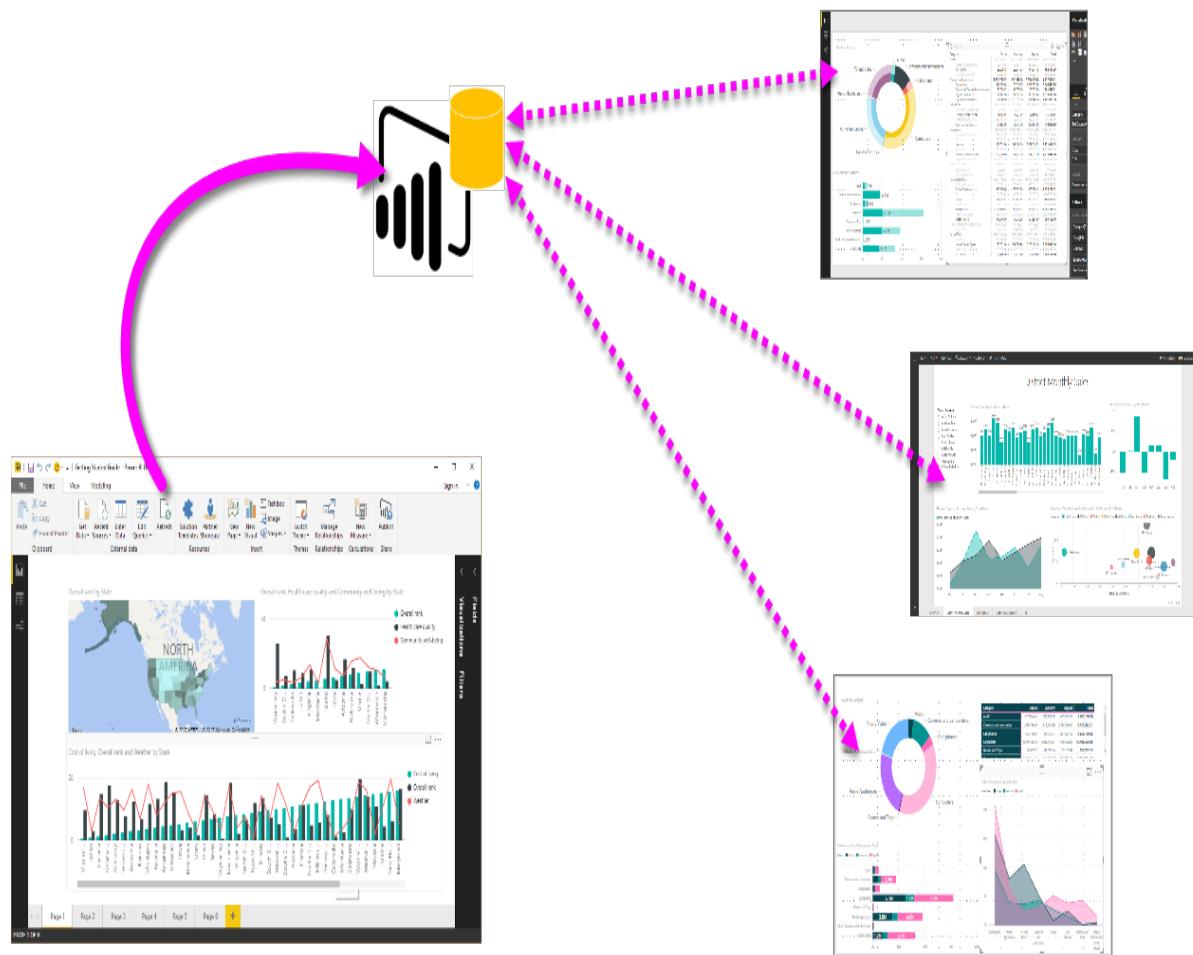
The DATASETS area of Power BI holds the data that makes up your reports. In the dataset when you select **Create Report** Option, the designer opens to build reports from the dataset. The designer can be used to do the following things.

- ✓ Create more Quick Insights
- ✓ Create new reports
- ✓ Refresh or schedule refreshes
- ✓ Download the Power BI Desktop file (.pbix)

When you start with a dataset, users can create new reports from your data, even when accessing it through the web. The entire user interface will feel nearly identical to Power BI Desktop, but you will be lacking the ability to modify the model, query, and relationships. The best part of building reports here is that you have a central dataset that IT can own, modify, and make human-readable so that the entire organization can build reports off of it.

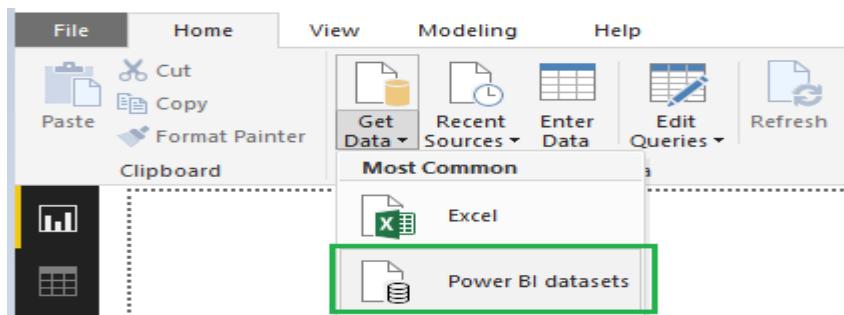
### Power BI Datasets Live Connection

You can establish connection to a shared dataset in the Power BI service, and create many different reports from the same dataset. This means you can create your perfect data model in Power BI Desktop, publish it to the Power BI service, then you and others can create multiple different reports (in separate. pbix files) from that same, common data model. This feature is called Power BI datasets Live connection.

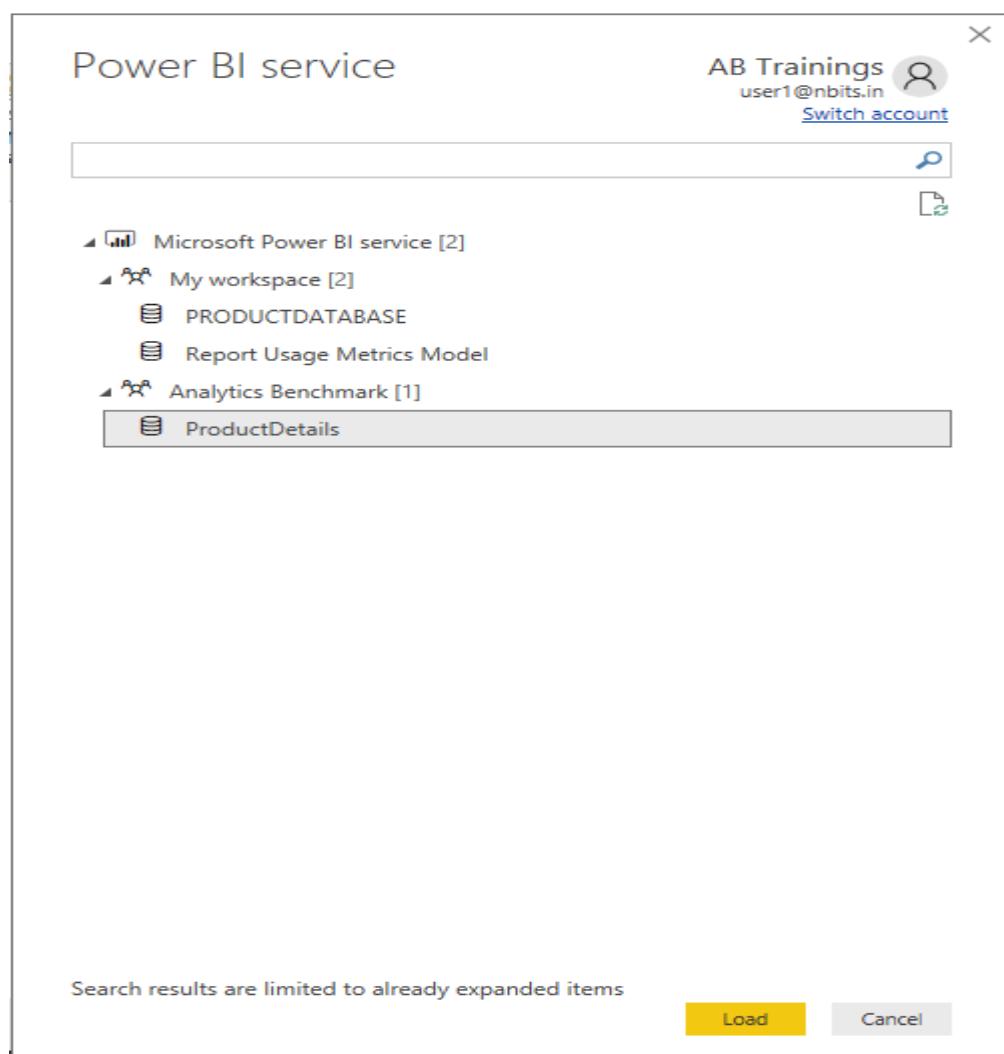


To create Shared dataset, you need to create a dataset and report in Desktop and publish it in a workspace which is common to all. Example in a project if 4 Members need to build a Reports on the shared dataset, create a **App Workspace** specific to this project and provide edit permissions to the 4 Members. Now publish the Dataset from Desktop to this Workspace.

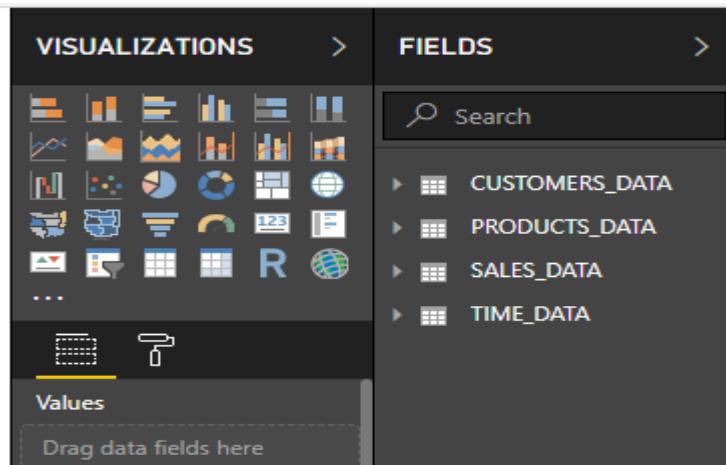
Establish a Power BI datasets live connection to the published dataset.



If you're not signed in to Power BI, you'll be prompted to do so. Once logged in, you're presented with a window that shows which workspaces you're a member of, and you can select which workspace contains the dataset to which you want to establish a Power BI datasets live connection. Here I will connect to **ProductDetails**.



Click on load and the dataset will be loaded as shown below and you can create the reports and publish it.



Below are some limitations to this as well

- Read-only members of a workspace cannot connect to datasets from Power BI Desktop.
- Only users who are part of the same Power BI service workspace can connect to a published dataset using the Power BI service live connection. Users can (and often do) belong to more than one workspace.

## Creating and interacting with dashboards

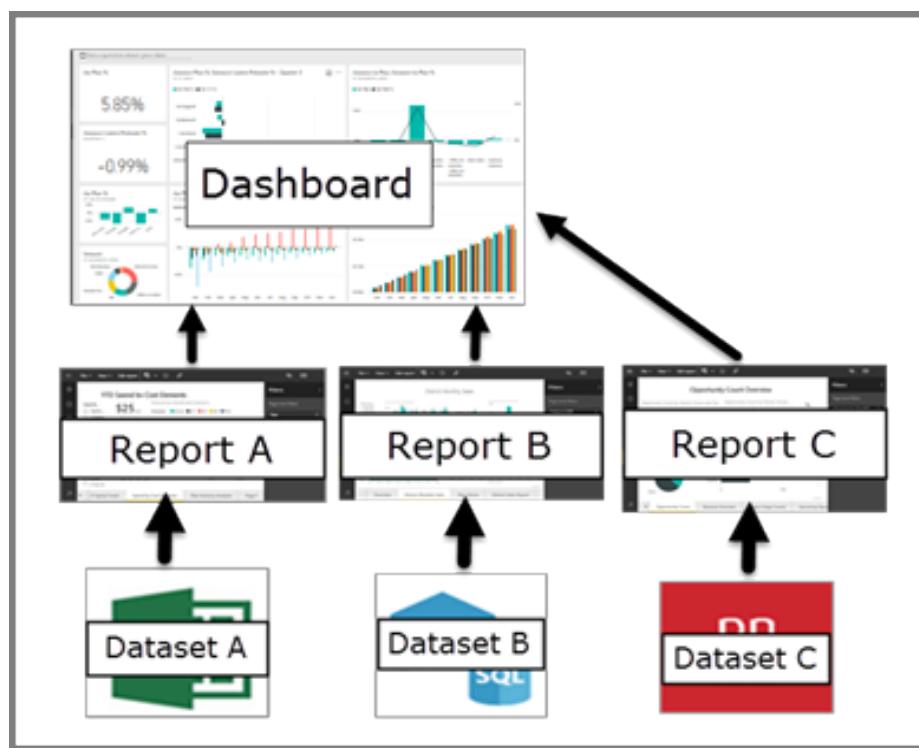
### Dashboards in Power BI service

A Power BI dashboard is a single page, often called a canvas, that uses visualizations to tell a story. Because it is limited to one page, a well-designed dashboard contains only the most-important elements of that story.



The visualizations you see on the dashboard are called tiles and are pinned to the dashboard from reports.

Once you have deployed your datasets and are using them in reports, you're ready to bring together the many report elements into a single dashboard. Often, your management team is going to want to have a unified executive dashboard that combines elements such as your sales numbers, bank balances, customer satisfaction scores, and more into a single dashboard. The visualizations on a dashboard come from reports and each report is based on one dataset. In fact, one way to think of a dashboard is as an entryway into the underlying reports and datasets. The amazing thing about dashboards in Power BI is that data can be actionable and reacted too quickly. You can click on any dashboard element and be immediately taken to the report that makes up that number.



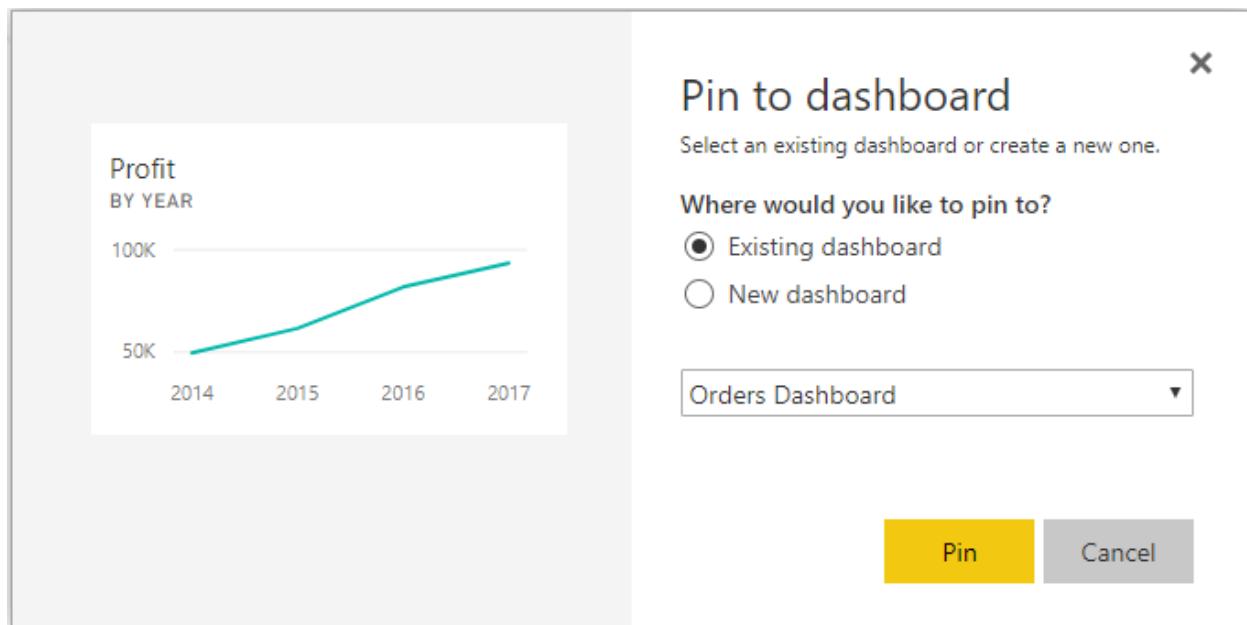
### Advantages of dashboards

Dashboards are a wonderful way to monitor your business, to look for answers, and to see all of your most-important metrics at a glance. The visualizations on a dashboard may come from one underlying dataset or many, and from one underlying report or many. A dashboard combines on-premises and cloud-born data, providing a consolidated view regardless of where the data lives.

A dashboard isn't just a pretty picture, it's highly interactive and highly customizable and the tiles update as the underlying data changes.

## Creating dashboard and Add Text and Images

To create dashboard, start by opening a report that has some interesting data. On each of the charts, tiles, and other elements, you'll see a pin icon in the top right of that object. After you click on the pin, it will ask you what dashboard you wish to pin that report element to. You can, at that point, select a new dashboard to create or choose an existing dashboard to add the element to, as shown in the following screenshot. This is what makes Power BI so magical—you're able to append data from your accounting department next to data from your sales and customer service teams, giving your executives one place to look.



## Who can create a dashboard?

Creating a dashboard is a creator feature and requires edit permissions to the report. Edit permissions are available to report creators and to those colleagues the creator grants access. For example, if USER1 creates a report in workspace ABC and then adds USER2 as a member of that workspace, USER1 and USER2 will both have edit permissions. On the other hand, if a report has been shared with USER3 directly or as part of a Power BI app (you are consuming the report), you won't be able to pin tiles to a dashboard.

## Add text and images

A Power BI dashboard is a single-page collection of visuals, which can be visuals from reports, as well as other objects such as text and videos. Dashboards are a feature of Power BI service; they are not available in Power BI Desktop. In the context of dashboards, visuals are called tiles, and the process of adding a tile to a dashboard is called pinning.

The dashboard that is created automatically after you publish a report contains one tile only, which also has the same title as the report. If you click on the tile, you will be taken to the report.

There are several ways in which you can add more tiles to a dashboard. When you are in a report and hover over a visual, you will see a pin icon.

Another way to add a tile is to click on the Add Tile button when you are in a dashboard, this gives you the following options

- Web Content
- Image
- Text Box
- Video
- Custom Streaming Data

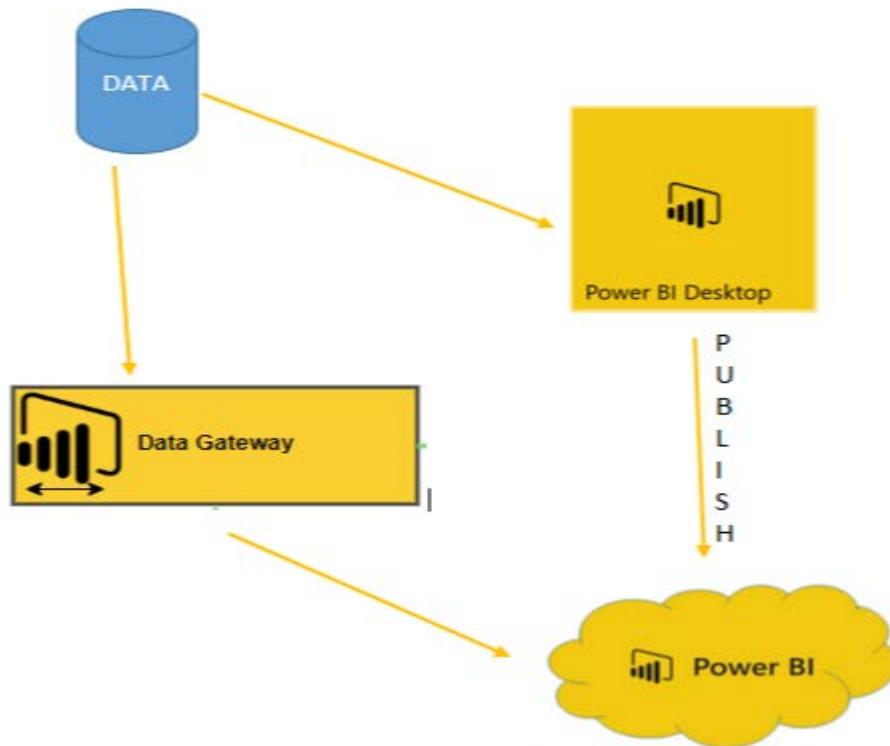
While you can pin an image visual from a Power BI report, the Add Tile option allows you to pin an image that is not contained in any report. When you select Image and click Next, you will be able to specify the URL of an image. Optionally, you can specify a title and subtitle. Clicking Apply will pin the image to your dashboard.

To add text to your dashboard, you can either pin an existing text visual from a report or choose the Text Box option in the Add Tile menu in a dashboard. If you choose the latter, you will be presented with a text editor, where you can use rich text formatting and create hyperlinks.

## Access on-premises data from Power BI Service - Data Gateway

### What are Power BI data gateways?

When you create your reports based on on-premises data and publish them online or Power BI Service, to refresh your datasets, you will need a way to access your on-premises data sources. This can be achieved with a data gateway.



### Connect to a data source by using a data gateway

In most cases, to share the reports you created in Power BI Desktop, you need to publish them to Power BI service in the cloud. Once this happens the mechanics of refreshing your dataset change, which means the cloud, not your machine, needs to have access to your data sources.

A gateway is a piece of software that acts as a bridge between the cloud and your on-premises data sources. With a data gateway, you can not only access your on-premises data sources but also schedule refresh for the datasets published to Power BI service.

A Power BI gateway is software that you install within an on-premises network, it facilitates access to data in that network. It's like a gatekeeper that listens for connection requests, and grants them only when a user's requests meet certain criteria. This lets organizations keep databases and other data sources on their on-premises networks, yet securely use that on-premises data in Power BI reports and dashboards.

A gateway can be used for a single data source or multiple data sources.

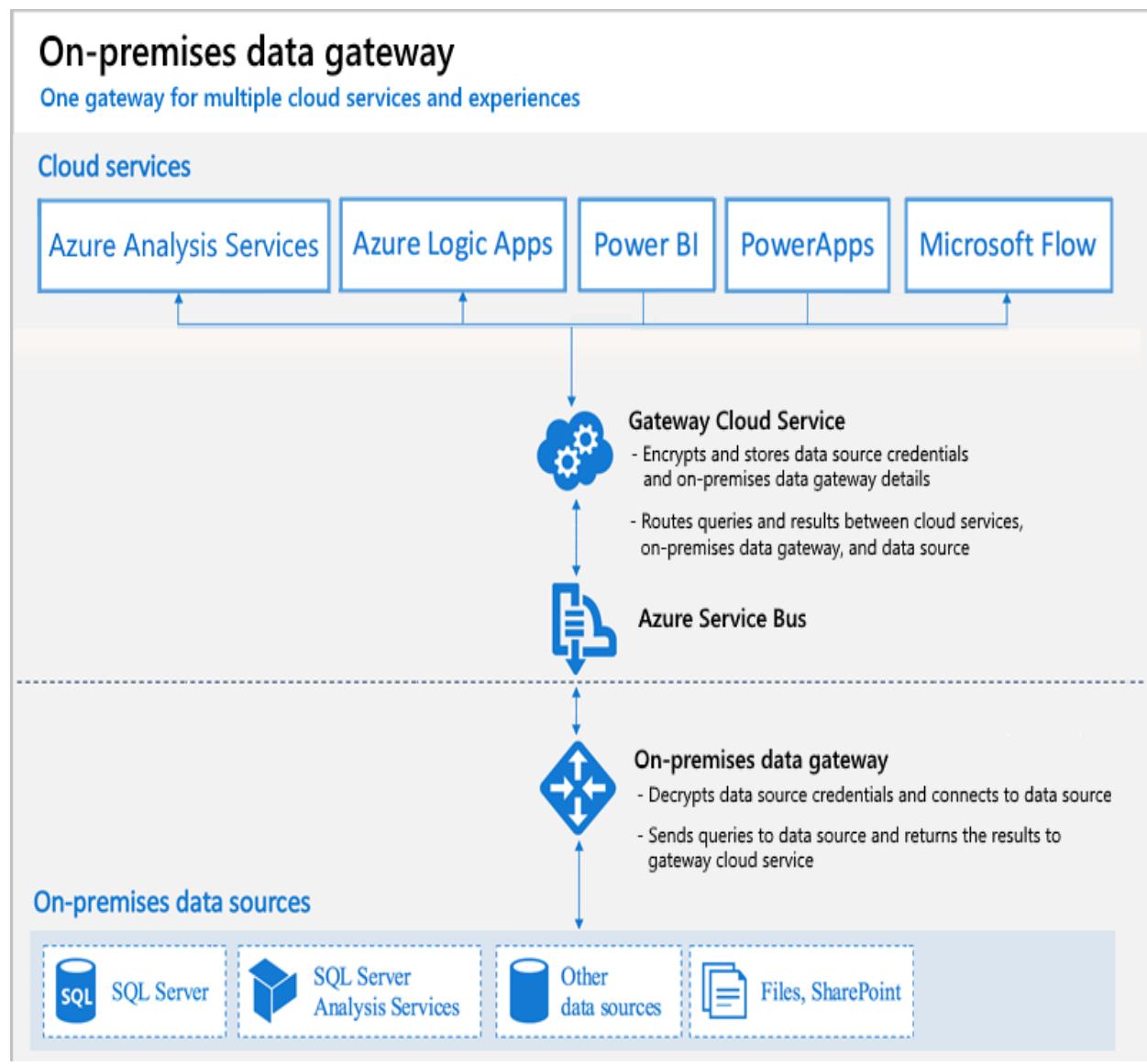
## Using a gateway

There are four main steps for using a gateway

- Install the gateway on a local computer, using the appropriate mode.
- Add users to the gateway and data sources, so they can access on-premises data sources.
- Connect to data sources, so they can be used in reports and dashboards.
- Refresh on-premises data, so Power BI reports are up to date.

