CONTENT

CHAPTER 1: INTRODUCTION OF POWER BI 1

ARCHITECTURE OF POWER BI 2
BUILDING BLOCKS / COMPONENTS OF POWER BI 3
VISUALIZATIONS 4
DATASETS 5
REPORT 5
DASHBOARD 6
COMPONENTS OF POWER BI DESKTOP 8
CREATING THE FIRST REPORT USING POWER BI DESKTOP 8
RECENT SOURCES 15
ENTER DATA 15
NEW PAGE 16
NEW VISUAL 16
BUTTONS 17
TEXTBOX 18
IMAGES 18
SHAPES 19
SWITCH THEME 19
MANAGE RELATIONSHIP 20
PUBLISH 20
REFRESH 21

CHAPTER 3 : POWER BI DATA SOURCES 23

CONNECTING WITH EXCEL 23
CONNECTING WITH WEB 28
CONNECTING WITH MYSQL 30
CONNECTING WITH SQL SERVER 32

CHAPTER 4 : POWER BI REPORTS 37

CREATING A REPORT USING POWER BI 37
STEPS FOR CREATING POWER BI REPORTS 38
CHART FEATURES 39
FORMATS 39
ANALYTICS 40
AXIS , Y - AXIS & LEGEND 41
TYPES OF CHARTS/VISUALIZATIONS IN POWER BI 42
LINE CHART 42
AREA CHART 43
STACKED AREA CHART 44

PIE CHART *44*
DONUT CHART *45*
TREEMAP *45*
SCATTER CHART *46*
FUNNEL *46*

CHAPTER 5: POWER BI - DATA TRANSFORMATION 48

STEPS TO OPEN POWER QUERY EDITOR IN POWER BI 48
DATA TRANFORMATION FEATURES IN POWER BI 51
CHOOSE COLUMNS 51
REMOVE COLUMNS 51
KEEP ROWS 52
REMOVE ROWS 52
SPLIT COLUMN 53
GROUP BY 53
DATA TYPES 54
USE FIRST ROW AS HEADERS 54
REPLACE VALUES 55
MERGE QUERIES 55
APPEND QUERIES 56

CHAPTER 6: POWER BI – ADVANCE TRANSFORMATIONS 59

GROUP BY 59
USE FIRST ROW AS HEADER 59
TRANSPOSE 60
REVERSE ROWS 60
COUNT ROWS 61
REPLACE VALUES 61
PIVOT COLUMN 62
UNPIVOT COLUMNS 63
CONVERT TO LIST 63
FORMAT 64
EXTRACT 64
PARSE 65
STATISTICS 65
STANDARD 66
SCIENTIFIC 66
TRIGNOMETRY 67
ROUNDING 67
INFORMATION 68
DATE & TIME COLUMN 68
DURATION 69