## How gateways work

The gateway you install runs as a Windows service, On-premises data gateway. This local service is registered with the Gateway Cloud Service through Azure Service Bus. The following diagram shows the flow between on-premises data and the cloud services that use the gateway.



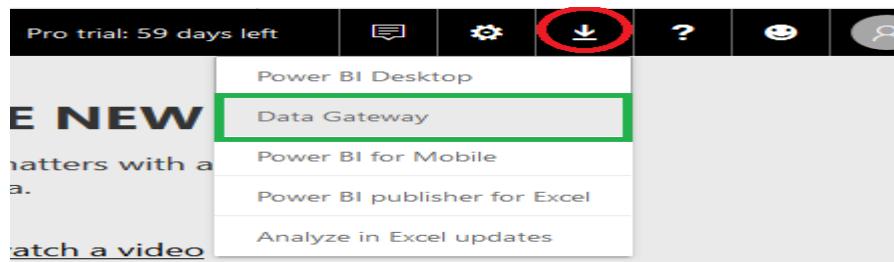
## Queries and data flow

- A query is created by the cloud service with the encrypted credentials for the on-premises data source. It's then sent to a queue for the gateway to process.
- The gateway cloud service analyzes the query and pushes the request to the Azure Service Bus.
- The on-premises data gateway polls the Azure Service Bus for pending requests.
- The gateway gets the query, decrypts the credentials, and connects to the data sources with those credentials.
- The gateway sends the query to the data source for execution.
- The results are sent from the data source, back to the gateway, and then onto the cloud service and your server.

## Downloading and Installing a data gateway for Power BI

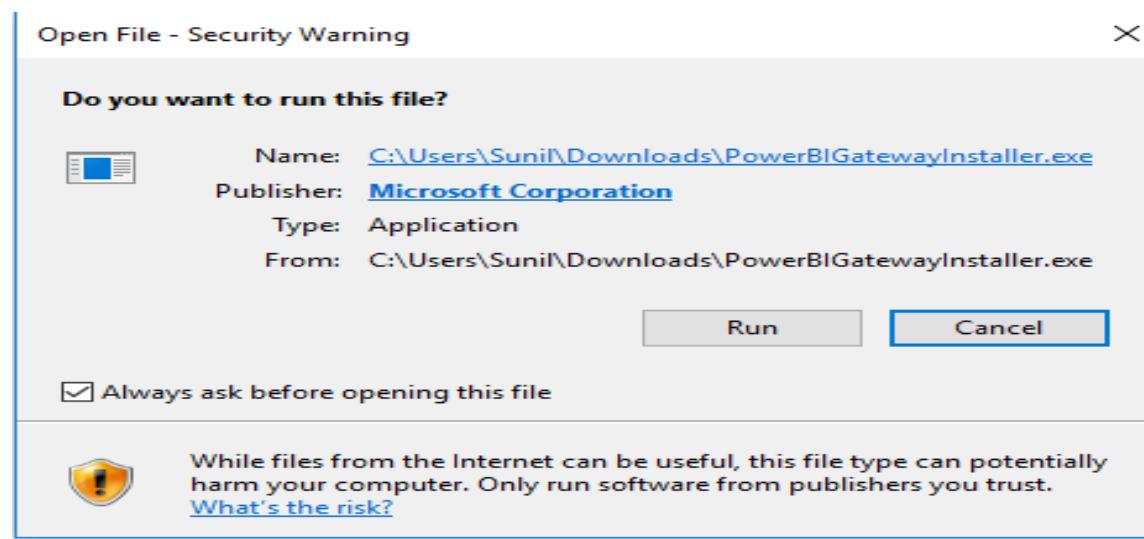
### Download Data Gateway

To install a gateway, you need to sign into Power BI Service and click on the Download button in the top-left corner of the screen, then click Data Gateway.



### Installing a Data Gateway

Double click on setup file → Run



Now you are seeing Gateway Usage and some recommendations. Select Next

 **On-premises data gateway installer**

Start your on-premises data gateway installation.

A gateway acts as a bridge between on-premises data (not in the cloud), and Power BI, PowerApps, Logic Apps, and Microsoft Flow.

Gateways should be installed on a computer that is always on.

Performance may be slower on a wireless network.

[Learn more](#)

**Next** **Cancel**

Select On-premises data gateway (recommended) → Next.

 **On-premises data gateway installer**

Choose the type of gateway you need.

On-premises data gateway (recommended)

- Can be shared and reused by multiple users
- Can be used by Power BI, PowerApps, Logic Apps, and Microsoft Flow
- Supports schedule refresh and live query for Power BI

[Learn more](#)

On-premises data gateway (personal mode)

- Can only be used by you
- Can only be used in Power BI
- Only schedule refresh is supported

[Learn more](#)

**Next** **Cancel**

## Types of gateways

The Power BI Gateway can be installed in two modes

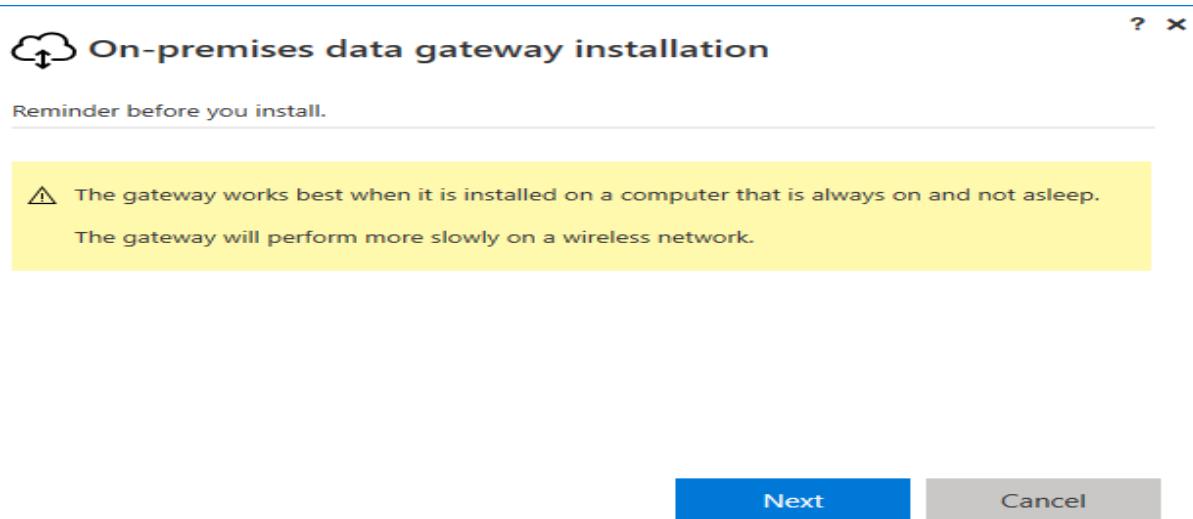
### On-premises data gateway

This gateway can be used by multiple users that have access to the server onto which you install the gateway. It can be used for both scheduling refresh and live queries in Power BI. You can also use it for PowerApps, Logic Apps, and Microsoft Flow. This gateway is well-suited to more complex scenarios with multiple people accessing multiple data sources.

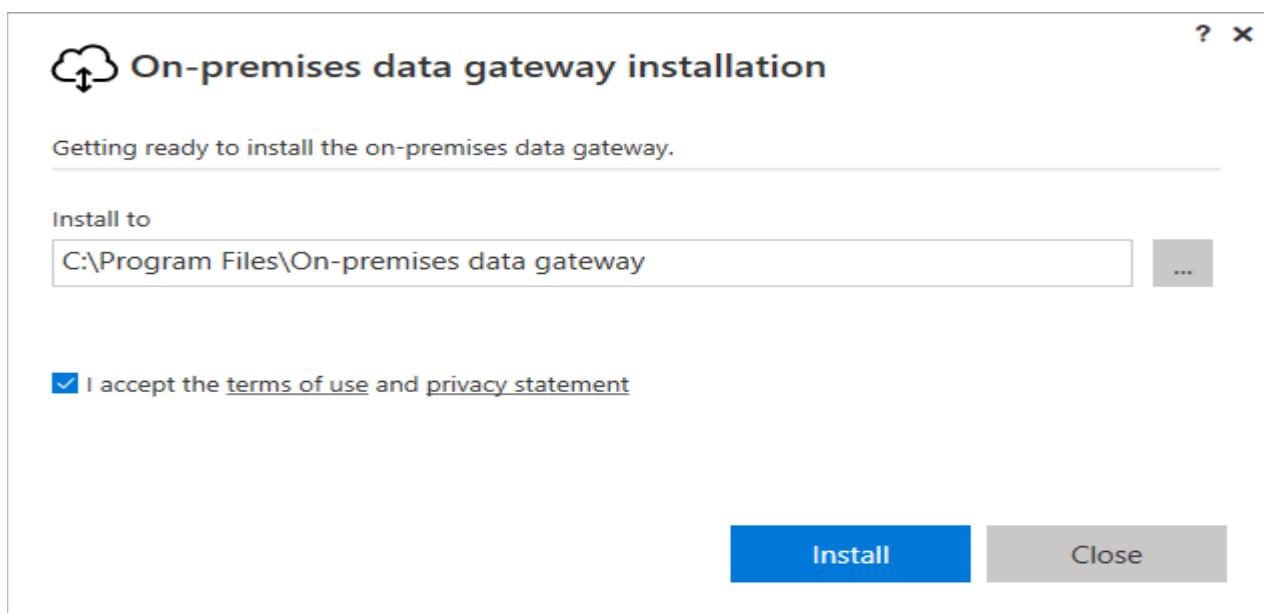
## On-premises data gateway (personal mode)

Only you can use this, and you can use it only for scheduling refresh in Power BI. The Live Connection connectivity mode, PowerApps, Logic Apps, Microsoft Flow are not supported. This gateway is well-suited to scenarios where you're the only person who creates reports, and you don't need to share the data sources with others.

In general, a data gateway should be installed on a machine that is always on and connected to the Internet, because a gateway cannot access on-premises data sources from a machine that is powered off. You can install up to one gateway in each mode on the same computer, and you can manage multiple gateways from the same interface on Power BI Service. Click Next.

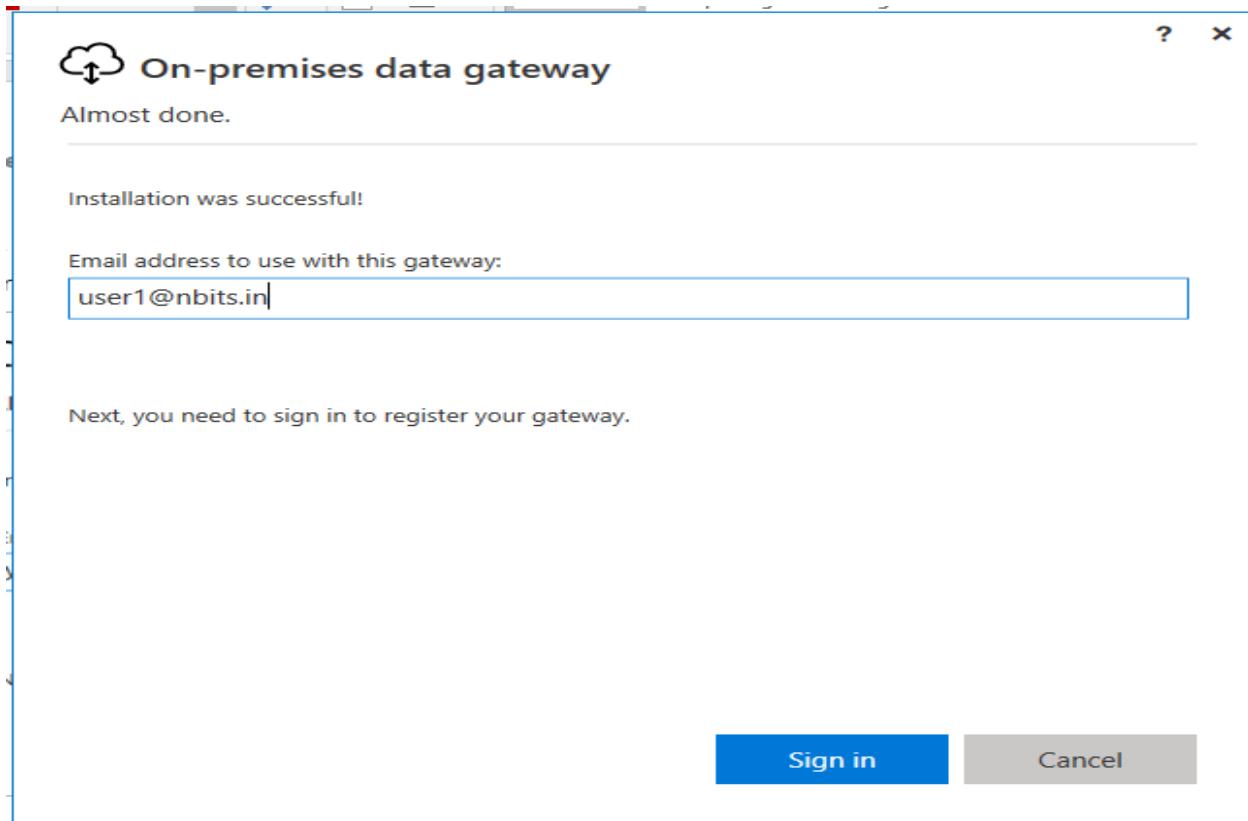


Keep the default install path, and accept the terms → Install.



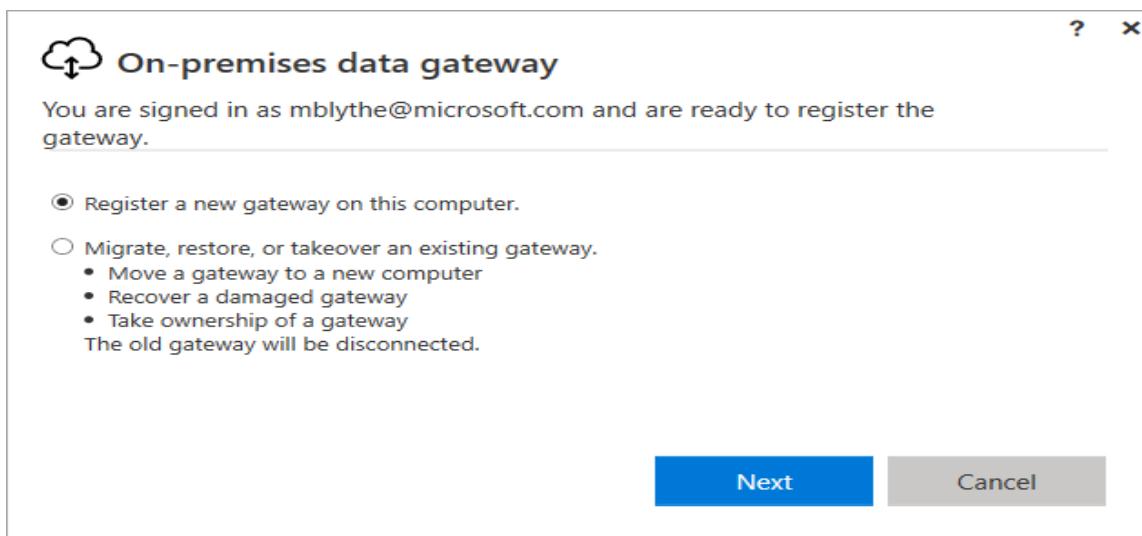
Click on Yes to make changes to the system.

Enter the account you use to sign in to Power BI → Select Sign in.



The gateway is associated with your Power BI account, and you manage gateways from within the Power BI service. You're now signed in to your account.

Select Register a new gateway on this computer → Next.



Enter a name for the gateway (must be unique across the tenant) and a recovery key. You need this key if you ever want to recover or move your gateway. Select Configure.

The screenshot shows the 'On-premises data gateway' configuration window. At the top, it says 'You are signed in as user1@nbts.in and are ready to register the gateway.' Below this, there are fields for 'New on-premises data gateway name' (containing 'ABTrainings Test'), 'Add to an existing gateway cluster' (unchecked), 'Recovery key (8 character minimum)' (containing '\*\*\*\*\*'), and 'Confirm recovery key' (containing '\*\*\*\*\*'). A note below the recovery key field states: '(1) This key is needed to restore the gateway and can't be changed. Record it in a safe place.' There is also a link 'Learn more about gateway clusters' and a note about connecting to cloud services in the 'West India' region. At the bottom are 'Configure' and 'Cancel' buttons.

Review the information in the final window. Notice that the gateway is available for Power BI. Click on Close.

The screenshot shows the 'On-premises data gateway' status window. On the left is a sidebar with 'Status', 'Service Settings' (selected), 'Diagnostics', 'Network', and 'Connectors'. The main area displays the following status information:

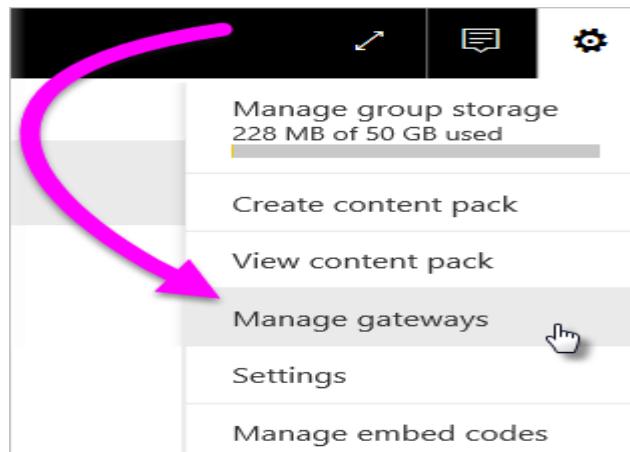
- Status:** The gateway ABTrainings Test is online and ready to be used.
- Service Settings:** Gateway version number: 14.16.6830.1 (September 2018). Includes a checkbox for sending usage information to Microsoft, which is checked, and a link to the privacy statement.
- Logic Apps, Azure Analysis Services:** West India. Status: Create a gateway in Azure.
- PowerApps, Microsoft Flow:** Default environment. Status: Ready.
- Power BI:** Default environment. Status: Ready.

At the bottom right is a 'Close' button.

Now you've successfully installed a gateway.

### Manage a Power BI gateway

You can manage a gateway through the Manage gateways area of the Power BI service.



### Manage data sources

Power BI supports many on-premises data sources, and each has its own requirements. A gateway can be used for a single data source or multiple data sources. For this example, we'll show you how to add SQL Server as a data source, but the steps are similar for other data sources.

#### Adding data sources to a data gateway

Even though we have a gateway installed, to schedule refresh, we need to add all data sources first. Once you have a gateway installed, you can add data sources to it in Power BI service. For that, In the upper-right corner of the Power BI service, select the gear icon → Then Select Manage gateways.

Select a gateway → Add data source

A screenshot of the Power BI service interface. On the left, there is a sidebar with 'ADD DATA SOURCE' (highlighted with a green box) and 'GATEWAY CLUSTERS'. Under 'GATEWAY CLUSTERS', there is a list with 'ABTrainings Test' (also highlighted with a green box). At the bottom of the sidebar is a 'Test all connections' button. On the right, a modal window titled 'ABTrainings Test' shows the status 'Connected' with a green checkmark. Below the status are 'ADD DATA SOURCE' (highlighted with a green box) and 'REMOVE' buttons.

Select the Data Source Type.

The screenshot shows the 'Add Data Source' dialog in Power BI. On the left, under 'GATEWAY CLUSTERS', there is a section for 'ABTrainings Test' and a button for 'New data source'. A 'Test all connections' button is also present. The main area is titled 'Data Source Settings' and contains a 'Data Source Name' field with 'PRODUCT DATABASE' and a 'Data Source Type' dropdown menu. The 'SQL Server' option is selected and highlighted with a yellow border. Other options listed in the dropdown include Tibco Rehana, Impala, MySQL, ODBC, OData, OleDb, Oracle, PostgreSQL, SAP Business Warehouse Message Server, SAP Business Warehouse Server, SAP HANA, and another 'SQL Server' option at the bottom, which is highlighted with a blue bar.

Enter information for the data source. For this example, it's Server, Database, and other information. Once you choose the SQL Server data source type, you will need to specify the server and database name, authentication method, username, and password. In the advanced settings, you have an option to use SSO via Kerberos for DirectQuery queries and choose a privacy level setting for this data source. In our case, we can specify just the server and database name, authentication method and credentials, leaving the advanced settings as-is. Note that you cannot leave the database name blank, even though you can connect to a server without specifying a database in Power BI Desktop. Select Add.

Data Source Settings      Users

Data Source Name  
PRODUCTDATABASE

Data Source Type  
SQL Server

Server  
DESKTOP-HFKOACI\ABTRAININGS

Database  
PRODUCTDATABASE

Authentication Method  
Windows

The credentials are encrypted using the key stored on-premises on the gateway server. [Learn more](#)

Username  
DESKTOP-HFKOACI\Sunil

Password  
\*\*\*\*\*

> Advanced settings

Add      Discard

You see **Connection Successful** if the process succeeds as shown in below image.

Data Source Settings      Users

✓ Connection Successful

① Next Step: Go to the [Users](#) tab above and add users to this Data Source

Data Source Name  
PRODUCTDATABASE

You can now use this data source to include data from SQL Server in your Power BI dashboards and reports.

## Manage users and administrators

### Add and remove administrators

If needed, you can enable others to administer the gateway in the Administrators Tab. Note that this does not automatically grant access to each data source in the gateway, you will need to add **Users** to each data source or he himself can add to that data source as he is now administrator for that data gateway. If you are the administrator for the data gateway, then you can add new Data Sources to data gateway and can perform refresh and scheduled refresh on the data sources.

On the Administrators tab for the gateway, add and remove users that can administer the gateway.

The screenshot shows the 'Administrators' tab selected in the 'Gateway Cluster Settings' section of the Power BI service. On the left, there's a sidebar with 'ADD DATA SOURCE' and 'GATEWAY CLUSTERS' sections, where 'test-gateway-docs' is selected. A 'Test all connections' button is visible. The main area displays a list of users who can administer the gateway, with 'Sunil A' currently listed. An input field 'Enter email addresses' and a yellow 'Add' button are present. Below the list, two users are shown: 'Analytics Benchmark' (unchecked) and 'AB Trainings' (checked). A 'Remove' button is located at the bottom right of the list area.

### Add users to a data source

After you add a data source to a gateway, you give users access to the specific data source (not the entire gateway).

In the upper-right corner of the Power BI service, select the gear icon → Manage gateways.

Select the data source where you want add users. Select Users, and enter a user from your organization who you want to grant access to the selected data source. In the following screen, you can see that I'm adding "Sunil A". Select Add, and the added member shows up in the box.

The screenshot shows the 'Data Source Settings' page for a data source named 'PRODUCTDATABASE'. On the left, under 'GATEWAY CLUSTERS', there is a section for 'test-gateway-docs' which contains 'Excel F' and a 'Test all connections' button. The main area is titled 'Data Source Settings' and has a 'Users' tab selected. Below the tab, it says 'People who can publish reports that use this data source'. A search bar contains 'Sunil A' and 'Enter email addresses'. There is a yellow 'Add' button. A list box shows two users: 'Analytics Benchmark' (unchecked) and 'AB Trainings' (checked). At the bottom of the list box is a 'Remove' button.

Remember that you need to add users to each data source to which you want to grant access. Each data source has a separate list of users, and you must add users to each data source separately.

### Remove users from a data source

On the Users tab for the data source, you can remove users that use this data source.

As shown in the above image select the user to remove and press Remove Button that is there in the bottom.

### Remove a gateway

You can remove a gateway if you're no longer using it. But be aware that removing a gateway deletes all the data sources connections under it, and if the data source is extract then you are unable to refresh or schedule refresh. If your data source is live you cannot see the data in visuals. This in turn breaks any dashboards and reports that rely on those data sources.

In the upper-right corner of the Power BI service, select the gear icon → Manage gateways.

Select the gateway → Open Menu by selecting the ... beside the gateway name → Remove

The screenshot shows the 'Manage Gateways' interface. On the left, under 'GATEWAY CLUSTERS', there is a section for 'test-gateway-docs'. The main area shows a card for 'test-gateway-docs' with a status of 'Connected'. Below the card is a 'REMOVE' button, which is highlighted with a red box. To the right of the card is a 'Test all connections' button.