CHAPTER 7: CALCULATED COLUMNS IN DATA TRANSFORMATION 71

CUSTOM COLUMN 71
INVOKE CUSTOM FUNCTION 72
CONDITIONAL COLUMN 73
INDEX COLUMN 74
DUPLICATE COLUMN 74

CHAPTER 8: DATA ANALYSIS EXPRESSION (DAX) 76

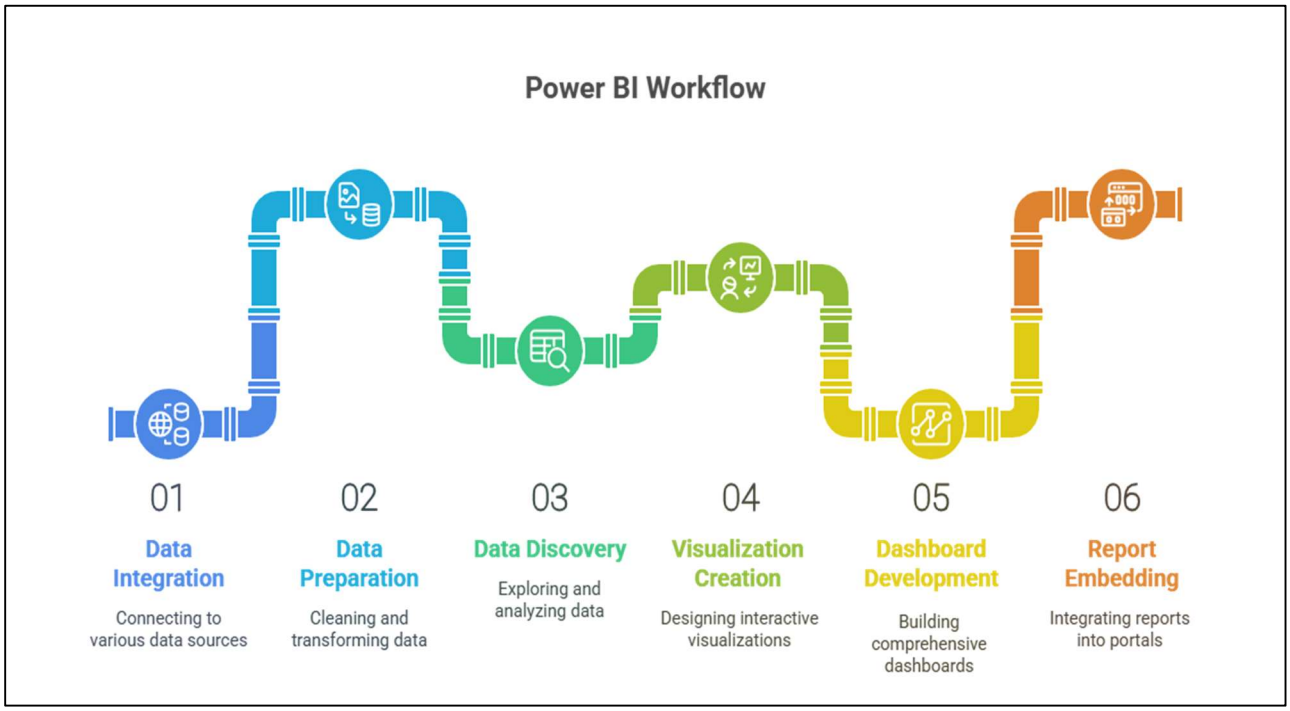
INTRODUCTION TO DAX 77
TYPES OF POWER BI DAX FUNCTIONS 78
USES OF DAX FUNCTIONS IN POWER BI 78
AGGREGRATE FUNCTIONS OF DAX 79
FILTER AND VALUE FUNCTIONS (TABLE FUNCTIONS) 79
LOGICAL FUNCTIONS 82
INFORMATION FUNCTIONS 83
MATH AND TRIG FUNCTIONS 84
TEXT FUNCTIONS 87
DATE AND TIME FUNCTIONS 89
USEFUL DATE FUNCTIONS 91
TIME INTELLIGENCE FUNCTIONS 92
FUNCTIONS 94
TABLE FUNCTIONS IN DAX 95
UNION, INTERSECT, EXCEPT 96
ITERATOR FUNCTIONS IN DAX 96

CHAPTER 9: DEPLOYING, SECURING, AND UPDATING POWER BI REPORTS 102

COMPONENTS REQUIRED FOR POWER BI CLOUD SERVER 103
COMPONENTS OF POWER BI APP 103
MAJOR FUNCTIONALITY OF POWER BI APPS 104
CONFIGURING A DATA GATEWAY IN POWER BI SERVICE 110