## Collaborate in your Power BI using App Workspaces

### About App Workspaces

App workspaces are where teams collaborate on content. Power BI content is made up of dashboards, reports, workbooks, and datasets. And each piece of content is associated with a workspace. **My workspace** is where you create and manage content you own. Shared with me is where read-only content that colleagues have shared with you is stored.

Workspaces are areas where groups of users can collaborate with datasets, reports, and dashboards.

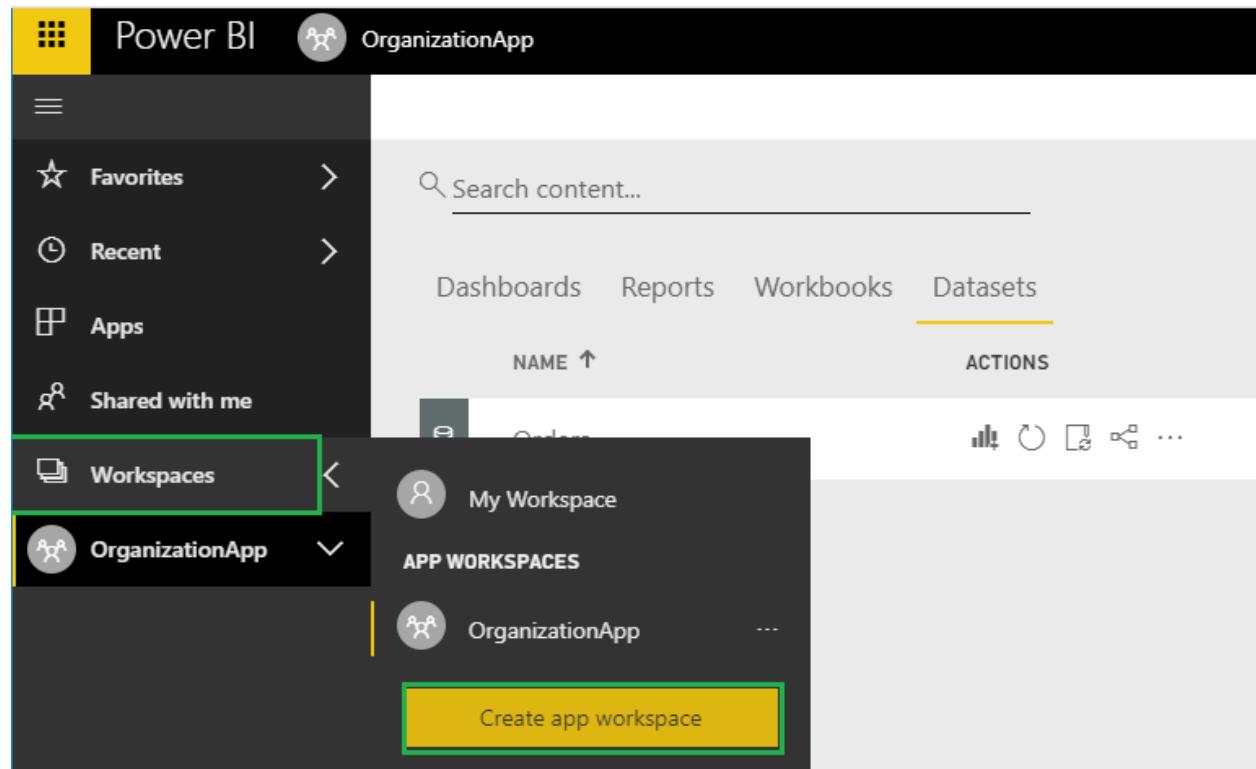
When you open Power BI for the first time, you'll have one top-level workspace named **My Workspace**. You can create and add new workspaces, and those will be organized as **APP WORKSPACES**.

You can create a workspace if you have a pro license of the Power BI service. This is the main way that your BI developers will be able to co-develop the same sets of data and reports. Typically, you'll create a workspace for each department in your company for the teams to store their items and data.

As the creator of a group, you are automatically an admin. As admin, you can add and delete members, and make other members admins. All admins can create, update, and delete the dashboards, reports, and other content of the group.

### Create a App Workspace and add members

To create one, simply expand the Workspaces section in the left navigation menu and click Create App Workspace.



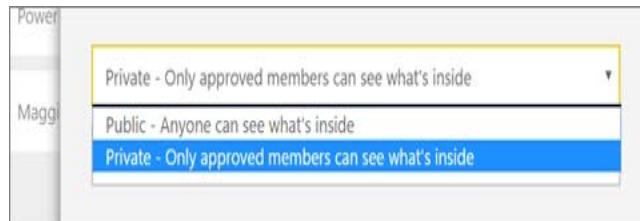
Give the workspace a name. If the corresponding Workspace ID isn't available, edit it to come up with a unique ID.

This will be the name of the app, too.

## Create an app workspace

The screenshot shows a user interface for creating an app workspace. At the top, there's a yellow banner with the text "PREVIEW IMPROVED WORKSPACES" and "Preview the new way to create workspaces." A "Try now" button and a close "X" icon are also present. Below the banner, there are two input fields: one for "Name your workspace" containing "AnalyticsBenchmark" and another for "Workspace ID" containing "analyticsbenchmark". Both fields have a placeholder "Available" below them.

You have a few options to set. If you choose Public, anyone in your organization can see what's in the workspace. Private, on the other hand, means only members of the workspace can see its contents.



You can also choose if members can edit or have view-only access.



Only add people to the app workspace so they can edit the content. In Privacy Select “Members can edit Power BI content” and Add “Workspace members”. If they're only going to view the content, don't add them to the workspace. You can include them when you publish the app.

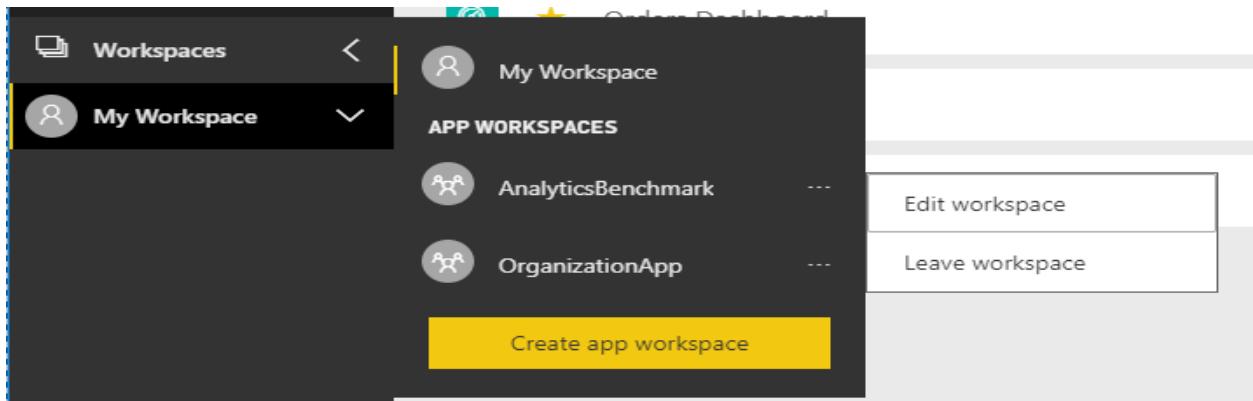
Add email addresses of people you want to have access to the workspace, and select Add. You can't add group aliases, just individuals.

Decide whether each person is a member or an admin.

Workspace members			
Enter email addresses			
Add			
user1@nbits.in	Admin	▼	trash bin
user2@nbits.in	Member	▼	trash bin

Admins can edit the workspace itself, including adding other members. Members can edit the content in the workspace, unless they have view-only access. Both admins and members can publish the app. Select Save.

Power BI creates the workspace and opens it. It appears in the list of workspaces you're a member of. Because you're an admin, you can select the ellipsis (...) to go back and make changes to it, adding new members or changing their permissions like Member to Admin or vice versa.

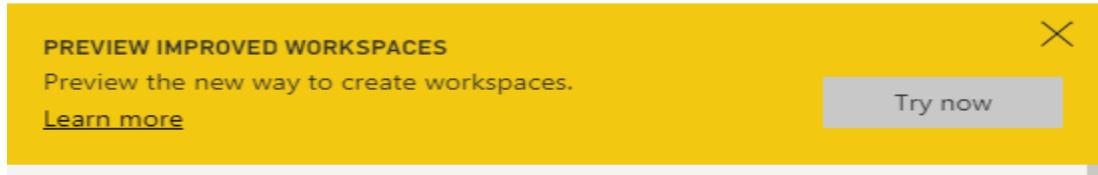


### Create one of the new app workspaces using Improved Workspaces

Start by creating the app workspace. Select Workspaces → Create app workspace.

In Preview improved workspaces, select Try now.

## Create an app workspace



Give the workspace a name. If the name isn't available, edit it to come up with a unique ID. The app will have the same name as the workspace.

### Create an app workspace

Image

Name your workspace

Available

Description

Testing App Work-space New|

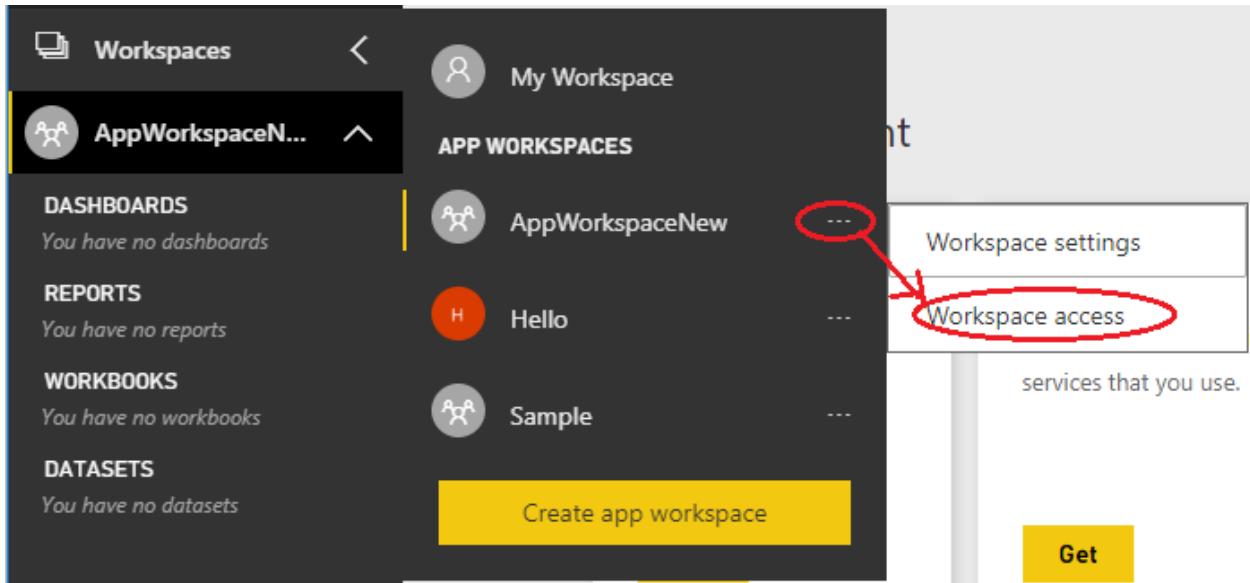
Save Cancel

Select Save.

Once save you can see a Workspace with the Name **AppWorkspaceNew**.

The screenshot shows a dark-themed interface for managing workspaces. On the left, there's a sidebar with sections for DASHBOARDS, REPORTS, WORKBOOKS, and DATASETS, each stating 'You have no [type]'. The main area is titled 'APP WORKSPACES' and lists three workspaces: 'AppWorkspaceNew' (highlighted with a red box), 'Hello' (with a red 'H' icon), and 'Sample'. At the bottom right of the workspace list is a yellow button labeled 'Create app workspace'.

As you created this Workspace you are the Admin for this Workspace. You can Add admins, members or contributors to this Workspace with different roles by selecting Workspace access.



The new workspaces offer three roles → admins, members, and contributors.

A screenshot of the 'Access' section for 'App Workspace New'. It shows a list of email addresses: 'Analytics Benchmark' and 'Enter email addresses'. Below is a dropdown menu with four options: 'Member' (highlighted in yellow), 'Member' (highlighted in blue), 'Contributor', and 'Admin'.

Member
Member
Contributor
Admin

#### Admins can

- Update and delete the workspace.
- Add/remove people, including other admins.
- Do everything members can do.

### **Members can**

- Add members or others with lower or equal permissions.
- Publish and update an app.
- Share an item or share an app.
- Allow others to re-share items.
- Can't delete or remove the Members of the Group or App Workspace.
- Do everything contributors can do.

### **Contributors can**

- Publish reports to the workspace.
- Create, edit, and delete content in the workspace.
- Can't modify (Add or Delete) the members of the group.
- Can't share an item or share an app.

Update and Delete App Workspace

Add other users to App workspace

Add content, delete content and create new content in App workspace

Publish and Update Appto share content with Business Users.

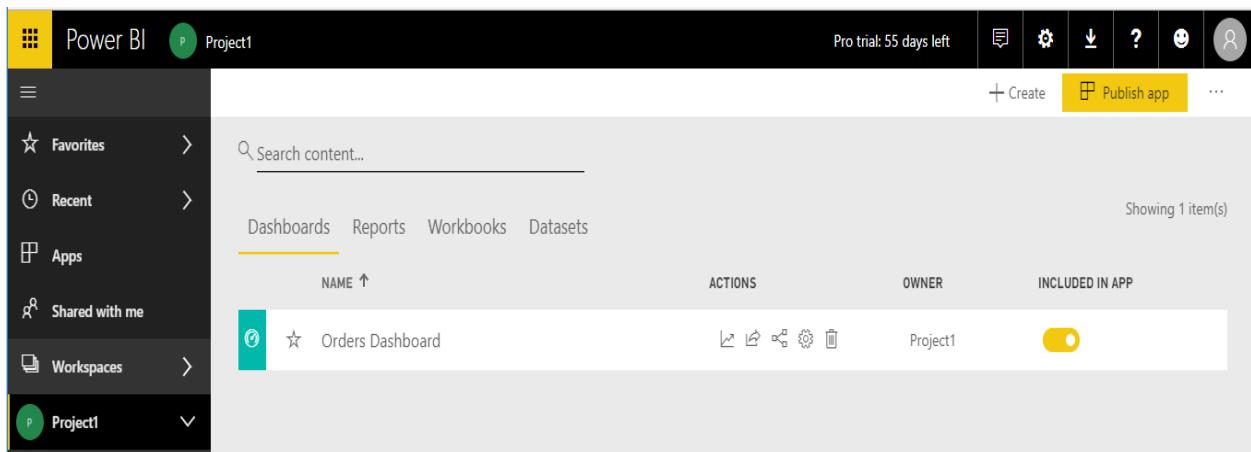
## App

An App is a collection of Power BI Content, such as Dashboard, Reports, Workbooks, and Datasets packaged together.

### Publish an App

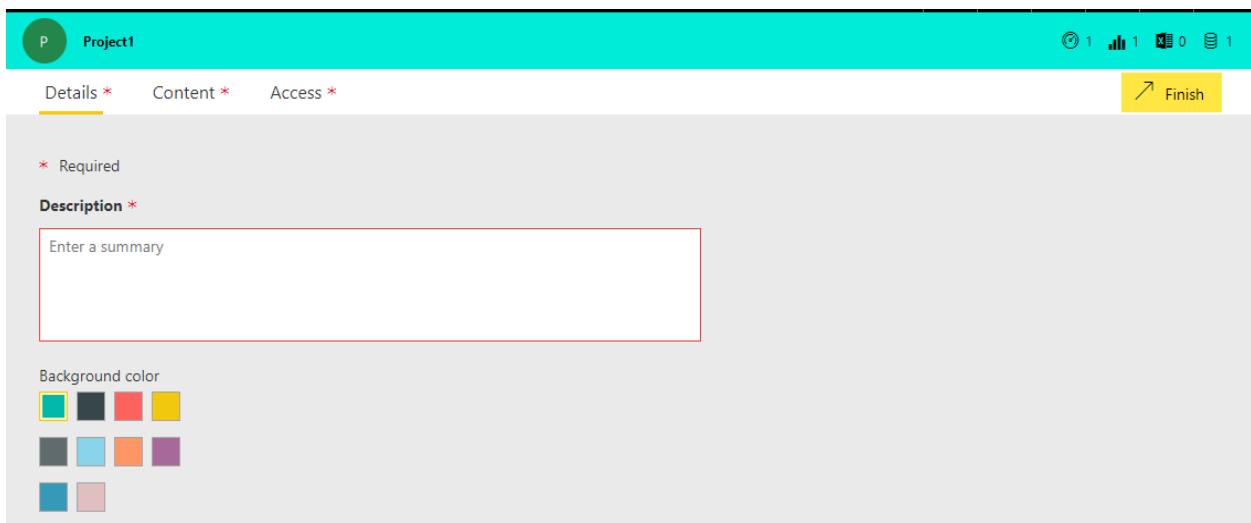
When you are ready to share your Reports and Dashboards with users in your Organization, you can publish an App. Only members who have edit access to an App Workspace Can Publish Apps. Currently, there can only be one App per App Workspace. In the New App Workspace Creation, who have permissions as Admin and Member can create a App but Contributor cannot have permissions to create App.

To publish an App in Power BI service, you need to go to the App workspace view first. For this, you need to click Workspaces in the navigation pane on the left and click the App workspace from which you want to publish an App as shown in image.



The screenshot shows the Power BI service interface. At the top, there's a navigation bar with 'Power BI' and 'Project1'. On the right side of the bar are icons for 'Create', 'Publish app' (which is highlighted in yellow), and other settings. Below the bar is a search bar labeled 'Search content..'. Underneath the search bar are tabs for 'Dashboards', 'Reports', 'Workbooks', and 'Datasets', with 'Dashboards' being the active tab. A message 'Showing 1 item(s)' is displayed. A table below lists a single item: 'Orders Dashboard'. The table has columns for 'NAME' (with an upward arrow), 'ACTIONS', 'OWNER' (labeled 'Project1'), and 'INCLUDED IN APP' (with a toggle switch that is turned on). On the far left, there's a sidebar with links for 'Favorites', 'Recent', 'Apps', 'Shared with me', 'Workspaces' (which is currently selected and has a green background), and 'Project1'.

To publish an app, you need to click the Publish App button in the top-right corner of the window. You will then be taken to the Publish App page, where you will see three tabs→ Details, Content, and Access.



The screenshot shows the 'Publish App' page. At the top, there's a header with 'Project1' and various status icons. Below the header are three tabs: 'Details \*' (which is selected and highlighted in yellow), 'Content \*', and 'Access \*'. To the right of the tabs is a 'Finish' button. The main area contains several sections: a note about required fields ('\* Required'), a 'Description \*' section with a text input field containing 'Enter a summary', and a 'Background color' section with a grid of color swatches. The background of the page is light gray.

In the Details tab, you need to enter an App Description to help users understand its contents. You can also personalize your app by selecting a background color.

In the Content tab, you can see the Power BI content that will be published across three categories → Dashboards, Reports, and Datasets. Workbooks are listed in the Reports section. At this stage, you cannot exclude any items.

Below the app content, you can specify the app landing page, which is what users who go to your app will see first. You can choose either of the following below two options.

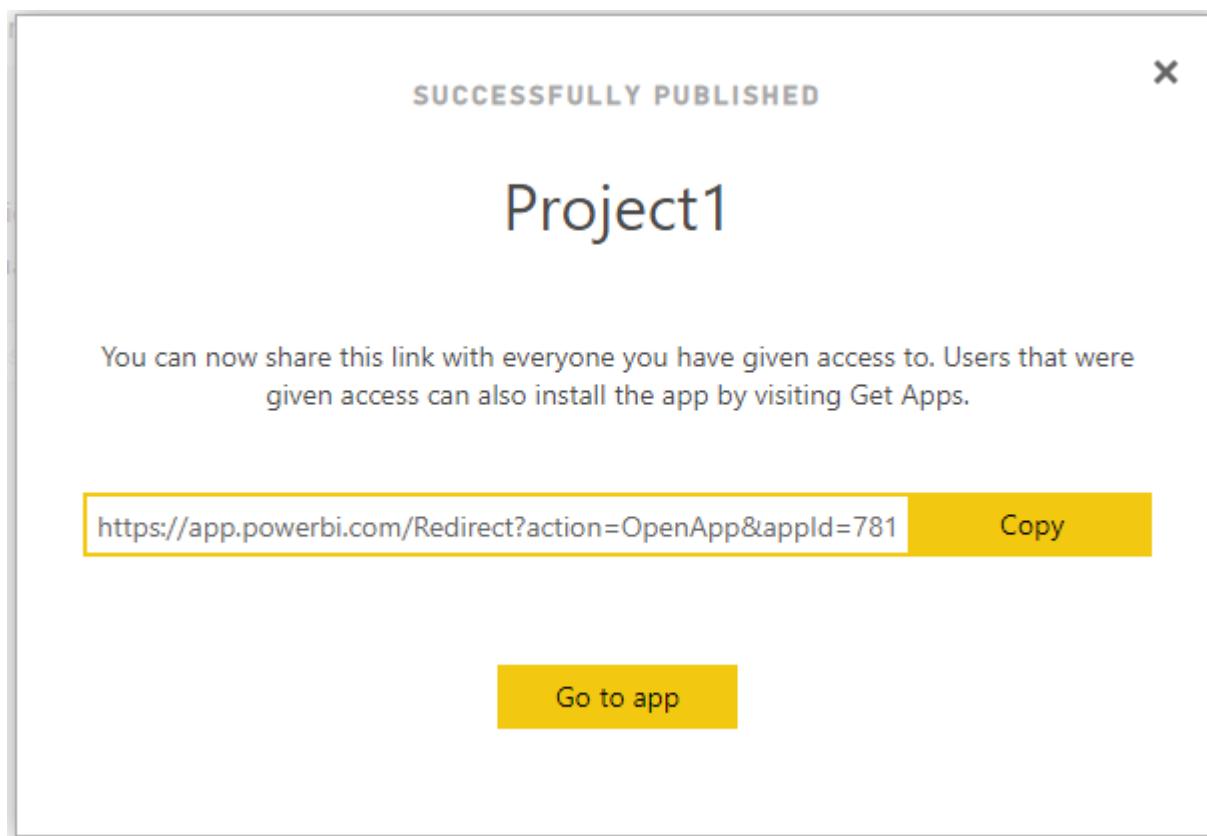
**Specific Content** This can be a Dashboard or a Report, which you can select from the drop-down list below. None Users will see the App Contents page instead of a specific item.

By default, Specific content is selected, though you need to select the landing item. In our example, we are going to select None.

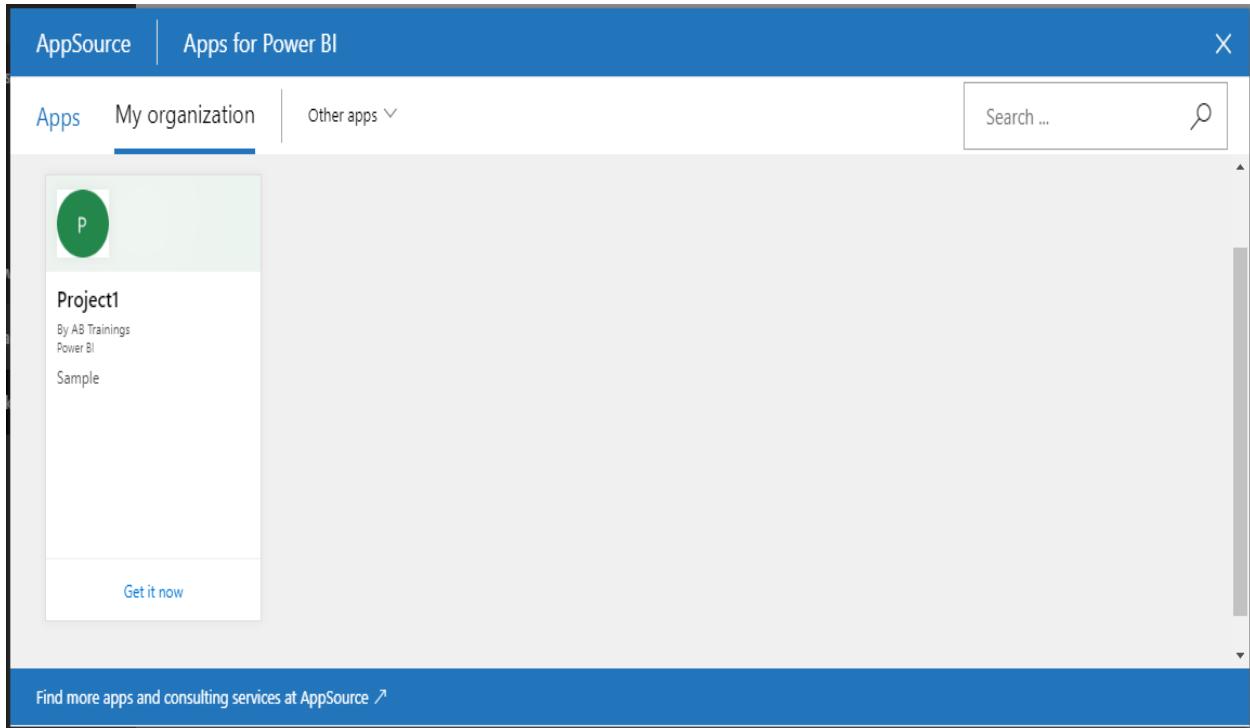
For your App, you can choose to publish it to the entire organization or for specific individuals or groups. The default selection is Specific Individuals or Group. If you choose to keep the setting, you will need to enter email addresses of individuals or groups below.

In our example, we can select Entire Organization. Once we click Finish in the top-right corner, we will see the Ready to Publish window, where we need to click Publish. After this, we will see the Successfully Published window.

In this window, you can copy the app link and share it with users to which you have given access.



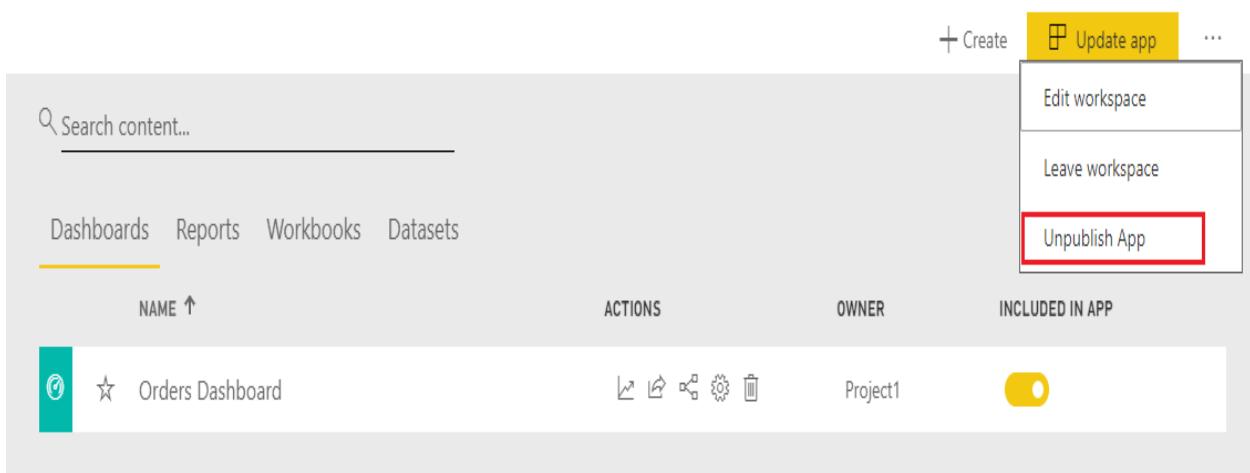
Alternatively, to get an app, a user can click Apps in the navigation pane in Power BI service, and then Get Apps, which will open the AppSource window, shown in Figure



In the AppSource window, they will need to click Get It Now next to the app they want to get. Note that a user does not need to be a member of the App workspace to get the App.

### Unpublishing an App

If you want to unpublish an app, you can do so by clicking the ellipsis in the top-right corner of the app workspace and clicking Unpublish App. You will need to confirm your action by clicking Unpublish in the Unpublishing an app window. Doing so will not delete the app workspace contents, instead, the App will be removed from Apps list of each user and become inaccessible.



## Update a published App

After you publish your app, you can make changes to it if you are an app workspace admin or a member with edit rights. For this, you need to go to the App workspace and make the changes you want, once you have made the changes, you need to go back to the app workspace list of contents and click Update App. You can also update the Details, Content, and Access settings that you configured when you created the app. Clicking Update App will open the Ready to update dialog box, where you will need to click Update to propagate the App changes.

When creating or updating an App, you have an option to exclude some dashboards or reports from the app. In the app workspace view, there is an Included in App switch next to each Dashboard, Report, and Workbook; datasets are automatically included in Apps. To exclude an item from the App, you need to click on the relevant Included in App switch. If you are excluding items from an app that has already been published, you will need to update the App.

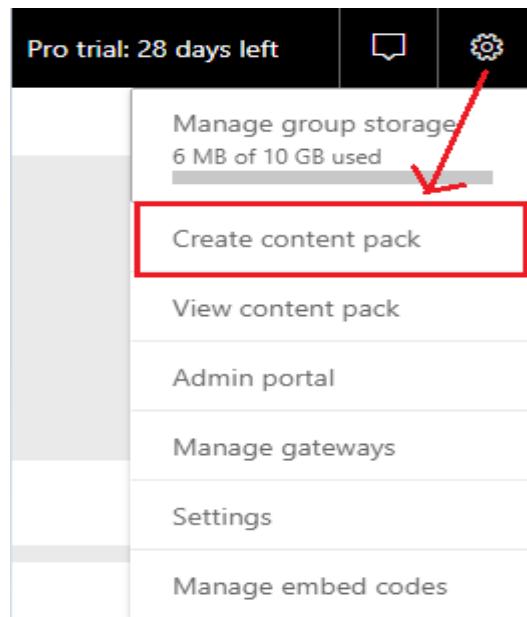
## Organizational Content Pack

Combining or Packing Power BI Content like Dashboards, Reports, Excel Workbooks, and Datasets is called Organizational Content Pack.

We publish or share this organizational content pack to the team or complete organization.

## Creating and Using Content Packs with Power BI

In order to create a content pack, the Dataset, Workbook, or Report must have already been deployed to the Power BI site, <http://app.powerbi.com>. Once the object is on the site, you will then click on the gears button in the upper right corner of the screen to open up the settings. Next, you select Create content pack as show below.



In the Create Content Pack window, enter the following information.

- Users Need access to content pack.
- Title of Content Pack
- Description for Content Pack
- Upload an Image for the Content Pack
- Select Content / Items to Publish

Choose who will have access to this content pack:

Specific groups  My entire organization

**Title**

Content Pack Orders

**Description**

Orders Content Pack



Upload an image or company logo

Image size: 45 KB or less, 4:3 aspect ratio, JPG or PNG format

Use default



Upload an image or company logo

Image size: 45 KB or less, 4:3 aspect ratio, JPG or PNG format

Use default

**Select items to publish**

**Dashboards**

Sample Report Demo.pbix

**Reports**

Report2

Sample Report Demo

**Datasets**

Sample Report Demo

The content pack will be available in your organization's content gallery. [Learn more](#)

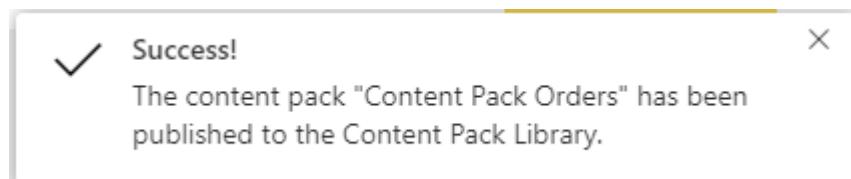
**Publish**

**Cancel**

If you have Excel workbooks, you see them under Reports, with an Excel icon. You can add them to the content pack, too.

Select Publish to add the content pack to the group's organizational content pack library.

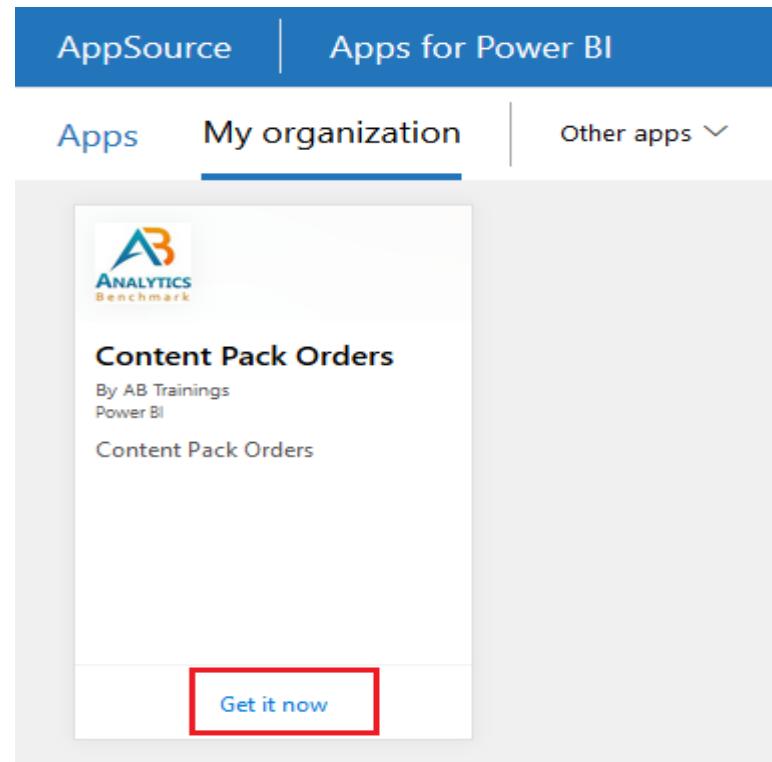
You see a success message when it publishes successfully.



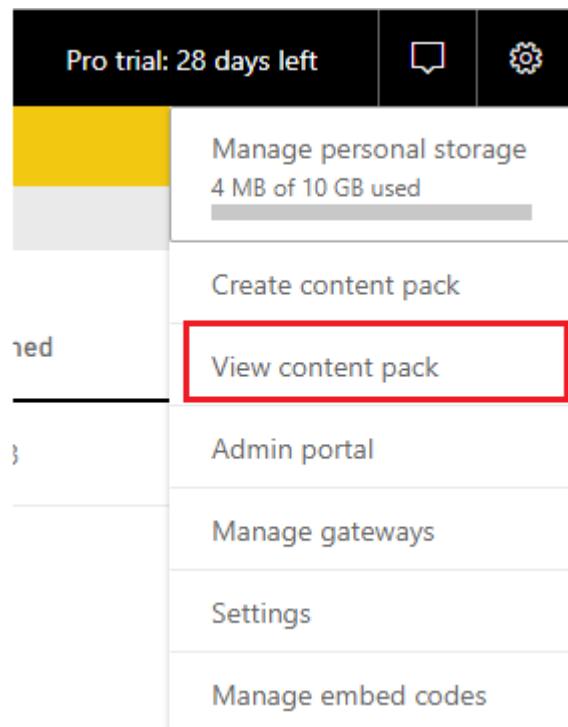
To View the content Pack. Select Get Data → Organizational Content Pack.

The screenshot shows the Power BI service interface. On the left, there is a sidebar with navigation options: Recent, Apps, Shared with me, Workspaces (with My Workspace selected), and Get Data. The main area is titled "Discover content" and contains two cards: "My organization" and "Services". Each card has a "Get" button. Below these cards, there is a section titled "More ways to create your own content" with links: Samples, Solution Templates, Partner Showcase, Organizational Content Packs (which is highlighted with a red arrow and a red box), and Service Content Packs. A red arrow also points from the "Get Data" button in the sidebar to the "Organizational Content Packs" link.

To view the content in Content Pack, Go to corresponding content pack and select Get it now



To Manage or Make any Changes to Content Pack, Settings → View Content Pack



When you select View content pack below window will open. You can go and Edit or Delete the content pack.

Name	Published To	Date published	Actions
Content Pack Sample	My organization	Oct 16, 2018	<a href="#">Edit</a>   <a href="#">Delete</a>

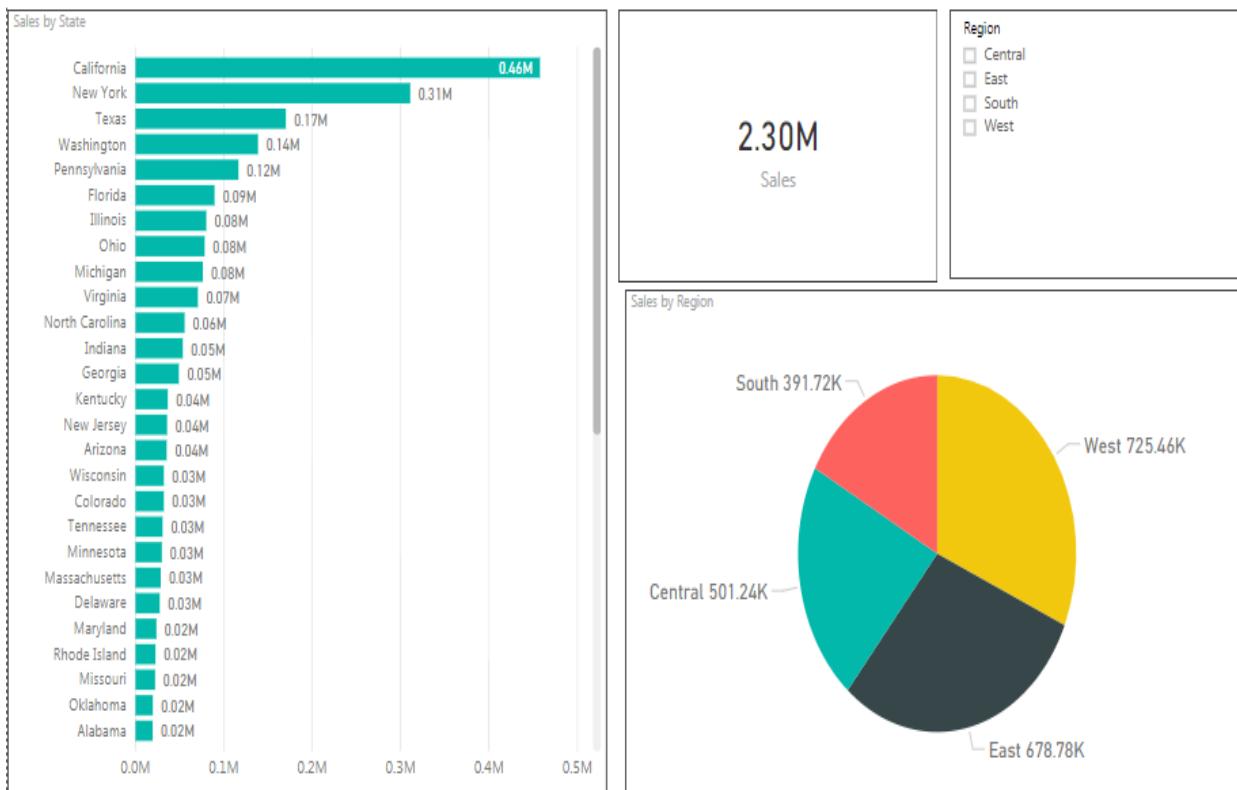
## Row Level Security

Enabling different roles and giving users access to different levels of data is called Row Level Security. Now we will see how to configure Row Level Security in Power BI Desktop.

Row Level Security enables you to apply security to roles and adds users to each role. An example is helpful when you want people from one branch, city, department, or store to be able to only see their part of the data and not the whole data set. Power BI applies that through a row level security configuration on the Power BI model itself. So regardless of what source you are importing your data from, you can apply row level security on it.

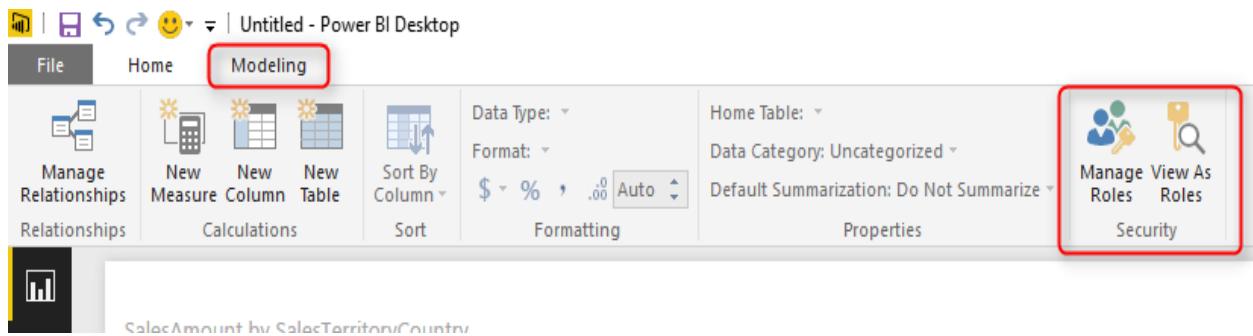
## Create Sample Report

Load order table into Power BI and create a Pie Chart for Region wise Sales, Total Sales in Card Visualization, Region in Slicer and Bar Chart for State Wise Sales.

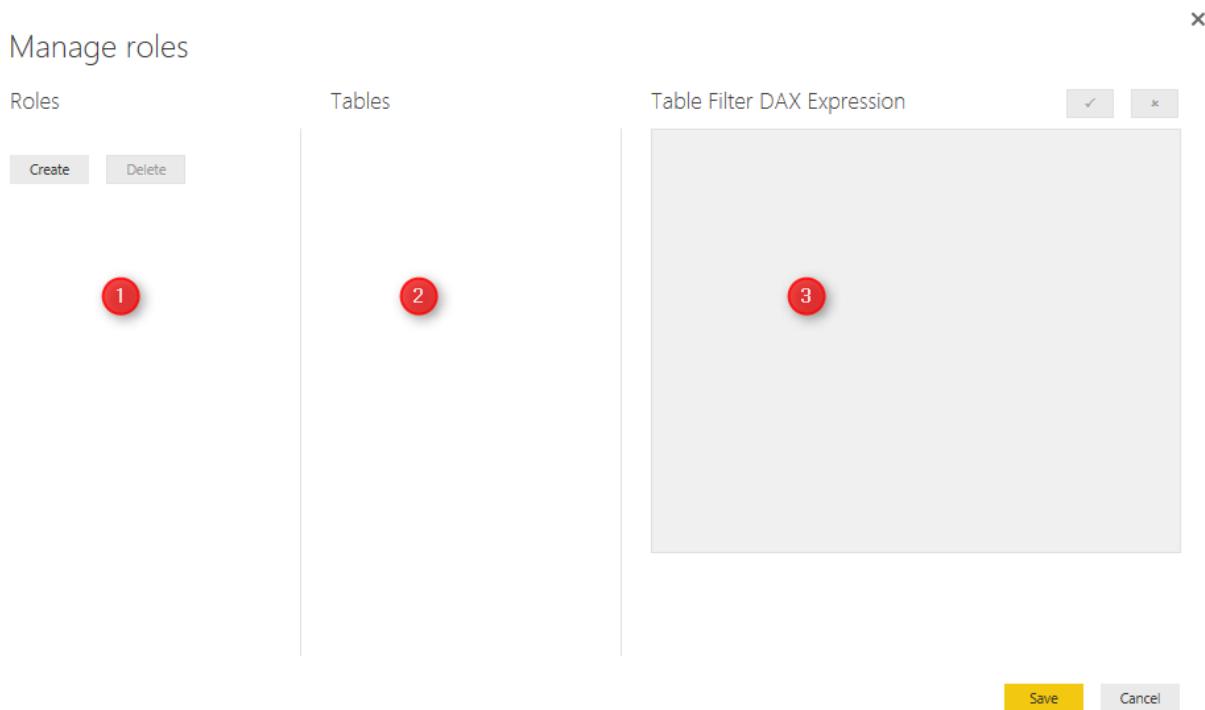


## Creating Roles

Now let's create roles for that. Our goal here is to build roles for sales manager of Central, East, South and West Regions. They should each only see their group or Region in the data set. For creating roles go to Modeling tab in Power BI Desktop. You will see a section named Security there as shown below.

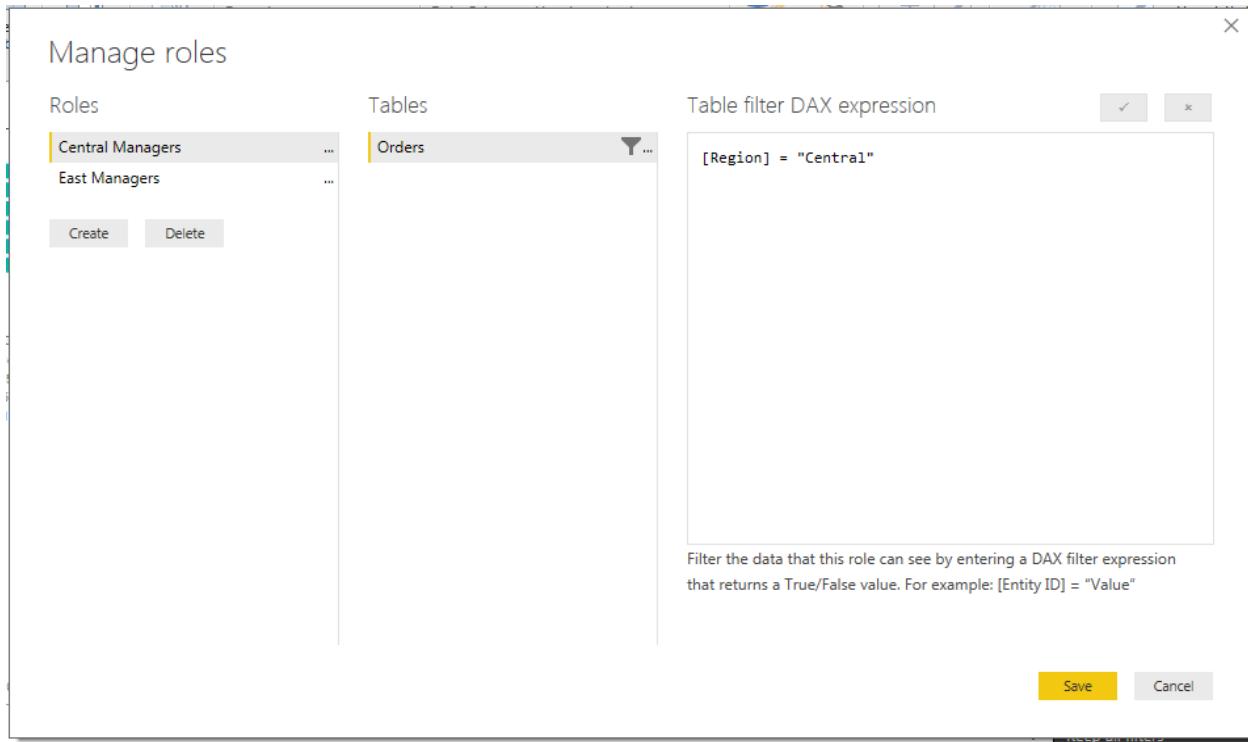


Click on Manage Roles to create a new role. You will see Manage Roles window which has three panes as below



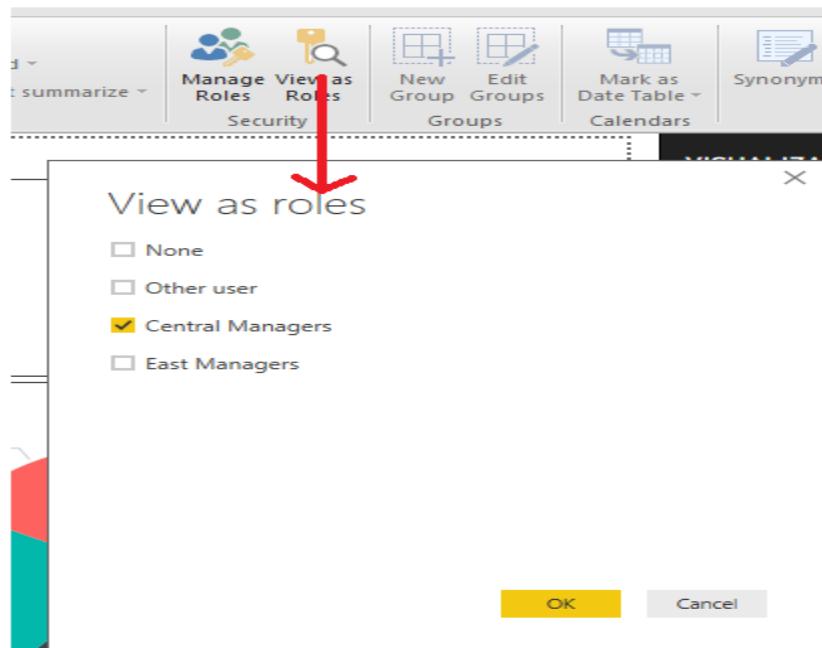
You can create or delete roles in numbered one pane, You can see tables in your model in numbered two pane (for this example you will see one table only, but not now, after creating the first role), and then you can write your DAX filtering expression in numbered three pane.

Now Create a Role, and name it as "Central Manager", you will see one table in the Tables section i.e., Orders Table. With a click on ellipsis button on table, you can create DAX filters for each column. From Orders create a filter for Region Column.