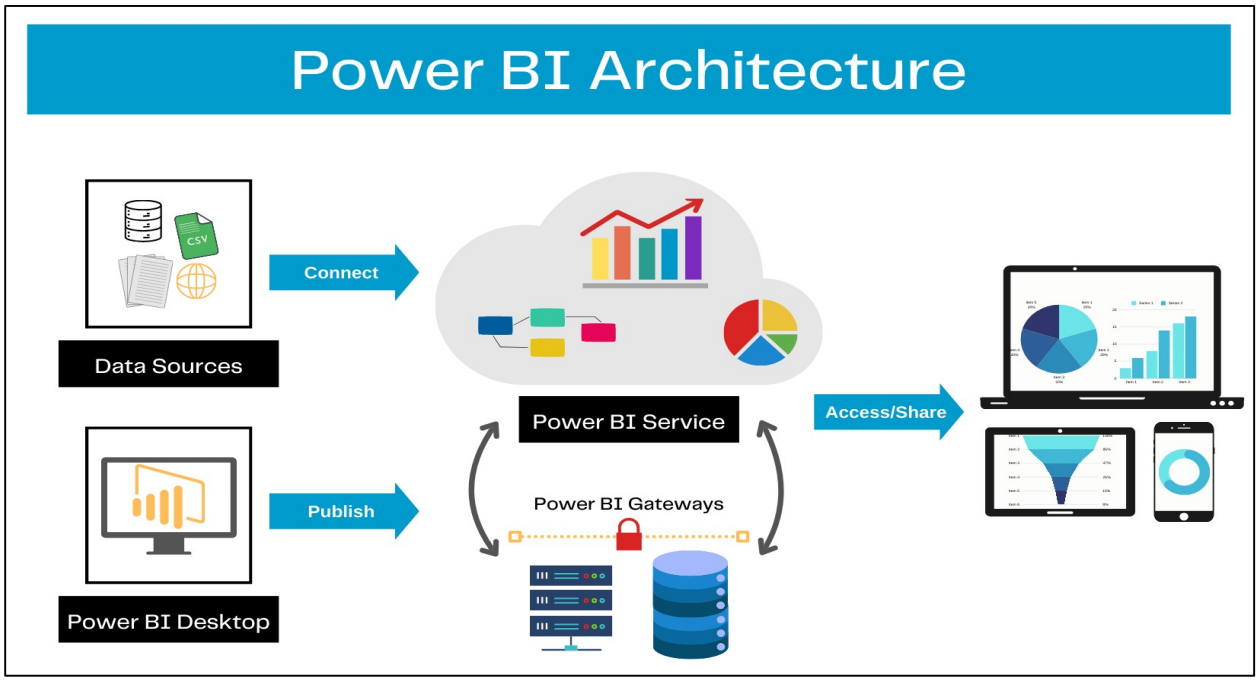
CHAPTER 1: INTRODUCTION OF POWER BI

- Power BI is a business analytics service provided by Microsoft.
- It provides interactive visualizations with self-service business intelligence capabilities, where end users can create reports and dashboards by themselves, without having to depend on information technology staff or database administrators.
- Power BI also gives you cloud-based BI services, known as “Power BI Services”, along with a desktop-based interface, called “Power BI Desktop”.
- It offers data warehouse capabilities, including data preparation, data discovery and interactive dashboards. In March 2016, Microsoft released an additional service called Power BI Embedded on its Azure cloud platform which enables the user to analyses data easily, perform various ETL operations and deliver reports with Power BI.
- Power BI gateways let you connect with SQL Server databases, Analytical Services, and many other data sources to your dashboard in Power BI and reporting portals, embed Power BI reports and dashboards to give you a unified experience. The image below shows Power BI’s general workflow.



ARCHITECTURE OF POWER BI

The following image shows Power BI's architecture.