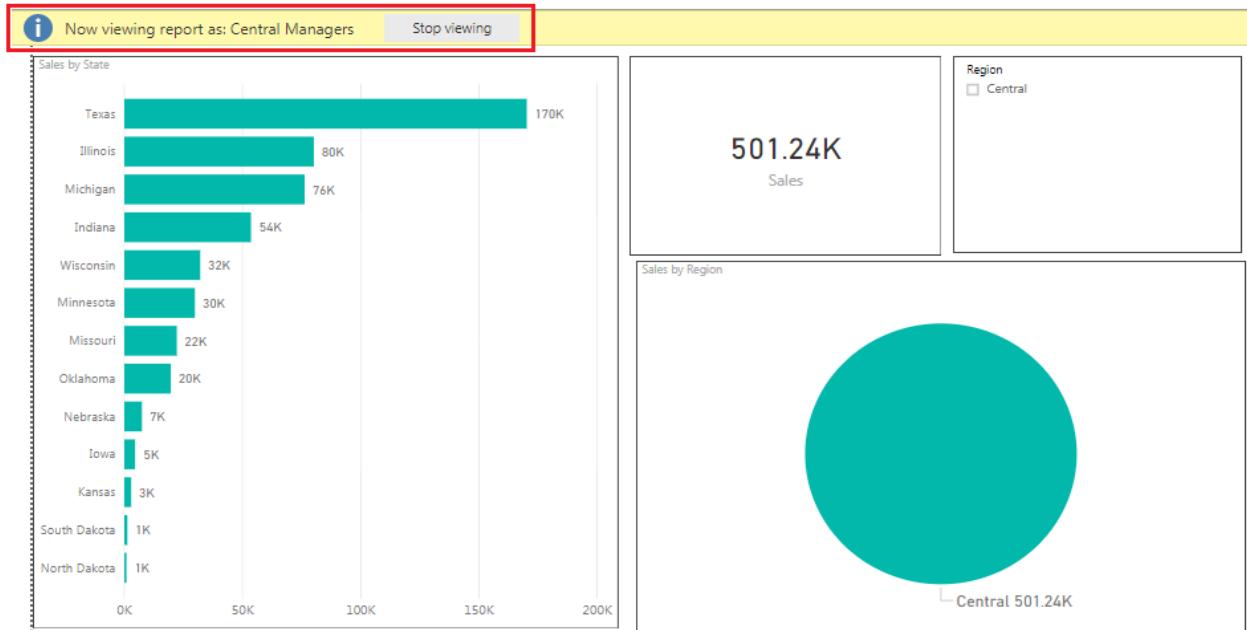


## Testing Roles in Desktop

We have created our two sample roles. Now let's test them here. We can test them in Power BI Desktop with **View as Roles** menu option. This option allows us to view the report exactly as the user with this role will see. We can even combine multiple roles to see a consolidated view of a person who has multiple roles by selecting Ctrl button to select multiple values or Roles. Go to Modeling tab, and choose **View as Role** option.



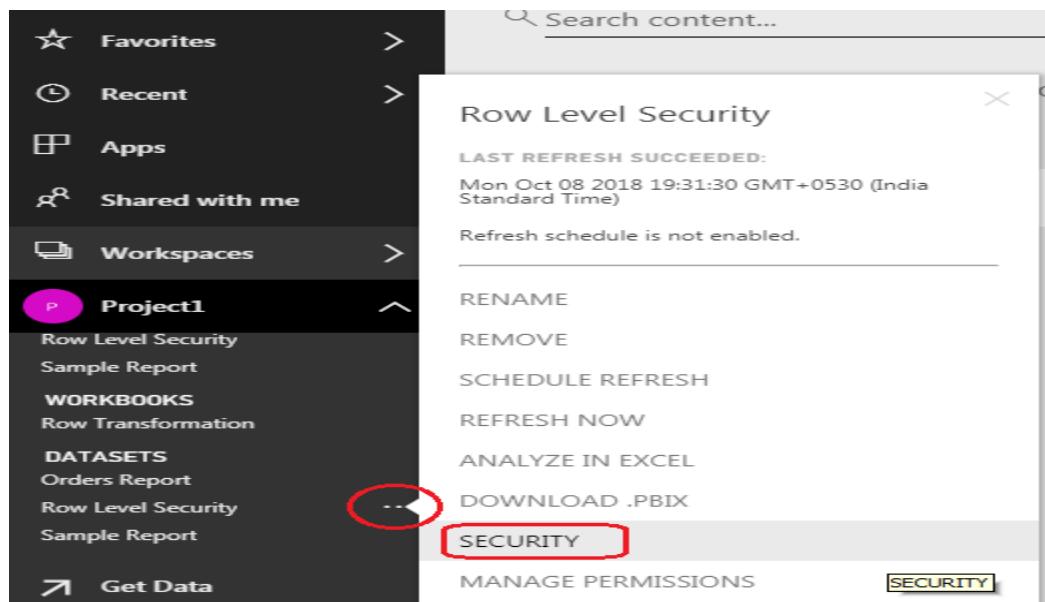
Choose Central Managers, and click on OK. You will see sales for Central Region only and showing only states of central region.



You can also see in the top of the report there is an information line highlighted showing that the view is Central Managers. If you click stop viewing, you will see the report as normal view (total view).

## Power BI Service Configuration

Roles should be assigned to Power BI users (or accounts in other words), and this part should be done in Power BI Service. Save and publish the report into Power BI. After publishing the report, click on Security for the data set.



Here you can see roles and assign them to Power BI accounts in your organization.

You can set each user to more than one role, and the user then will have a consolidated view of both roles. For example, a user with both roles for the Central and EastManagers will see data from All Central and East Regions.

## Row-Level Security

The screenshot shows the 'Central Managers (1)' role selected. The 'Members (1)' section displays a single member named 'Analytics Benchmark'. An 'Add' button is available to add more members. A red 'X' icon is present to remove the member.

### Test Roles in Power BI Service

You can also test each role here, just click on ellipsis button beside each role, and click on Test as Role.

## Row-Level Security

The screenshot shows the 'Central Managers (1)' role selected. The 'Members (1)' section displays a single member named 'Analytics Benchmark'. A red box highlights the 'Test as role' button next to the member name. An 'Add' button is available to add more members. A red 'X' icon is present to remove the member.

Test as Role will show you the report in view mode for that role. As you see the blue bar shows that the report showed the role of Central Managers. You can change it there if you like.

The screenshot shows the Power BI desktop interface. At the top, a blue header bar displays the text "Now viewing as: Central Managers" with a dropdown arrow. Below the header is a toolbar with icons for Back to Row-Level Security, File, View, Edit report, Explore, Refresh, Pin Live Page, Reset to default, View related, Favorite, and a message icon. On the left, a dark sidebar menu includes Home (preview), Favorites, Recent, Apps, Shared with me, Workspaces, Project1 (selected), Reports (Orders Report, Row Level Security, Sample Report), and Workbooks (Row Transformation). The main workspace contains two visualizations: a horizontal bar chart titled "Sales by State" showing sales for various US states, and a donut chart titled "Sales by Region" showing sales for the Central region. The bar chart data is as follows:

State	Sales
Texas	170K
Illinois	80K
Michigan	78K
Indiana	54K
Wisconsin	32K
Minnesota	30K
Missouri	22K
Oklahoma	20K
Nebraska	7K
Iowa	5K
Kansas	8K
South Dakota	1K
North Dakota	1K

The donut chart indicates a total sales value of 501.24K for the Central region.

With setting users for each role, now your role level security is ready to work. If the user login with their account, they will only see data for their roles.

## Dynamic Row Level Security

### Why Dynamic Row Level Security?

The most important question is why dynamic row level security? To answer this question, you need to think about the limitation of static row level security. Static row level security is simple to implement, however, if you have thousands of roles, then it would be a difficult to maintain. In this case Dynamic Row Level Security is the solution.

By dynamic row-level security, I mean the definition of security be beside the user account information in the data source. For example when User2 logs in to the system, based on data tables that show User2 is sales manager for specific Region he should be able to see only those Region data. This method is possible in Power BI using DAX `USERNAME()` or `USERPRINCIPALNAME()` function.

`USERNAME()` → DESKTOP-HFKOACI\Sunil

`USERPRINCIPALNAME()` → user1@nbits.in

## **Deployment Pipelines**

Power BI Deployment Pipelines is an Option / Tool to support BI Creators to ship their new content or updated content, from Development to Test, Test to Production Environments.

This is a Power BI Premium-only feature.

### **Advantages**

- Improve BI Creator Productivity
- Deliver Content Updates faster
- Reduce Errors & Manual Work

The tool is designed as a pipeline with three stages

### **Development**

This stage is used to design, build, and upload new content with fellow creators. This is the first stage in deployment pipelines.

### **Test**

After the content is uploaded and all changes are made in the development stage, the content can be moved to this stage for testing. Here are three examples of what can be done in the testing environment.

- Share content with testers and reviewers
- Load and run tests with larger volumes of data
- Test your app to see how it will look for your end users

### **Production**

After testing the content, use the production stage to share the final version of your content with business users across the organization.

## **Incremental Refresh**

In Power BI Datasets, Data in Tables Can be refreshed in Two Ways

### **1. Full Refresh / Full Load**

In Full Refresh / Full Load all the Tables Data in the Dataset will be Truncated and Reloaded again.

The default Configuration for Power BI Dataset is to wipe out / truncate the entire data in all the Tables and re-load it again.

If the Dataset is small, or the refresh process is not taking a longer time, then the full load is not a problem at all.

When Either the Dataset is big, or the Refresh Process of Dataset takes longer Time, better to Configure Incremental Refresh.

### **2. Incremental Refresh / Incremental Load**

In a Dataset Rather than Refreshing the Entire Data in the Tables, you can set it up to only add the rows that are new since the last time you Refreshed data. For example, you may have a Table that is updated daily with new sales transactions. Rather than rebuild the entire Refresh each day, you can just add the new transactions that occurred that day.

Incremental refresh enables very large Datasets in Power BI with the following benefits.

#### **Refreshes are faster**

Only data that has changed needs to be refreshed. For example, refresh only the last five days of a ten-year.

#### **Refreshes are more reliable**

It's no longer necessary to maintain long-running connections to volatile source systems.

#### **Resource consumption is reduced**

Less data to refresh reduces overall consumption of memory and other resources.

## Requirements to Perform Incremental Refresh

1. Table with Date Field(s) / Columns(s)
2. Data Source should supports Query Folding

Incremental Refresh Policies are defined in Power BI Desktop and applied when published to the Power BI service.

## Configure Incremental Refresh

### RangeStart and RangeEnd parameters

For Incremental Refresh, Tables in Datasets are filtered by using Power Query date/time parameters with the reserved, case-sensitive names **RangeStart** and **RangeEnd**. These parameters are used to filter the data imported into Power BI Desktop, and also to dynamically partition the data into ranges once published to the Power BI service. The parameter values are substituted by the service to filter for each partition. There's no need to set them in dataset settings in the service. Once published, the parameter values are overridden automatically by the Power BI service.

To define the parameters with default values, in the Power Query Editor, select Manage Parameters.

The screenshot shows the Power Query Editor interface with the 'Manage Parameters' dialog open. The 'File' tab is selected in the ribbon. A red box highlights the 'Parameters' icon in the ribbon, and a red arrow points from it to the 'Parameters' section in the dialog. The 'RangeStart' parameter is selected, with its details visible: Name is 'RangeStart', Description is 'RangeStart', Type is 'Date/Time' (with a checked 'Required' checkbox), Suggested Values is 'Any value', and Current Value is '01-12-2010 00:00:00'. The 'RangeEnd' parameter is also listed in the dialog. On the left, the 'Queries [6]' pane shows the 'FactInternetSales' query selected, and the 'OrderDate' column is highlighted. The main area displays the 'OrderDate' column data from the 'Adventure' database.

Row	OrderDate
1	29-12-2010 00:00:00
2	29-12-2010 00:00:00
3	29-12-2010 00:00:00
4	29-12-2010 00:00:00
5	29-12-2010 00:00:00
6	30-12-2010 00:00:00
7	30-12-2010 00:00:00
8	30-12-2010 00:00:00
9	30-12-2010 00:00:00
10	31-12-2010 00:00:00
11	31-12-2010 00:00:00
12	31-12-2010 00:00:00
13	31-12-2010 00:00:00
14	31-12-2010 00:00:00

With the parameters defined, you can then apply the filter by selecting the Custom Filter menu option for a column.

The screenshot shows the Power BI Desktop interface with a context menu open over the 'OrderDate' column header. The menu includes options like 'Sort Ascending', 'Sort Descending', 'Clear Sort', 'Clear Filter', 'Remove Empty', and 'Date/Time Filters'. Under 'Date/Time Filters', there is a search bar and a list of dates from 12/29/2010 to 1/14/2011. A warning message '⚠ List may be incomplete.' is displayed, along with a 'Load more' link. At the bottom of the menu, there are 'OK' and 'Cancel' buttons, and a 'Custom Filter...' button which is highlighted with a red box.

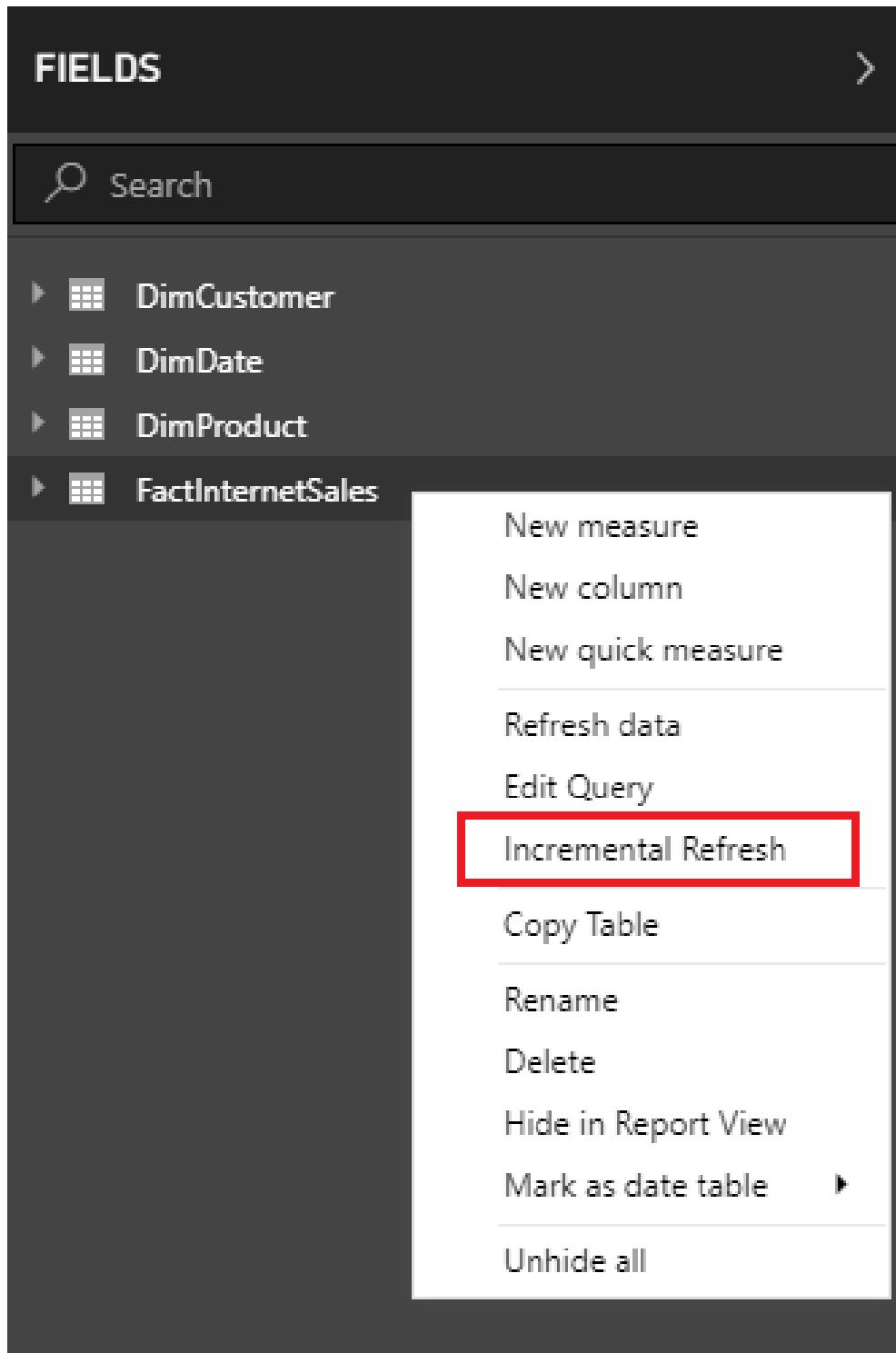
Ensure rows are filtered where the column value is after or equal to RangeStart and before RangeEnd. Other filter combinations may result in double counting of rows.

The screenshot shows the 'Filter Rows' dialog box. It has two tabs: 'Basic' (selected) and 'Advanced'. The 'Basic' tab allows filtering rows based on a single column ('OrderDate'). The 'Keep rows where 'OrderDate'' section contains the following logic:  
 - Condition: 'is after or equal to'  
 - Value: 'RangeStart'  
 - Operator: 'And'  
 - Condition: 'is before'  
 - Value: 'RangeEnd'  
 At the bottom are 'OK' and 'Cancel' buttons.

Select Close and Apply from the Power Query Editor. You should have a subset of the dataset in Power BI Desktop.

## Define the refresh policy

Incremental refresh is available on the context menu for tables, except for Live Connection models.



Incremental refresh dialog

The incremental refresh dialog is displayed. Use the toggle to enable the dialog.

X

## Incremental refresh

You can improve the speed of refresh for large tables by using incremental refresh. This setting will apply once you've published a report to the Power BI service.

- ⓘ Once you've deployed this table to the Power BI service, you won't be able to download it back to Power BI Desktop. [Learn more](#)



Store rows where column "OrderDate" is in the last:

Enter value...  Select value... ▾

Refresh rows where column "OrderDate" is in the last:

Enter value...  Select value... ▾

Detect data changes [Learn more](#)

Only refresh complete periods [Learn more](#)

Apply all

Cancel

### Note

If the Power Query expression for the table doesn't refer to the parameters with reserved names, the toggle is disabled.

#### *The header text explains the following*

Refresh policies are defined in Power BI Desktop and they are applied by refresh operations in the service.

It is not possible to download the PBIX file containing an incremental-refresh policy from the Power BI service

## Refresh Ranges

Incremental refresh

You can improve the speed of refresh for large tables by using incremental refresh. This setting will apply once you've published a report to the Power BI service.

Once you've deployed this table to the Power BI service, you won't be able to download it back to Power BI Desktop. [Learn more](#)

Table Incremental refresh  
FactInternetSales ▾  On

Store rows where column "OrderDate" is in the last:  
5 Years ▾

Refresh rows where column "OrderDate" is in the last:  
10 Days ▾

Detect data changes [Learn more](#)  
 Only refresh complete days [Learn more](#)

The following example defines a refresh policy to store data for five full calendar years plus data for the current year up to the current date, and incrementally refresh ten days of data. The first refresh operation loads historical data. Subsequent refreshes are incremental, and (if scheduled to run daily) perform the below operations

- Add a new day of data.
- Refresh ten days up to the current date.
- Remove calendar years that are older than five years prior to the current date. For example, if the current date is January 1 2019, the year 2013 is removed.

The first refresh in the Power BI service may take longer to import all five full calendar years. Subsequent refreshes may be finished in a fraction of the time.

## Advanced policy options

### Detect data changes

Incremental refresh of ten days is more efficient than full refresh of five years. However, it's possible to do even better. If you select the Detect data changes checkbox, you can select a date/time column used to identify and refresh only the days where the data has changed. This assumes such a column exists in the source system, which is typically for auditing purposes. This should not be the same column used to partition the data with the RangeStart/RangeEnd parameters. The maximum value of this column is evaluated for each of the periods in the incremental range. If it has not changed since the last refresh, there is no need to refresh the period. In the example, this could further reduce the days incrementally refreshed from ten to around two.

Incremental refresh

You can improve the speed of refresh for large tables by using incremental refresh. This setting will apply once you've published a report to the Power BI service.

Once you've deployed this table to the Power BI service, you won't be able to download it back to Power BI Desktop. [Learn more](#)

Table Incremental refresh  
FactInternetSales ▾  On

Store rows where column "OrderDate" is in the last:  
5 Years ▾

Refresh rows where column "OrderDate" is in the last:  
10 Days ▾

Detect data changes [Learn more](#)

Only refresh data in the last 10 days if the maximum value of this datetime column changes:  
LastUpdateDate ▾

Only refresh complete days [Learn more](#)

## Only refresh complete periods

Incremental refresh

You can improve the speed of refresh for large tables by using incremental refresh. This setting will apply once you've published a report to the Power BI service.

Once you've deployed this table to the Power BI service, you won't be able to download it back to Power BI Desktop. [Learn more](#)

Table	Incremental refresh
FactInternetSales ▾	<input checked="" type="checkbox"/> On

Store rows where column "OrderDate" is in the last:

5 Years ▾

Refresh rows where column "OrderDate" is in the last:

10 Days ▾

Detect data changes [Learn more](#)

Only refresh data in the last 10 days if the maximum value of this datetime column changes:

LastUpdateDate ▾

Only refresh complete days [Learn more](#)

Let's say your refresh is scheduled to run at 4:00 AM every morning. If data appears in the source system during those 4 hours, you may not want to account for it. Some business metrics -- such as barrels per day in the oil and gas industry -- make no sense with partial days.

## Publish to the service

You can now refresh the model. The first refresh may take longer to import the historical data. Subsequent refreshes can be much quicker because they use incremental refresh.

## Connect to files stored in OneDrive for your Power BI Workspace

After you've created a workspace in Power BI, you can store your Power BI Desktop files on the OneDrive for Business for your Power BI Workspace. You can continue updating the files you store in OneDrive. Those updates are automatically reflected in the Power BI Reports and Dashboards based on the files.

Importing files from OneDrive into the Power BI service is a great way to make sure your work in Power BI Desktop stays in sync with the Power BI service.

### Advantages of storing a Power BI Desktop file on OneDrive

When you store a Power BI Desktop file on OneDrive, any data you've loaded into your file's model is imported into the dataset. Any reports you've created in the file are loaded into Reports in the Power BI service. Let's say you make changes to your file on OneDrive. These changes can include adding new measures, changing column names, or editing visualizations. Once you save the file, Power BI service syncs with those changes too, usually within about an hour.

*Adding files to your workspace is a two-step process*

1. First you upload files to the OneDrive for Business for your workspace.
2. Then you connect to those files from Power BI Service.

## **Report Settings**

### **Report Name**

You can use this Section to Rename a Report.

### **Description**

You can use this Section to Describe the Report.

### **Contact**

You can use this Section to Set contact information for this report.

The info card is shown when viewing a report.

When you click the list of contacts, an email is created so you can ask questions or get help of this Report.

### **Snapshot&Featured**

You can use this Section to promote your reports to Featured content on Power BI Home.

### **Persistent filters**

By Default when Clients interact with the Visuals, the filters will be saved.

Sometime we don't want Clients to save the filters on the Report. Enable the Option "Don't allow end user to save filters on this report".

### **Visual options**

1. Hide the visual header in reading view.
2. Change default visual interaction from cross highlighting to cross filtering.

### **Export data**

Choose the type of data you allow your end users to export.

- Summarized Data
- Summarized Data & Underlying Data
- None

### **Filtering experience**

1. Enable the updated filter pane, and show filters in the visual header for this report.
2. Allow users to change filter types.
3. Enable search for the filter pane.

### **Cross-report drill through**

Allow visuals in this report to use drill-through targets from other reports.

### **Comments**

Allow users to add comments to this report.

### **Personalize visuals (preview)**

Allow report readers to personalize visuals to suit their needs.

## Shared and Certified Datasets

### SharedDatasets

In many cases, Data already exists to answer your Business Questions. The challenge for the data provider is to share it in a way that encourages reuse and preserves a single version of the truth. For the Report Author or Developer, the challenge is to find reliable and high-quality data quickly and easily in both the Power BI Service and Power BI Desktop.

With shared Datasets in Power BI, a single Dataset Can used by Multiple Reports, across Workspaces. You can either build new Reports based on Datasets in different Workspaces or you can copy existing Reports across Workspaces.

### Certify Datasets / Endorse Dataset

*Endorsement - Certified - Acceptance, Approval, Sanction, Consistence*

Your Power BI Report Creators may now have access to many different Datasets, so enterprises need to guide them to the reliable, high-quality Datasets. Power BI provides two ways to endorse Datasets.

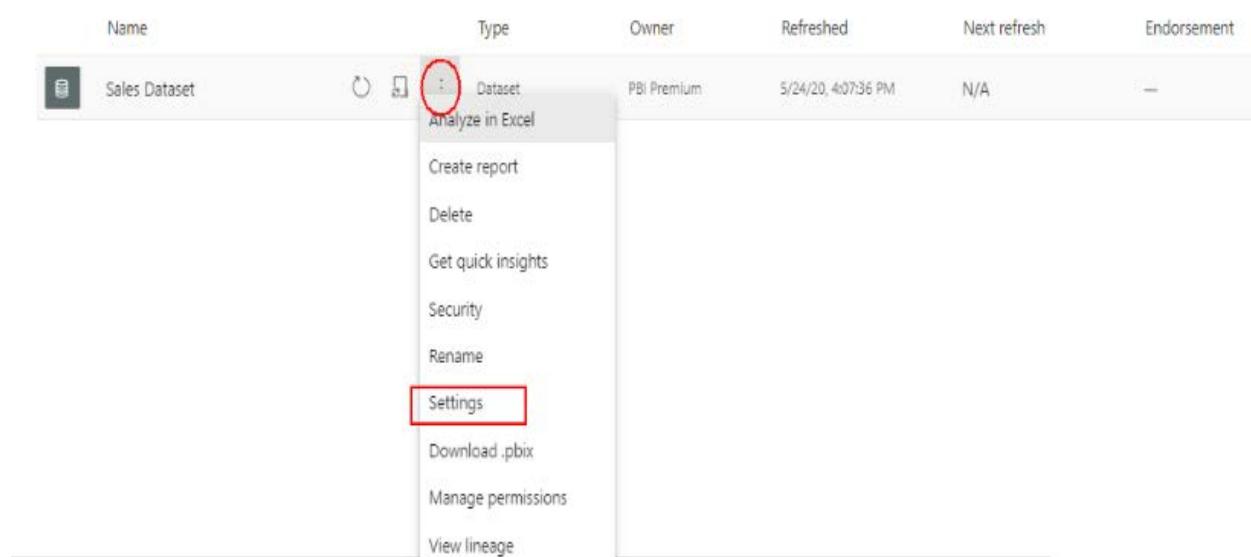
#### 1. Promote

As a Dataset owner, you can promote your own Datasets when they're ready for wide-spread usage. Any Workspace member with Write permissions can promote a Dataset. There are no restrictions on who can promote a dataset. Promotion supports the collaborative spread of datasets within organizations. Now we will see how to promote a Dataset.

#### Promote a Dataset

Go to the list of datasets in the workspace.

Select more options (...), and then select Settings.



Expand **Endorsement**> select **Promoted**.

The screenshot shows the 'Endorsement' section of a dataset's properties. It includes options for 'Default', 'Promoted' (which is selected and highlighted with a red box), and 'Certified'. A description input field is present, and the 'Apply' button is highlighted with a red box.

Select **Apply**.

## 2. Certify

### Request Dataset Certification

#### Certification

You can request certification for a promoted dataset. A select group of users defined in the Dataset Certification tenant admin setting decides which datasets to certify.

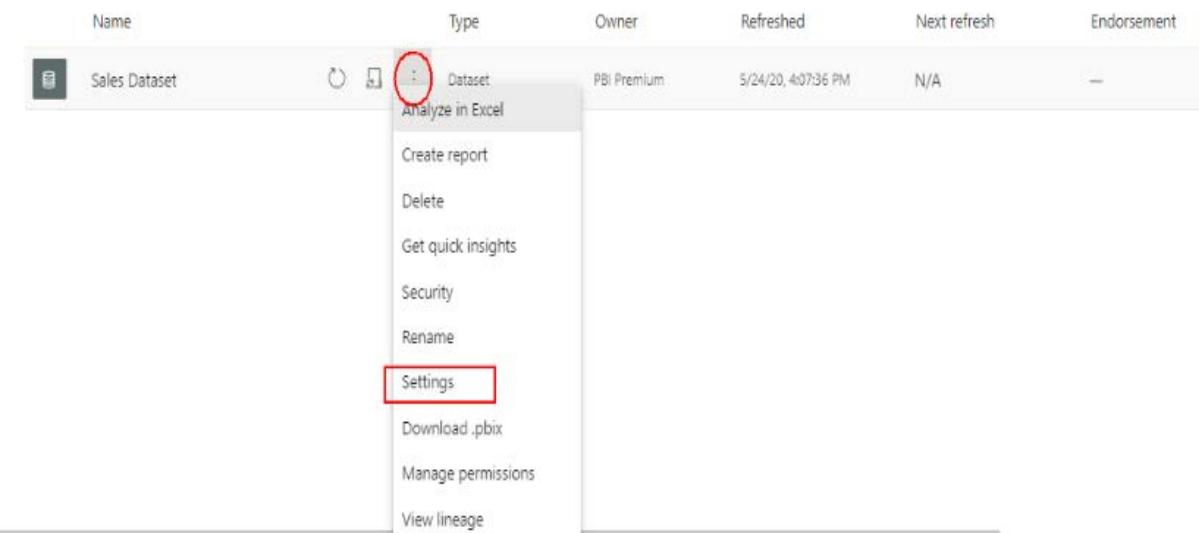
Your organization can certify datasets that are the authoritative source for critical information. These datasets are featured prominently when report designers start creating a report and looking for reliable data. Certification can be a highly selective process, with only the most valuable datasets getting certified. Power BI tenant admins have a new setting, so they can tightly control who can certify datasets. Thus, admins can ensure that dataset certification results in truly reliable and authoritative datasets designed for use across the organization.

Your tenant admin has identified people in your organization who can certify datasets. You can ask that they certify your dataset.

Give the certifier member permissions for the workspace where the dataset resides.

Go to the list of datasets in the workspace.

Select more options (...), and then select Settings.



1. The dataset owner needs to give you member permissions for the workspace where the dataset resides.
2. If your tenant admin has named you as someone who can certify datasets, the certified option in the Endorsement section of Settings for the dataset is available. Select Certified.

**Endorsement**

Help your colleagues find, learn about, and connect to your dataset.

Default  
This dataset can be searched for and used by others.

Certified  
Request certification from experts in your org to get a badge that shows it's recommended for use by others. [Learn more](#)

Description

Describe the contents of this dataset.

500 characters left

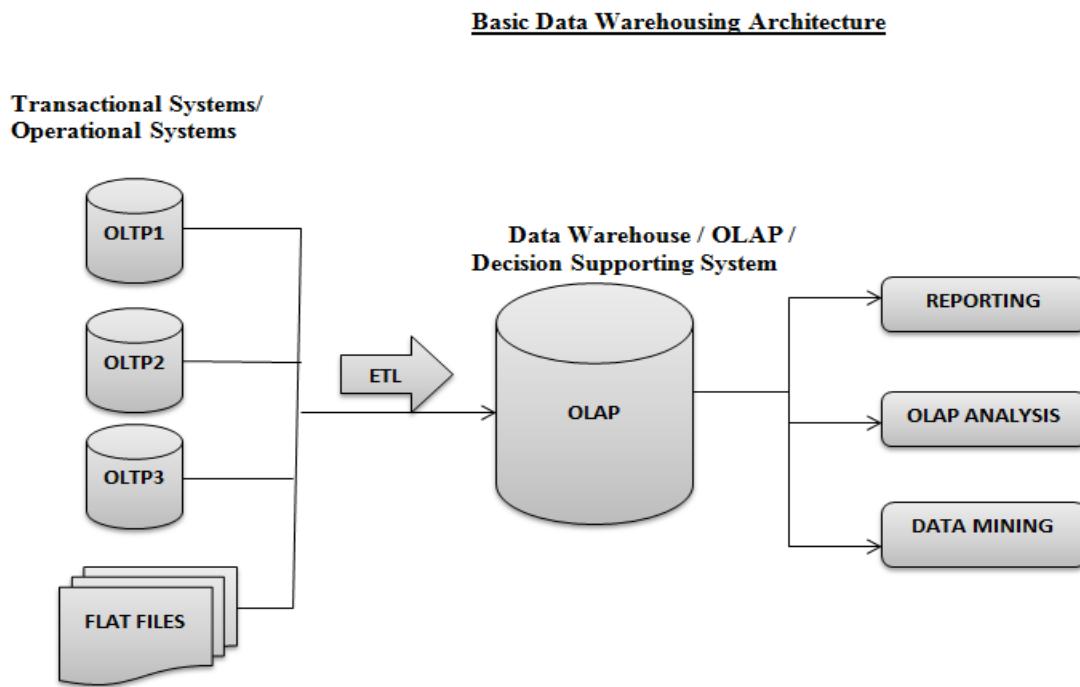
**Apply** **Discard**

3. Select **Apply**.

### Data Warehousing Definition

Data Warehousing is the process of constructing a Data Warehouse and using it.

### Basic Data Warehousing Architecture



Basic Data Warehousing Architecture contains Operational Systems or Transactional Systems which stores the Transactional Data and OLAP / Data Warehouse / Decision Supporting System (DSS) which stores the Analytical Data that is used for Analysis and Reporting. From Operational Systems we will Extract the Data and Transform the Data as per the Organization Needs and load the Data into Data Warehouse using ETL tools (Informatica). From Data Warehouse by using OLAP Tools or Reporting Tools we will generate reports, perform OLAP Analysis and Data Mining.

The Operational Systems may be OLTP Databases or Flat Files etc., which stores Transactional Data.

### Transaction & Transactional Data

Transactional data describe about a transaction that takes place in an organization. Examples for Transactions are sales orders, invoices, purchase orders, shipping documents, passport applications, credit card payments, and insurance claims etc.

### OLTP / Transactional System / Operational System

- ✓ OLTP stands for Online Transactional Process which stores transactional data.
- ✓ The Operational systems are where the data is put in.
- ✓ The users of an operational system turn the wheels of the organization.
- ✓ Users of an operational system almost always deal with one record at a time.
- ✓ They repeatedly perform the same operational tasks over and over.

### OLAP / Data Warehouse / Decision Supporting System (DSS)

- ✓ OLAP stands for Online Analytical Process which stores Analytical data that is used for Analysis and Reporting.
- ✓ Data Warehouse is where we get the data out.
- ✓ The users of the Data Warehouse watch the wheels of organization.
- ✓ Users of the Data Warehouse almost never deal with one row at a time rather, their questions often require hundreds or thousands of rows are searched and compressed into an answer set.
- ✓ Users of a Data Warehouse continuously change the kind of required questions they ask.

### Need of Data Warehouse?

Let us assume 'ABC' bank operates in multiple Countries.

And let us say Country1 data is residing in OLTP1, Country2 data in OLTP2 and Country3 data in OLTP3.

If one day ABC Bank requires consolidated Reports,

There are two ways

1. Completely manual, generate different reports from different OLTP's and integrate them to get the consolidated report.
2. Fetch all the data from different OLTP's, made it coherent (consistent manner), load to Data Warehouse and generate the Reports from Data Warehouse.

Obviously second approach is the best.

### Issues with OLTP Reporting

#### 1. Less History

The OLTP Systems does not maintain Complete History in order to have the better transaction performance. So it is not possible to analyze the data completely for a wide range.

#### 2. Performance issue

- ✓ As we are fetching the data from multiple transactional systems to generate consolidated report obviously it takes some time to get the final consolidated report.
- ✓ As the OLTP systems are highly normalized and hence to get the report output we need to join more number of tables.
- ✓ Also it is not recommendable to Insert and Retrieve Data from the same system at the same time.

#### 3. Data quality and data consistency issues

As we are generating the Reports from OLTP's on the fly with some data transformations at the Report level to make data consistent and as it is not possible to test the reports output every time so there might be a chance of data quality and data consistency issues and also performance will degrades as we are performing the data transformation.

#### 4. Confident level

Obviously there will be low confident level from the users (BI People)

If we generate reports using Data Warehouse we will overcome the above issues.

#### Benefits with OLAP Reporting

- ✓ OLAP will maintain complete History so that we can make Better Analysis using complete Data.
- ✓ There will be no performance issues because we have the complete data from all the transactional sources in Data Warehouse.
- ✓ As the data in the Data Warehouse is in de-normalized form, hence to get the report output you no need to join more number of tables.
- ✓ There will be no data quality and data consistency issues as a Data Warehouse gets the preprocessed data from well Designed and Tested ETL code developed using any of the ETL tools (Informatica).
- ✓ Obviously user (BI People) will be confident at the report output.

#### Benefits of Building a Data Warehouse

- ✓ Data Warehouse provides more accurate and complete data.
- ✓ Data Warehouse provides easy access for end users.

#### Need of ETL Tools?

- ✓ ETL stands for Extraction, Transformation and Load.
- ✓ ETL tool provides the facility to extract data from different source systems, transform the data and load the data into target systems.
- ✓ To fetch the data from different sources and to make the data coherent or consistent and to load the data into the Data Warehouse we need ETL Tools.

#### Need of BI Tools?

As the Business Users don't know the SQL to communicate with the Databases so we generate the Reports using any of the BI Tools or Reporting Tools and we will share the Reports to end users for Analysis.

#### Data Warehouse Definition

A Data Warehouse is a copy of transactional data specifically structured for Reporting and Analysis.

--Ralph Kimball

A Data Warehouse is a Subject Oriented, Integrated, Time Variant and Non Volatile collection of data in support of management's decision making process.

--Bill Inmon

### Characteristics of Data Warehouse

#### 1. Subject Oriented

Data Warehouse is Subject Oriented rather than Application Oriented. Data in the Data Warehouse belongs to a specific subject.

Example: Sales, Finance, HR etc.

#### 2. Integrated

Usually the source data for Data Warehouse is coming from different sources so we will integrate and load data into Data Warehouse.

#### 3. Time Variant

Data entered into the Data Warehouse belongs to a specific time period. Data Warehouse will contain entire History of Data and to compare the data we need to maintain the timestamp for every record loaded into the Data Warehouse.

#### 4. Non Volatile

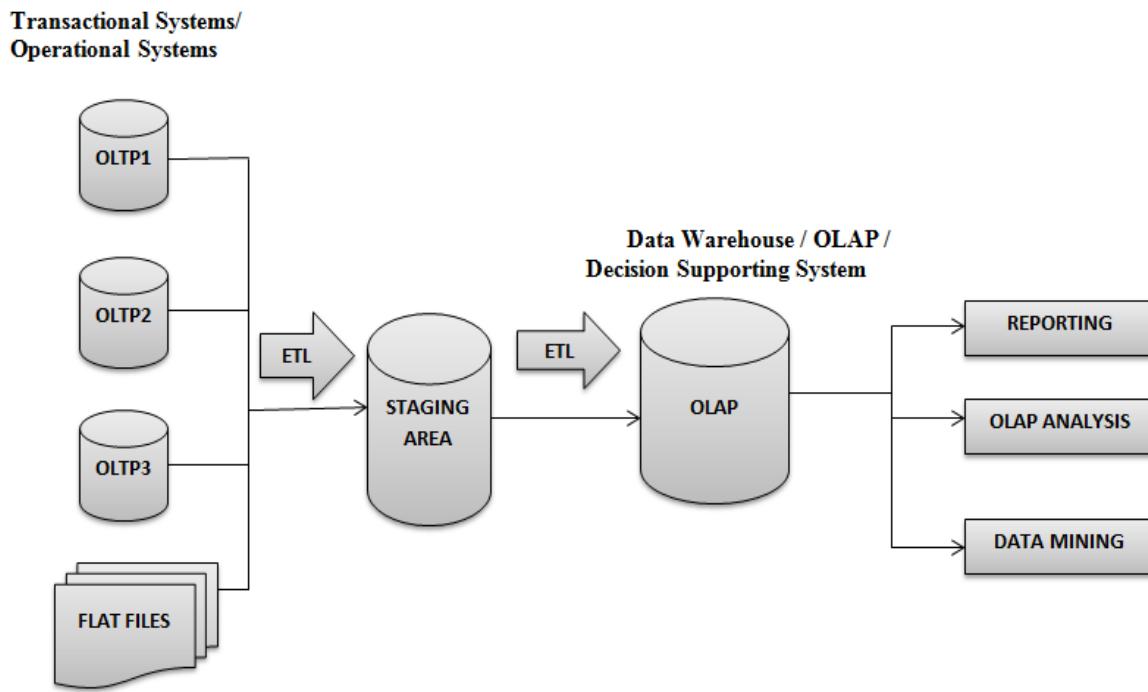
Data entered into Data Warehouse never deleted or removed.

### Differences between OLTP and OLAP

OLTP	OLAP
OLTP stands for Online Transactional Process which stores Transactional Data	OLAP stands for Online Analytical Process which stores Analytical Data that is used for Analysis
Designed to support business transactional process	Design to support business decision making process
Application Oriented Data	Subject Oriented Data
Less History	Complete History
Designed for Clerical access	Designed for Managerial access
Designed using ER Modeling	Designed using Dimension Modeling
More joins	Less joins
Normalized data (No data redundancy)	De Normalized data (Data Redundancy)

## Data Warehousing Architecture with Staging Area

### Data Warehousing Architecture with Staging Area



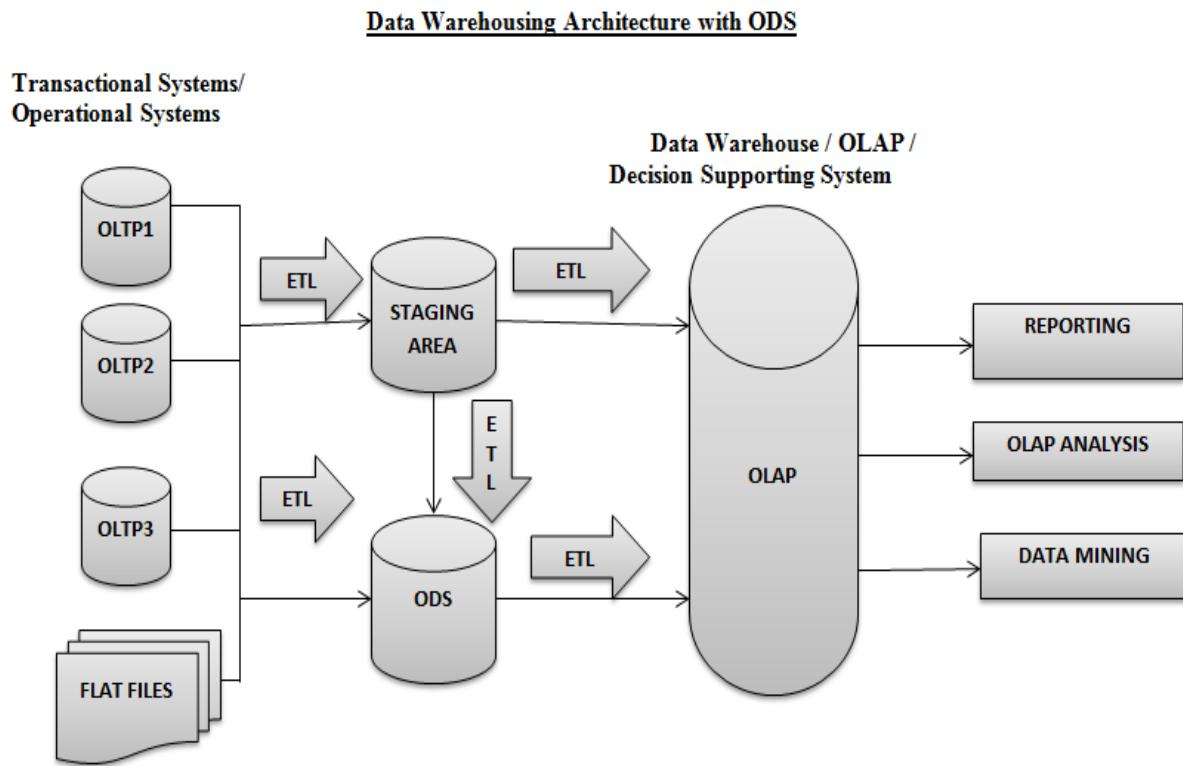
### Staging Area

- ✓ Data Staging Area is a database and is an intermediate storage area between the Transactional sources and Data Warehouse / Data Mart.
- ✓ It is an integrated view of multiple Transactional sources.
- ✓ It is usually of temporary in nature, and its contents can be erased after the Data Warehouse / Data Mart has been loaded successfully.

### Need of Staging Area

- ✓ If we have different sources for easy communication we use Data Staging Area.
- ✓ As we don't have the access to the data sources throughout the day. So it's better to get the data into staging and then to do the transformations on the Data.
- ✓ Suppose if we need to join 2 Tables data which comes on a particular day. Joining 2 tables in OLTP and filtering the data for a particular day will be time taking process. So we will get particular day data to the staging database and then we will join.

### Data Warehousing Architecture with ODS



### Operational Data Store (ODS)

ODS is used for many purposes. Let us see few of them.

1. An ODS is a database designed to integrate data from multiple sources for **additional operations on the data**. It may be passed for further operations and to the data warehouse for reporting.
2. An ODS is a database designed to do immediate reporting with current operational data. An ODS must be frequently refreshed so that it contains very current data. An ODS can be updated daily, hourly or even immediately after transactions on operational sources. **It is used for real time and near real time reporting.**

The Data Warehouse will not update immediately once the transactions happened in operational sources.

- ✓ ODS is directly loaded from Operational sources or Staging Tables.
- ✓ It can optionally serve as a data source for the Data Warehouse.

### Data Mart

- ✓ A Data Mart is a Subject Oriented database which supports the business needs of specific department business managers.
- ✓ A Data Mart is subset of an Enterprise Data Warehouse.
- ✓ A Data Mart is a single Subject View and integration of multiple subject views is called an Enterprise Data Warehouse.
- ✓ A Data Warehouse is a database provides enterprise business view and Data Mart is also a database provides department specific business view.
- ✓ A Data Mart is known as High Performance Query Structure (HPQS).

### Data Warehouse Design Approaches

1. Top - Down Data Warehouse Design Approach
2. Bottom - Up Data Warehouse Design Approach

#### Top-Down Data Warehouse Design Approach

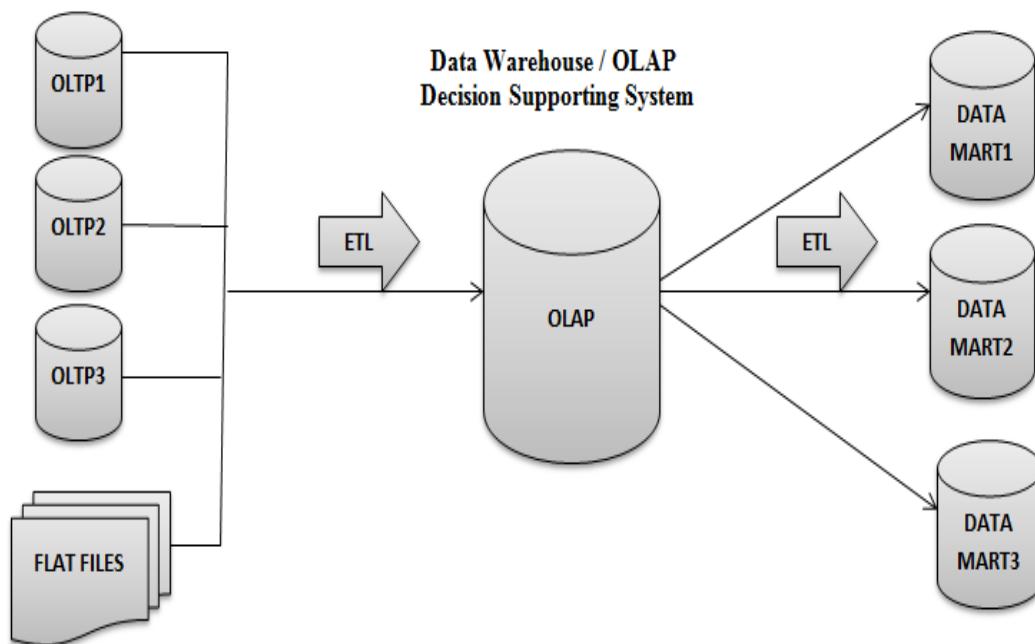
--Bill Inmon

#### Top - Down Data Warehouse Design Approach

Bill Inmon

Transactional Systems/  
Operational Systems

Data Marts



According to **Bill Inmon** first we need design an Enterprise Data Warehouse, from EDWH design the Subject Oriented, department specific databases known Data Marts are Design.

### Dependent Data Marts

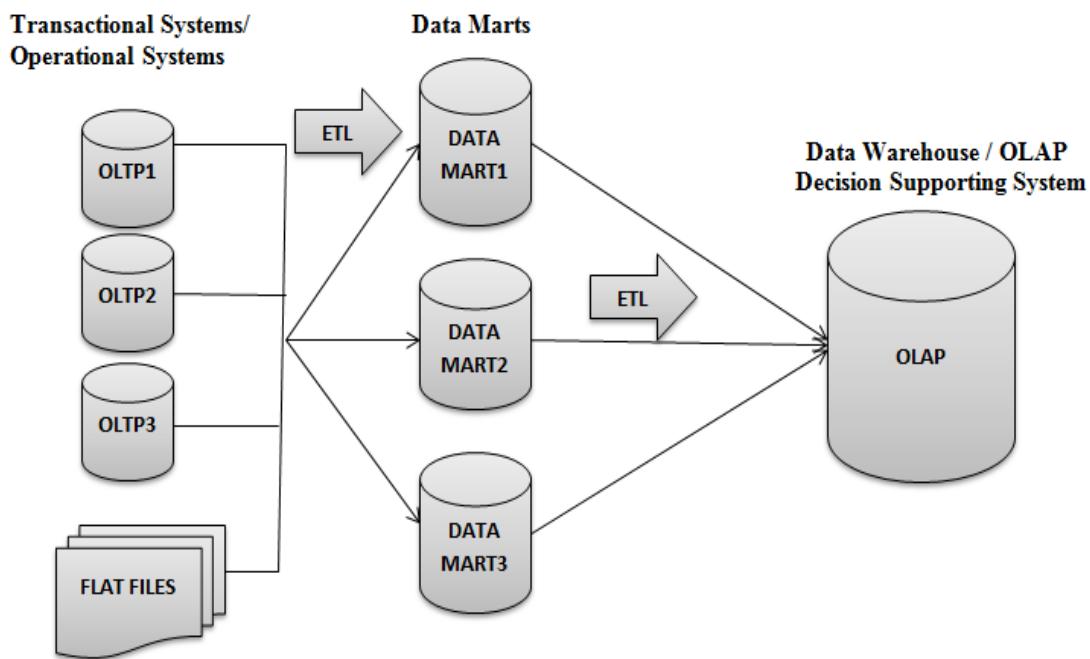
In Top - Down DWH design approach a Data Mart Development is dependent on EDWH hence such Data Marts are known as dependent Data Marts.

### Bottom - Up Data Warehouse Design Approach

--Ralph Kimball

### Bottom - Up Data Warehouse Design Approach

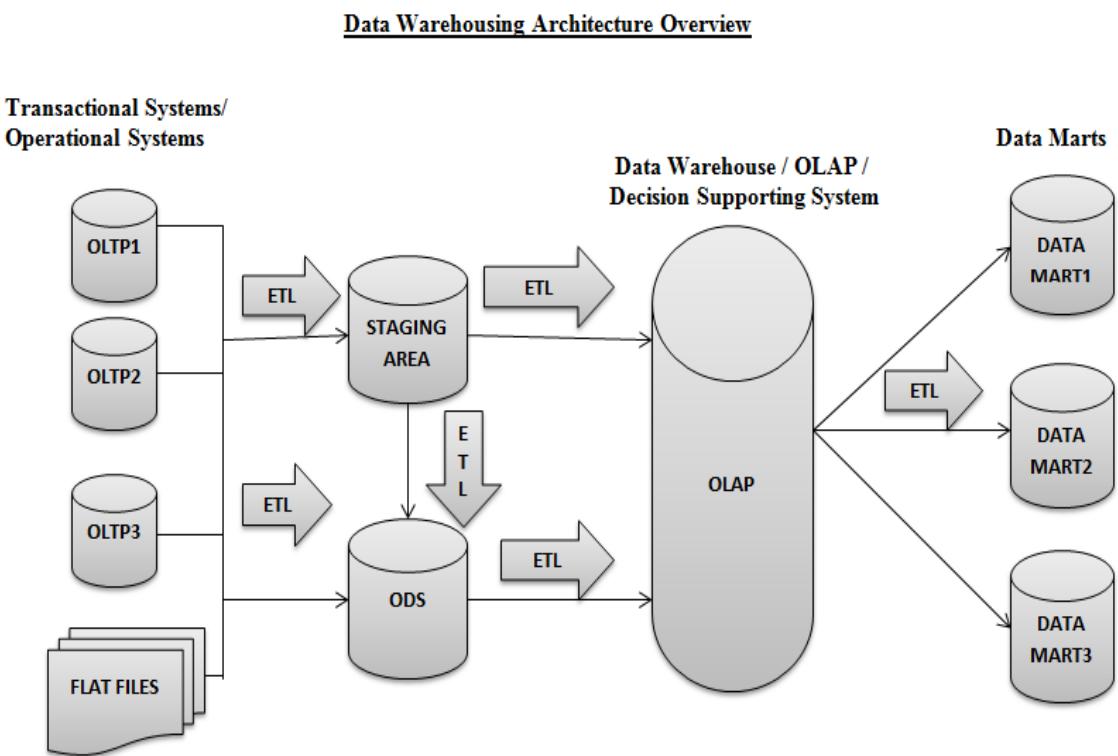
Ralph Kimball



According to **Ralph Kimball** first we need to design subject specific databases know as Data Marts and the integrate Data Marts into an EDWH.

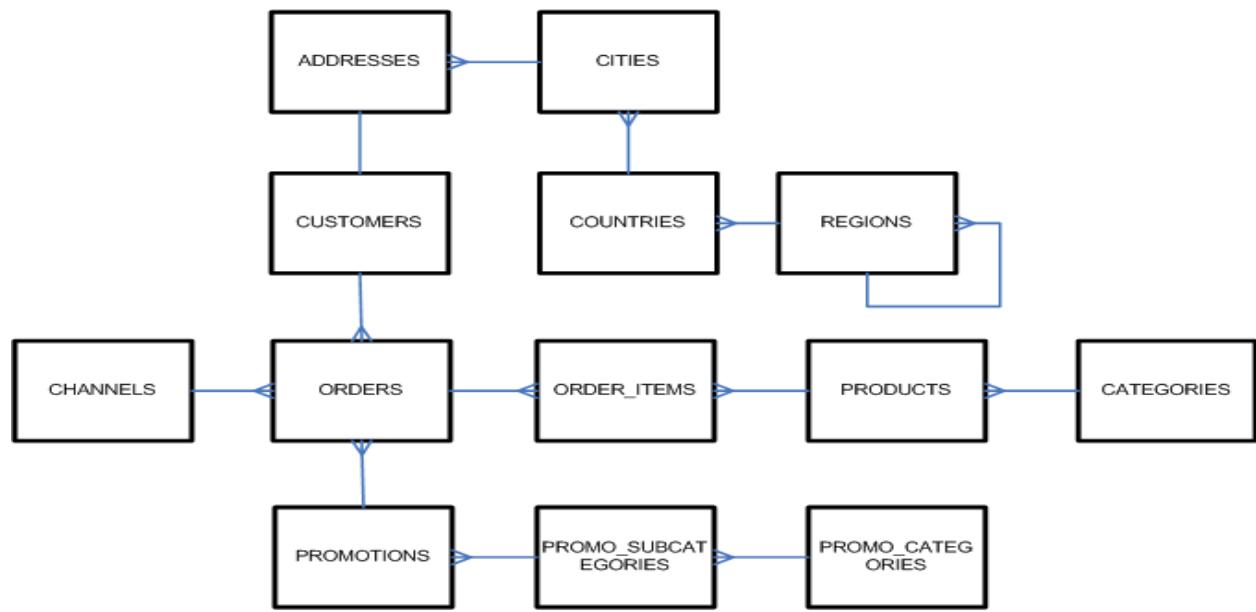
### Independent Data Marts

In a Bottom - Up Data Warehouse design approach a Data Mart development is independent on EDWH hence such Data Marts are known as independent Data Marts.

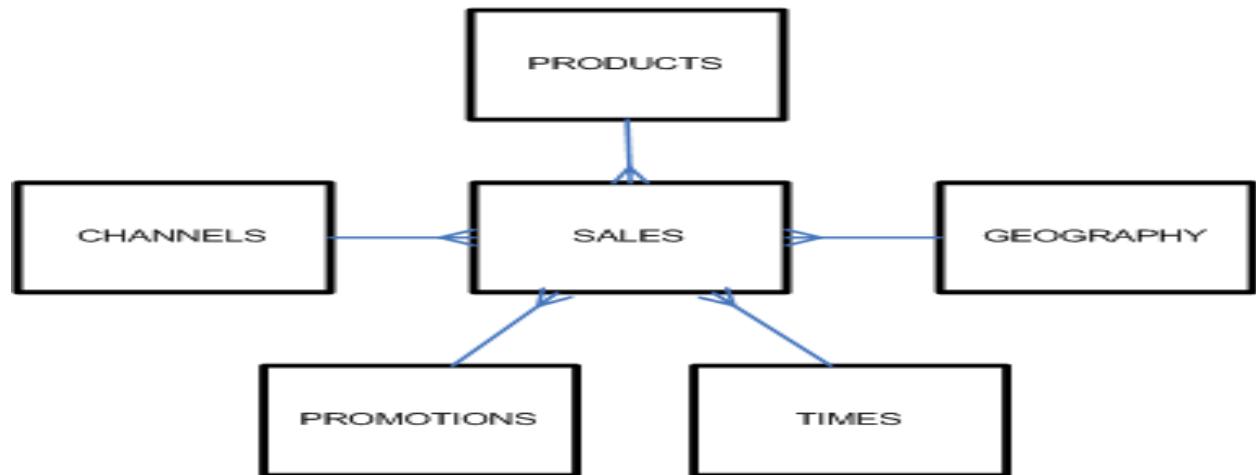
Data Warehousing Architecture Overview

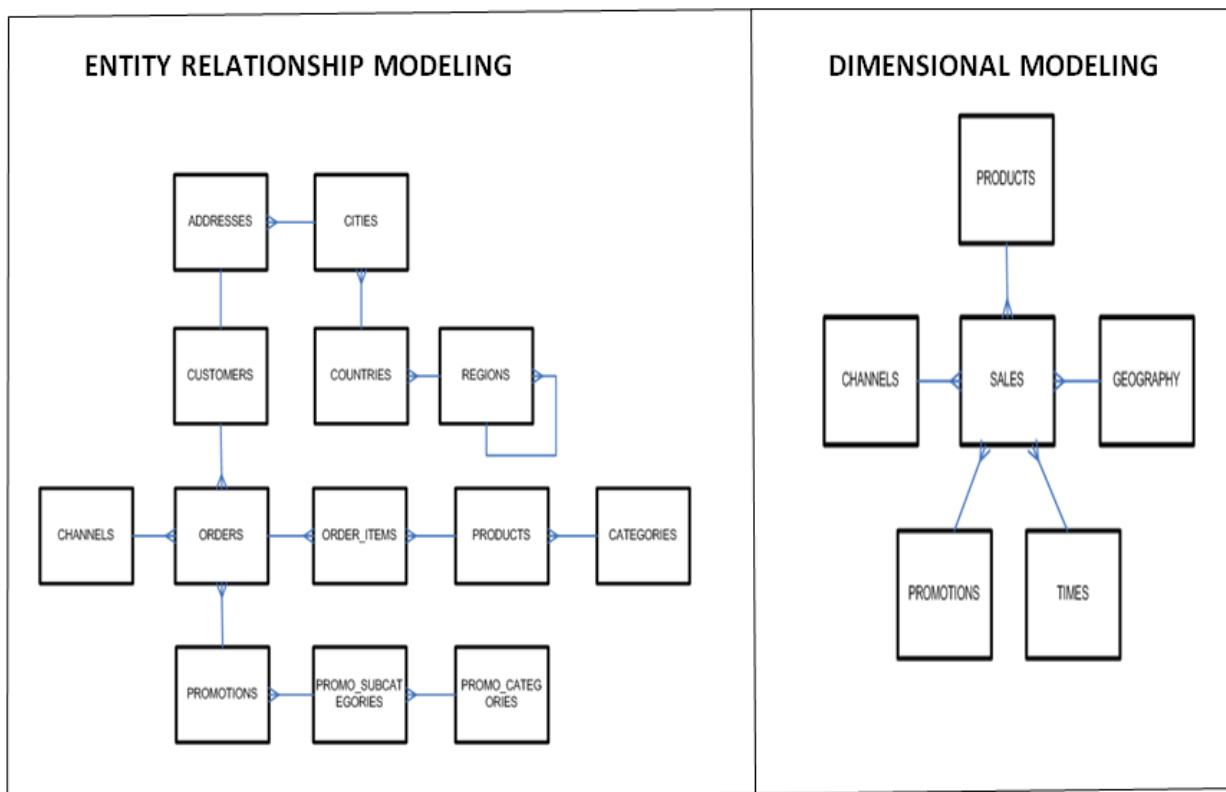
**ER Modeling (Entity Relationship Modeling)**

- ✓ Entity Relationship Modeling is used to design OLTP databases.
- ✓ Entity - Relationship modeling is a particular design methodology of data modeling wherein the goal of modeling is to **normalize** the data by reducing data redundancy.

**Dimensional Modeling**

- ✓ Dimensional Modeling is used to design OLAP databases.
- ✓ Dimensional Modeling is a particular design methodology of data modeling wherein the goal of modeling is to improve **query performance**.



ER Modeling Vs Dimensional Modeling

- ✓ REGIONS
- ✓ COUNTRIES
- ✓ CITIES

- ✓ GEOGRAPHY

Entity Relationship Modeling Table Structure

REGION	
REGION_CODE (P.K)	REGION_NAME
STH	SOUTH
NTH	NORTH
WST	WEST
EST	EAST

COUNTRY		
COUNTRY_CODE(P.K)	COUNTRY_NAME	REGION_CODE(F.K)
IND	INDIA	STH
SL	SRI LANKA	STH
PAK	PAKISTAN	STH
SA	SOUTH AFRICA	STH

CITY		
CITY_CODE(P.K)	CITY_NAME	COUNTRY_CODE(F.K)
AP	ANDHRA PRADESH	IND
TG	TELANGANA	IND
TN	TAMIL NADU	IND
KA	KARNATAKA	IND

### Dimensional Modeling Table Structure

GEOGRAPHY						
GEOGRAPHY_KEY(P.K)	REGION_CODE	REGION_NAME	COUNTRY_CODE	COUNTRY_NAME	CITY_CODE	CITY_NAME
1001	STH	SOUTH	IND	INDIA	AP	ANDHRA PRADESH
1002	STH	SOUTH	IND	INDIA	TG	TELANGANA
1003	STH	SOUTH	IND	INDIA	TN	TAMIL NADU
1004	STH	SOUTH	IND	INDIA	KA	KARNATAKA

In the process of Data Modeling to design a Data Warehouse using Dimensional Modeling we will be getting any one of the below schemas as a result.

### Types of Dimensional Schemas

1. Star Schema
  2. Snow Flake Schema
  3. Galaxy Schema or Fact Constellation Schema
- ✓ Data Warehouse Schemas are designed using Dimensional Modeling. A Schema consists of dimension tables and fact tables.
  - ✓ A dimension table contains dimensions and primary keys.
  - ✓ A fact table contains facts and foreign keys to the dimension tables.

### Dimension

- ✓ A dimension is a descriptive data which describes the key performance indicators known as facts.

E.g. Product, Customer Name, Date etc.

### Fact

- ✓ A fact is something that is measurable or quantifiable.
- ✓ Fact is the metric that business users would use for making business decisions.
- ✓ Measurable information of the business which can be analyzed and has the impact on the business is known as fact.

E.g. Profit, Revenue, Price etc.

Without dimensions we cannot measure the facts.

Profit is a fact, and if I say 1000\$ as profit it doesn't have any meaning and we need to add some dimensions to a fact to give a meaningful sentence.

Product, Month and Region are the dimensions and, if I say we had a profit of 1000\$ by selling Samsung Mobiles (Product) for the Month of Nov 2015 under the region South, then it give us the meaningful sentence.

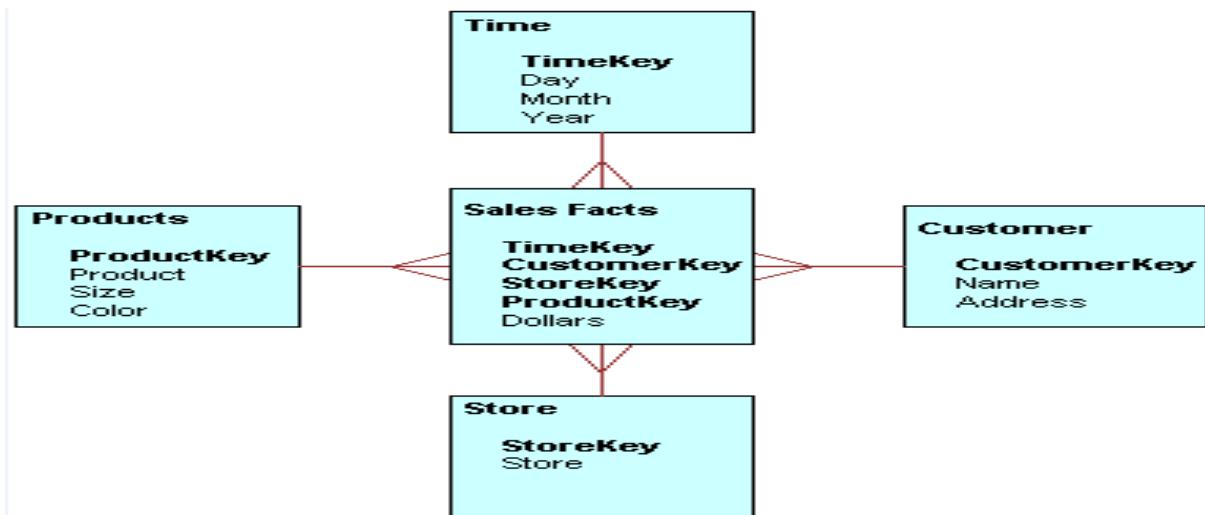
A fact table works with dimension table.

A fact table holds the data to be analyzed, and a dimension table stores data about the ways in which the data in the fact table can be analyzed.

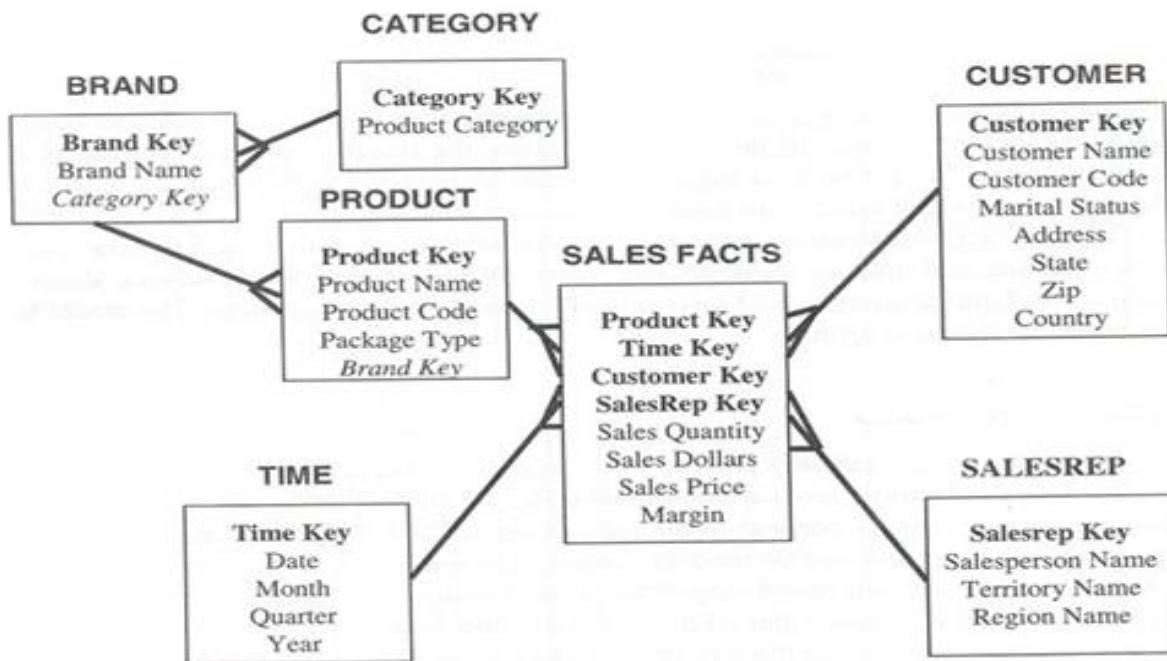
### **Example**

- ✓ South Region Profit
- ✓ Mobile sales in the year 2015 under North Region

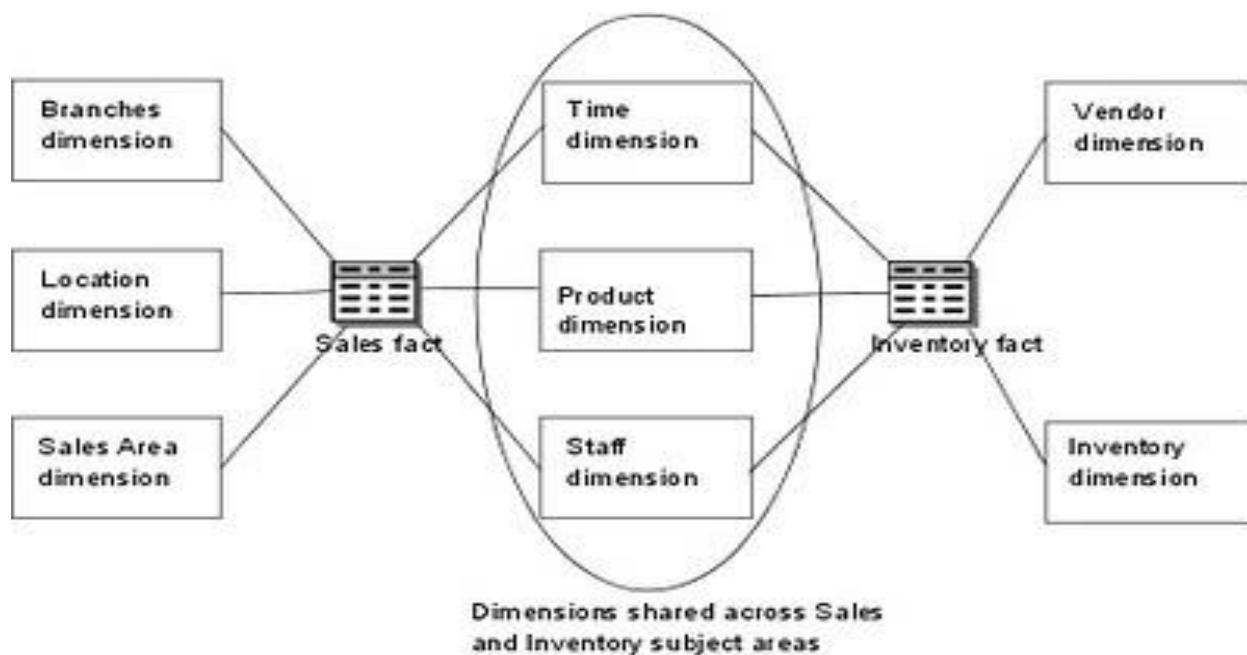
### Star Schema



- ✓ A Star Schema is a Data Warehouse database design which contains a centrally located fact table which is surrounded by multiple dimension tables.
- ✓ In star schema all the dimensional tables directly connect to the fact table.

Snow Flake Schema

- ✓ A Snow Flake Schema consists of a fact table surrounded by multiple dimension tables which can be connected to other dimension tables.
- ✓ In snowflake schema some of the dimensions will not directly connect to fact table.
- ✓ When dimension tables stores large number of rows with redundancy of data and space is such an issue, we can use snowflake schema to save space.
- ✓ The tables are partially de normalized in structure.
- ✓ Performance of SQL queries are a bit less when compared to star schema as more number of joins are involved.
- ✓ Data Redundancy is low and occupies less disc space when compared to star schema.
- ✓ In the above figure PRODUCT Dimension is partially normalized.

Fact Constellation Schema or Galaxy Schema

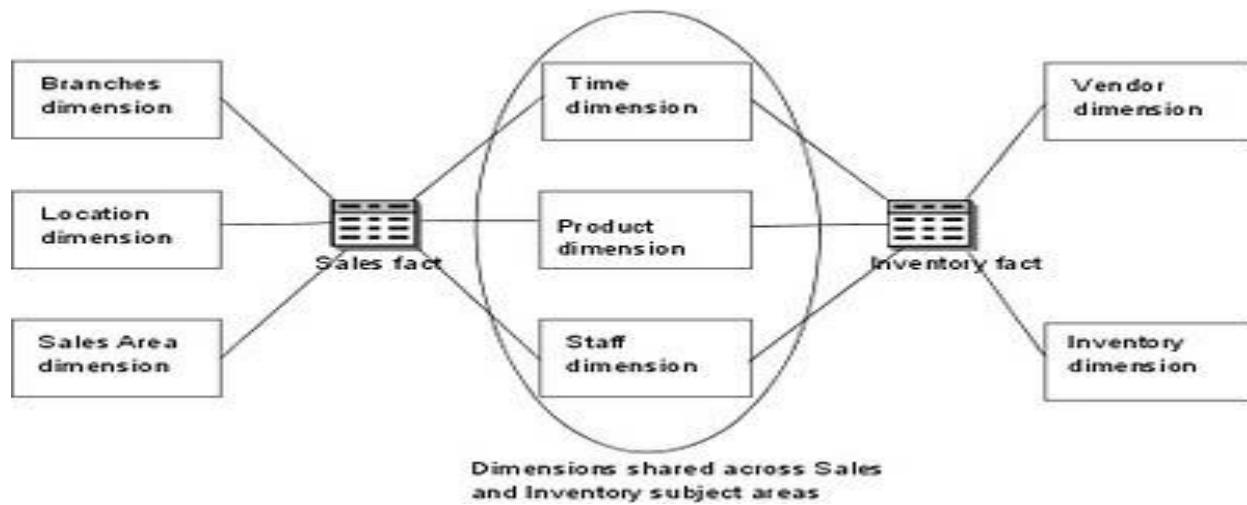
- ✓ This schema is viewed as collection of stars hence called Galaxy Schema or Fact Constellation Schema.
- ✓ In a Galaxy schema a single dimension table is shared with multiple fact tables.

Benefits of Dimensional Model

- ✓ Faster data Retrieval
- ✓ Better Understandability
- ✓ Easy Extensibility

### Types of Dimensions and Dimension Tables

#### Conformed Dimension Table



A Dimension Table that is shared across multiple subject areas or relates to multiple fact tables within the same data warehouse is known as conformed dimension.

Multiple fact tables are used in Data Warehouse that address multiple business functions such as Sales, Inventory and Finance etc.

Each business function will typically have its own Schema (Subject Area) and each schema contains a fact table, some conforming dimension tables and some dimension tables unique to the specific business function.

Date or Time Dimension is a common conformed dimension because its attributes (day, week, month, quarter, year etc.) have the same meaning when joined to any fact table.

#### Junk Dimension Table

A Junk Dimension is a collection of random transactional codes or flags or text attributes that are unrelated to any particular dimension.

The Junk dimension is simply a structure that provides a convenient place to store the Junk attributes.

Assume that we have a gender dimension and Marital Status dimension as below

Dim_Gender	
Gender_Key	Gender_Value
1001	M
1002	F

Dim_Marital_Status	
Marital_Key	Marital_Value
2001	Y
2002	N

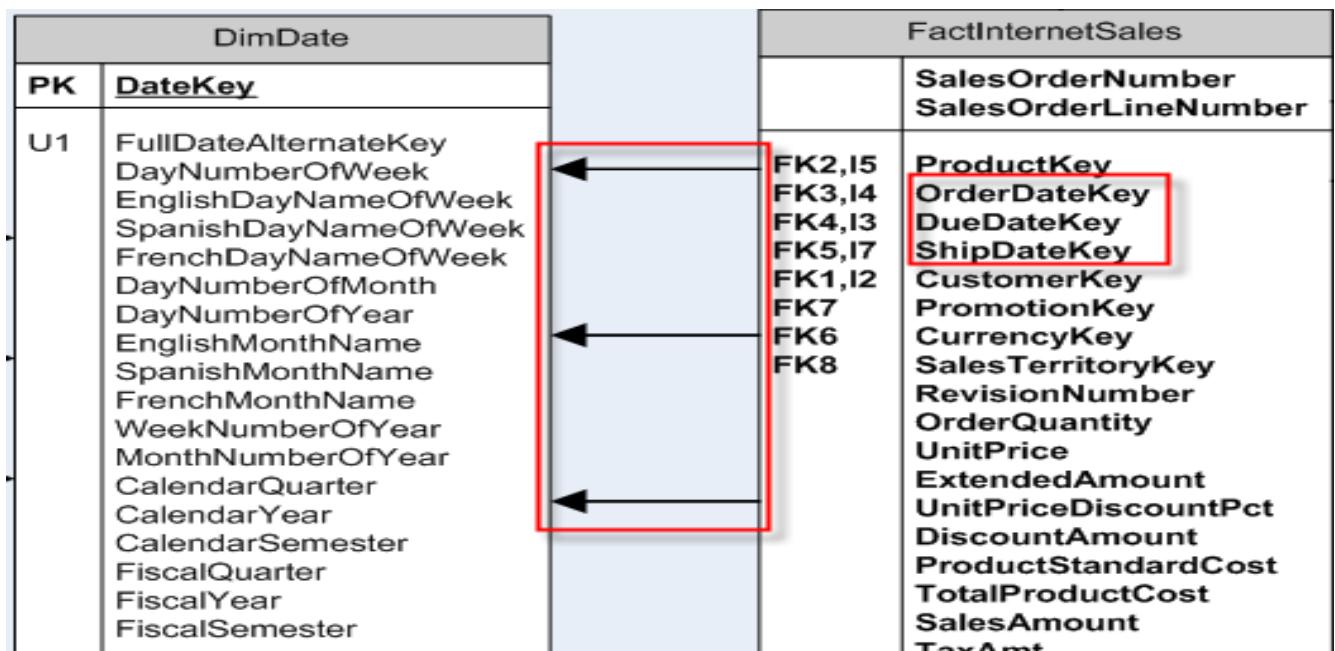
If we have two dimension tables then in the fact table we need to maintain two keys referring to these dimensions.

Instead create a Junk dimension Dim\_Junk which has all the combinations of Gender and Martial Status (cross join Gender and Martial Status) as below. Now we can maintain only one key in the fact table.

Dim_Junk		
Junk_Key	Gender_value	Marital_status_value
1001	M	Y
1002	M	N
1003	F	Y
1004	F	N

With this Junk dimensional table we will avoid small dimension tables and maintaining them and also we will save space in the fact table by avoiding multiple foreign keys.

### Role Playing Dimension Table



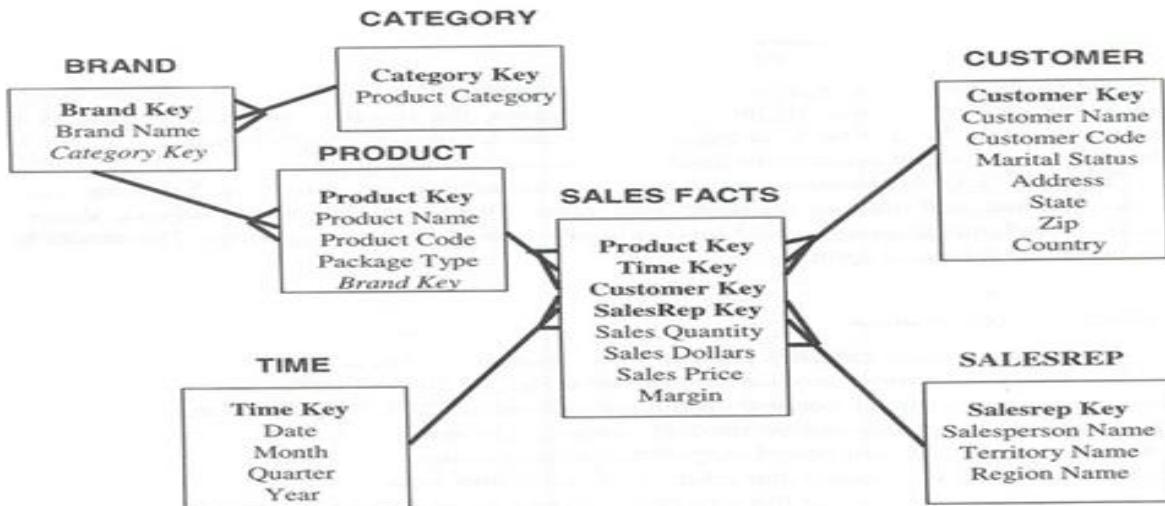
Role playing dimension refers to a dimension that can play different roles in a fact table depending on the content.

If one dimension key is attached to multiple foreign keys in the fact table then such a kind of dimension is known as role playing dimension.

For example, the Date Dimension can be used for the order date, scheduled shipping date, and shipped date in an order fact table.

In the data warehouse you will have a single dimension table for Dates and you will have multiple warehouse foreign keys from the fact table to the same dimension.

### Shrunken Dimension

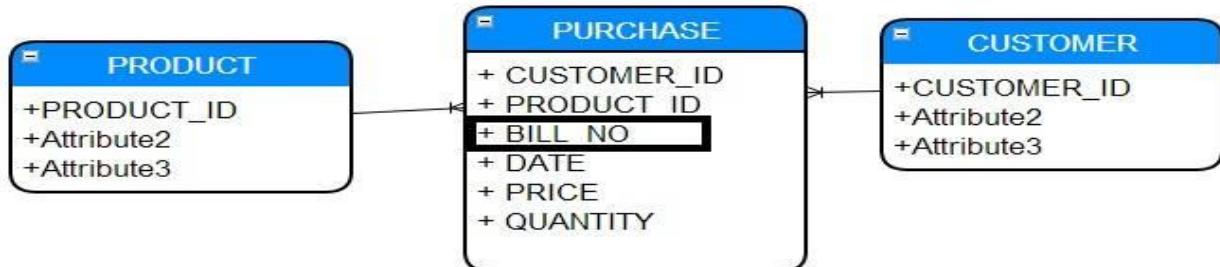


A shrunken dimension is a subset of another dimension.

E.g. In the above figure Product is the shrunken dimension of Brand and Brand is the shrunken dimension of Category.

Similarly Quarter is the Shrunken Dimension of Year and Month is the Shrunken Dimension of Quarter.

### Degenerated Dimension



A degenerated dimension is a dimension that is stored in the fact table rather than the dimension table.

A degenerated dimension is a dimension that is derived from fact table.

A dimension such as transaction number, receipt number, Invoice number, Bill No etc. does not have any more associated attributes and hence cannot be designed as a dimension table.

E.g. If you have a dimension that only has Bill number, you would have a 1:1 relation with the fact table.

Do you want to have two tables with a billion rows or one table with a billion rows?

Therefore, this would be a degenerate dimension and Bill number would be stored in the fact table.

### Inferred Dimensions

Early arriving facts, late arriving Dimensions, Inferred Dimensions

While loading fact records, a dimension record may not yet be ready. One solution is to generate a surrogate key with Null for all the other attributes. This should technically be called an inferred member, but is often called an inferred dimension.

*Based on how frequently the data inside a dimension table changes, we can further classify dimension as*

### Unchanged Dimension / Static Dimension Table

Static dimensions are not extracted from the original data source, but are created within the context of the data warehouse. A static dimension can be loaded manually for example with Status codes or it can be generated by a procedure for example Date or Time dimension.

### Slowly Changing Dimension Table

A dimension is considered to be a slowly changing dimension if its attributes changes over a period of time.

E.g. Employee Dimension Table

Attributes like Age, Location changes over a period of time in Employee Dimension table.

- ✓ SCD Type1 - Contains Current data
- ✓ SCD Type2 - Contains Entire History

### Rapidly Changing Dimension

A dimension is considered to be a rapidly changing dimension if one or more of its attributes changes frequently in many rows.

E.g. Currency Dimension

Attribute Currency conversion Rate will be changing frequently in Currency Dimension.

### Types of Fact

#### Additive Fact

Additive facts are facts that can be summed up through all of the dimensions in the fact table.

A sales fact table is a good example for additive facts.

Fact_Sales
Date_Key
Store_Key
Region_Key
Sales
Profit

Date, Store and Region are Dimensions in the fact table, Sales and Profit are facts.

We can measure day wise sales and profit, Store wise sales and Profit, Region wise sales and profit and all will provide us the correct results.

Hence Sales and Profit are additive measures or additive facts.

### **Semi Additive Fact**

Semi Additive facts are facts that can be summed up for some of the dimensions in the fact table, but not the others.

Daily Balances fact table is example for Semi Additive Facts.

Fact_Daily_Balances
Day_Key
Customer_Key
Balance

Day and Customer are the Dimensions in the fact table, Balance is a fact.

We can measure the current balance of the customer and will give the meaningful output.

We can also measure the current balance of Day but it doesn't have any meaning.

### **Non Additive Fact**

Non Additive facts are the facts that cannot be summed up for any of the dimensions present in the fact table.

E.g. Percentages, Ratios are examples for Non Additive facts.

## **Types of Fact Tables**

### **Fact less Fact Table**

A fact table without any facts and has only foreign keys to the dimension tables are known as fact less fact table.

E.g. A fact table which has only product key and date key is a fact less fact table. There are no measures in this table. But still you can get the number of products sold over a period of time.

### **Detailed Fact Table**

A fact tables which stores the details of the transactions is known as detailed fact table.

### **Additive Fact Table or Cumulative Fact Table**

A fact table which stores summary of transactions is known as Additive fact table or Cumulative fact table.

E.g. This fact table may describe the total sales by Product or by Store or by Day.

### Granularity

Granularity refers to the level of detail of the data stored in any tables of a data warehouse.

High granularity refers to data that is at or near the transaction level. Data that is at the transaction level is usually referred to as atomic level data.

Low granularity refers to data that is summarized or aggregated, usually from the atomic level data.

### Surrogate Key

Surrogate Key is sequentially generated meaningless unique number attached with each and every record in a table in any Data Warehouse.

This Surrogate Key is generated using database sequences or Sequence Generator Transformation in Informatica.

### Primary Key Vs. Surrogate Key

Both are primary keys. A natural primary key has some meaning and a surrogate primary key has no meaning.

A Primary key would be anything that has some meaning and will be used to identify the row uniquely.

E.g. EmpNo - A7164500  
A7164501

A Surrogate key is something which is having the sequence generated numbers with no meaning, and just used to identify the row uniquely.

E.g. Sequences 10001  
10002  
10003

### What is ETL?

ETL stands for Extraction, Transformation and Loading.

We can perform this ETL process in two ways.

1. Code Based ETL
2. GUI Based ETL

### Code Based ETL

An ETL application can be developed using programming languages such as SQL and PL/SQL is said to be a code based ETL.

### GUI Based ETL

An ETL application can be designed using simple Graphical User Interface, Point and Click techniques is said to be GUI based ETL.

Examples for GUI based ETL tools

- ✓ Informatica
- ✓ Datastage
- ✓ ODI (Oracle Data Integrator)
- ✓ Ab Initio

### Data Acquisition

Data Acquisition is the process of Extracting, Transforming and Loading the data.

Data Acquisition will be having below processes

- ✓ Data Extraction (E)
- ✓ Data Transformation (T)
- ✓ Data Loading (L)

### Data Extraction

It is the process of reading or extracting the data from various types of source systems.

### Data Transformation

It is the process of transforming the data into the required business format.

The following are the some of the data processing activities takes place.

- ✓ Data Cleansing - Removing unwanted data
- ✓ Data Merging - Joins, Union
- ✓ Data Scrubbing - Deriving New Attributes
- ✓ Data Aggregation - Summaries of the detailed data using aggregate functions.

### Data Loading

It is the process of inserting the data into the destination systems.

There are two types of data loading

- ✓ Initial Load or Full Load
- ✓ Incremental Load Or Delta Load

### Initial Load

It's the process of inserting the data records into an empty target table.  
At first time load all the required data inserts into destination table.

### Incremental Load

It's the process of inserting only new records after initial load or any Load.

## Data Warehouse (DWH) /Business Intelligence (BI)

What is Business Intelligence (BI)?

What is Dimension Table?

What is Fact Table?

What is Dimension?

What is Fact?

What is Star Schema?

Which Schema will give Better Performance, Star Schema or Snowflake Schema?

What is the Schema in Your Project?

## Power BI

What is Power BI?

What is the Different Power BI Software's available?

What is the key difference between the Power BI Desktop and the Power BI service?

What are Building Blocks in Power BI?

What are Tiles in Power BI?

What is difference between PBIX and PBIT file extensions?

## Power BI Desktop

What is Power BI Desktop?

What are the different views or Multiple Views in Power BI Desktop?

## Power Pivot

What is Power Pivot?

Which In-memory Analytics Engine used in Power Pivot?

What are the Different types of Cardinalities we have in Power Pivot?

Can we have more than one active relationship between two tables in data model of power pivot?

What are many-to-many relationships and how can they addressed in Power BI? What is Bidirectional Cross-filtering in Power BI?

What is composite model in Power BI?

What is DAX?

What are the most common DAX Functions used in your project?

What are Row Context and Filter Context?

What is difference between MAX, MAXX and MAXA functions?

What is a calculated column?

What does DATEADD function do?

What does DATEDIFF function do?

How is the FILTER function used?

What is special or unique about the CALCULATE and CALCULATETABLE functions?

What is the common table function for grouping data?

What does Related, Related Table?

What does Calculate, Calculate Table?

What does Summarize, Summarize Columns?

## Power View

What is Data Visualization?

What are the three Edit Interactions options of a visual tile in Power BI Desktop?

State the three edit interaction options in a visual tile in Power BI desktop.

What are the different types of filters in Power BI Reports?

Explain the term Custom Visuals?

How can geographic data be mapped into Power BI Reports?

Why TOP N is not available for the Page level and Report Level Filter?

Explain z-order in Power BI?

What is "What if parameter" in power bi?

What is the need of selection pane in Power BI?

What are slicers?

What are synchronizing slicers?

What is a Bookmark?

What is Filter? How many types? What are they?

How do you compare Target and Actual Values from Power Bi Report?

What is major difference between a Filter and Slicer?

In Power View Advanced Filtering, how to Pass More than 2 Values?

### Power BI Service

What is Row-level Security?

Can the Power BI Report refreshed after they are published to the cloud?

What is incremental refresh?

What is Data refresh?

Is it possible to refresh Power BI Reports after they are published to the cloud?

What gateways are available in Power BI and why use them?

What are the kind of Licenses are available in Power BI?

What are different connectivity modes in Power BI?

What is a Dashboard?

### Others

What are the challenges you faced in your project?

## **Power Query Interview Questions**

**Q) What is Power Query?**

A) Power Query is an ETL Software in Power BI. Power Query is one of the Software in Power BI Desktop which is used for performing ETL Activities or Data Preparation Activities in Power BI.

**Q) Why do we need Power Query?(OR) What we will do with Power Query Software?**

A) Power Query is used to Move the Data that is required for Analysis from Data Sources to Power Pivot. Using Power Query we can connect to almost any Kind of Data Source(Excel, Oracle, SQL Server etc.) to Extract the Data that is Required for Analysis, we can Transform or Prepare the Data as per the Reporting Needs and finally Loads the Data into Power Pivot. In Power Query all these ETL work will be done with just Point and Click with the help of GUI Options.

**Q) Which language is used in Power Query?**

A) M (Mash up) is the Language used in Power Query behind the GUI Options to perform the ETL Activities.

**Q) What is the data destination for Power Queries / Power Query?**

A) All the Transformed Queries or Tables in Power Query will be loaded into Power Pivot. Power Pivot is the Destination for Power Query.

**Q) How to Connect to Data Sources in Power Query?**

A) Using New Source or Recent Sources you can connects to Data Sources in Power Query.

**Q) What is Recent Sources? What it Contain?**

A) Recent Sources is the place where Power Query will Store the Data Sources Connection information. Recent Sources will contain all the Data Sources Connection information to which we connected in Recent Times.

**Q) Name types of Connectivity Modes in Power BI?**

A) Below are the Connectivity Modes in Power BI

1. Import
2. Direct Query
3. Connect Live / Live

Q) Briefly explain about Power Query User Interface (UI)?

The screenshot shows the Power Query Editor window with several key components highlighted:

- Queries [2]** pane (left): Shows two tables: Emp (selected, circled 1) and Dept.
- Data Pane** (center): Displays the contents of the "Emp" table with columns: EMPNO, ENAME, JOB, MGR, HIREDATE. A specific row (row 7, circled 2) is selected.
- Power Query Ribbon** (top): Has a red circle around the "Transform" tab.
- Query Settings pane** (right): Shows "Properties" for the selected "Emp" table and a list of "Applied Steps" (one step is circled 4).
- Formula Bar** (top center): Shows the formula: Table.TransformColumnTypes(#"Promoted Headers", {{"EMPNO", Int64.Type}, {"ENAME", type text}, ...}). A circled 5 points to this bar.

A) Power Query Editor User Interface Contains 5 Major Sections

### 1. Queries Pane

In Queries Pane we will see list of Tables we brought into Power Query. In Power Query Terminology Tables are called as Queries.

### 2. Data Pane / Results Pane

If we select a Table in Queries Pane, the selected table Data Can see in Data Pane. Also when you perform any Transformations on the Data, the transformed Results will be seen in Data Pane.

### 3. Power Query Ribbon

Power Query Ribbon will contain Multiple Tabs and each Tab will contain 100's of GUI Options to Perform ETL Activities.

### 4. Query Settings Pane

Query Settings Pane Contains a Major Section Called Applied Steps. Applied Steps Contains all the ETL Activities we performed in Sequential Order for a Selected Table in Queries Pane.

### 5. Formula Bar

Formula Bar will Show M Language Code for a Selected ETL Step in Applied Steps Section.

**Q) What are some common Power Query Transforms you performed?**

A) Removing the unwanted Columns, Removing Unwanted Rows using Filters, Adding Conditional Columns, Merging the Columns, Splitting the Column, Joining the Tables using Merge Queries, Union the Tables using the Append Queriesetc.

**Q) How to Load Data into Power Pivot from Power Query?**

A) Using Close & Apply GUI Option we can load the Data into Power Pivot.

**Q) Can SQL and Power Query/Query Editor be used together?**

A) Yes. We Can Write SQL Select Statements while Connecting to Data Source in Advanced Options and Once after Bring the Data using Select Statement into Power Query you can perform additional Transformations on the Data in Query Editor using the GUI Options. Hence we can use SQL and Power Query/Query Editor together.

**Q) Where we can see the M Language Code of Each ETL Step we performed in Power Query?**

A) Formula Bar is the Place Where we Can See ETL Code for each Step we Performed in Power Query. By Default Formula Bar will not Show in Power Query Editor User Interface, We need go to View Tab and Select the Formula Bar Check Box to see the Formula Bar in Power Query Editor User Interface.

**Q) What is Advanced Editor in Power Query?**

A) Advanced Editor is the Place where we can see Complete M Language Code for a Selected Table in Power Query.

**Q) Can we Create Table in Power Query? If Yes How can we create?**

A) Yes. Using Enter Data GUI Option we Can Create a Table in Power Query.

**Q) How Can You Change the Data Source Connection information in Power Query?**

A) Using **Data Source Settings** we can Change the Data Source Connection Information.

**Q) How to Join the Tables in Power Query?**

A) Using **Merge Queries** we will join the Tables in Power Query.

**Q) What are the Prerequisites for Joining the Tables in Power Query? And what are Join Types / Join Kinds we have in Power Query?**

**A) The Prerequisites to Join Tables in Power Query is we need a Common Column in the Tables.**

**Below are the Join Types / Join Kinds available in Power Query**

1. Left Outer
2. Right Outer
3. Full Outer
4. Inner
5. Left Anti
6. Right Anti

**Q) How to join 3 Tables in Power Query?**

**A) First Join the Two Tables and the Result Join with Third Table.**

**Q) How to Union the Tables in Power Query?**

**A) Using Append Queries we Can Union the Tables in Power Query. When You Append the Queries default you will get Union ALL Result.**

**Q) How to Get Union Result in Power Query?**

**A) By Default when you append the Queries in Power Query will give Union All Result. To get Union result Once After Appending, Remove Duplicates at Table Level.**

**Q) You need to Import 150 CSV files to Power BI Desktop into One Table. All Files have the Same Structure. What do you do?**

**A) By Taking Folder as Source and Combine Files Option in Power Query we will append all 150 CSV files Data into One Table in Single Go.**

**Q) What is Pivot and UnPivot in Power Query?**

**A) Power Query in Power BI provides very effective functionality to pivot and unpivot columns. For the pivot functionality you turn rows to columns and for the unpivot functionality the inverse is true where columns are transformed into rows.**

## **Q) Difference between Copy, Duplicate and Reference in Power Query?**

A) Let's see difference between Copy, Duplicate and Reference

When you press COPY then you can paste this query many times using PASTE option. As result you receive the same query with same connection and transformations. No connection to initial query.

When you press DUPLICATE you can have only 1 copy of query at one time. To repeat this action you need to press DUPLICATE one more time. As result you receive the same query with same connection and transformations. No connection to initial query.

If you press REFERENCE then it uses previous query as reference and apply all new steps based on reference query. Once you change initial query then it will automatically apply to dependent query.

## **Q) What is Query Dependencies Option in Power Query?**

A) It is a Table View or Graphical View of all your tables/queries, showing the linkages between the tables/queries.

## **Q) What is Enable Load and Include in Report Refresh Query option?**

A) Enable Load will help us to stop loading the unwanted tables into Power Pivot. By default all queries are enabled to Load into the Power Pivot model. Simply uncheck enable load option for queries that are not required in the Power Pivot Model.

Include in Report Refresh option will help us to stop refreshing the Static Dimension Tables when you refresh the Data Model. By Default all the Tables will refresh when you refresh the Data Model / Dataset. For Static Tables / Static Dimension Table Simply uncheck this Include in Report Refresh option.

## **Q) Can we Aggregate the Data in Power Query, if yes, How to Aggregate the Data in Power Query?**

A) Yes we can aggregate the Data in Power Query using “**Group By**” GUI Option.

## **Q) What is Columns from Examples Option in Power Query? How it will help us?**

A) “Column from Examples” Option will help us to create a New Conditional Column in the table by using “All Columns” or “Selected Columns”as source in the Table.

**Q) When we will use Custom Column Option?**

A) Custom Column Option will help us to Derive New Conditional Columns by writing our Own M Language Code.

**Q) What is user-defined or custom function in Power Query?**

A) In Power Query we have 100's of inbuiltfunctions available we still might need to create our own custom function to tackle repetitive tasks. Instead of writing the same code over and over again, create a user defined or custom function and save the time by using it.

**Q) What is Invoke Custom Function Option? When we will use it?**

A) Invoke Custom Function is used for calling the Custom Functions created by Developers.

**What is Query Folding in Power Query? (OR) What is Query Collapsing in Power Query?**

A) Query Folding is a feature in Power BI that is designed for Query Optimization. Query Folding is the ability for a Power Query to generate a Single Query statement that Retrieves and Transforms source data on Data Source itself. When you define Transformations on the Data in Power Query, using Query Folding Option, it is possible that those Transformations are sent back to the source to improve performance. Query Folding will not support for all the Data Sources.

**Q) What is Parameters in Power Query and how they are helpful?**

A) Power BI allows you to use Parameters to make your reports dynamic. Power Query parameters are used for applying changes in the data sourceconnection informationor Query Filters or data transformation dynamically.