Power BI's architecture has three phases:

The first two phases partially use ETL (Extract, Transform and Load) to handle the data. Let us take a look at these phases one by one:

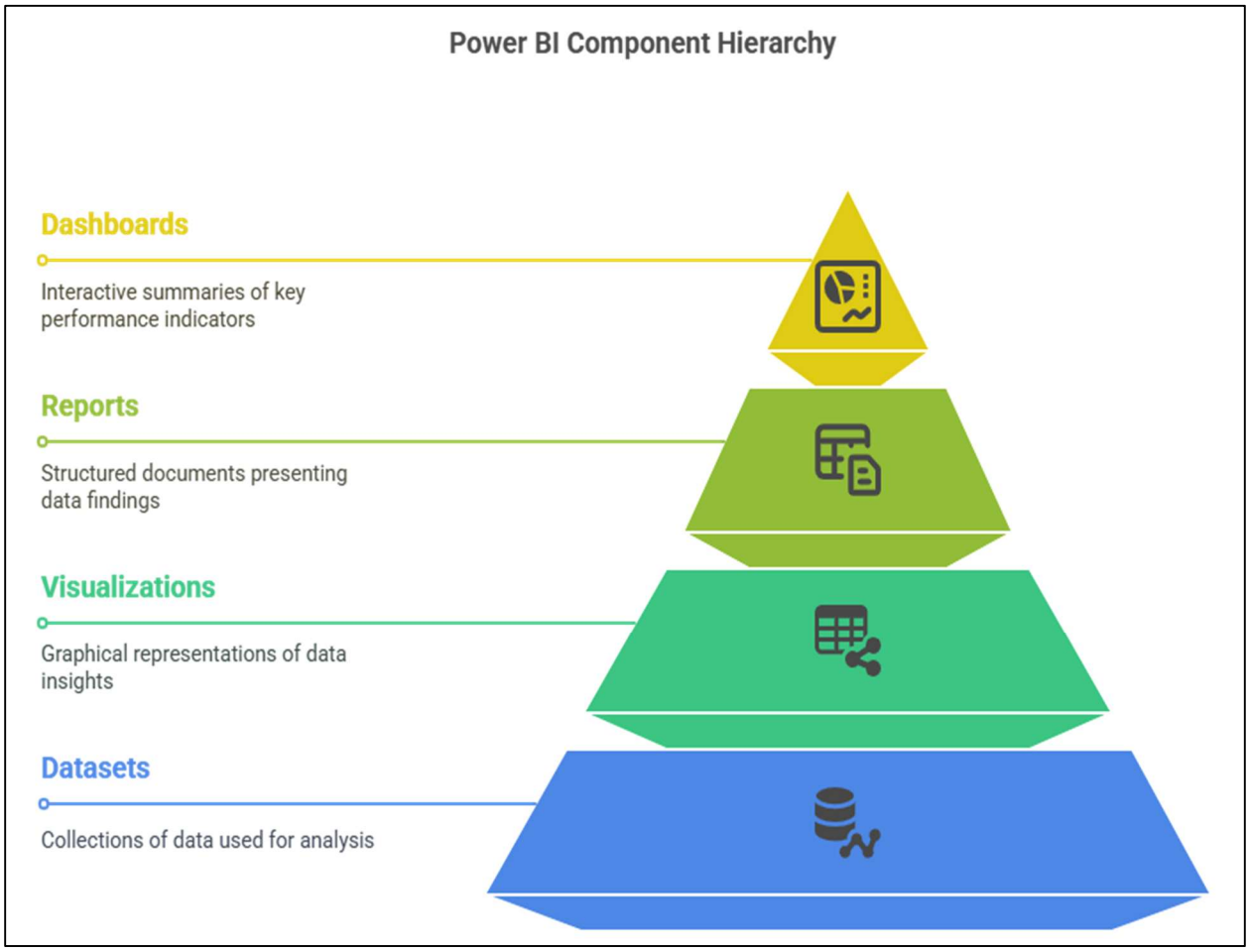
PHASE 1: DATA INTEGRATION /IMPORTING OF DATA
An organization can be required to deal with data that comes from different sources. The data from data sources can be in different file formats. The data is first extracted from different sources which can be your different servers or databases etc. This data is then integrated in a standard format and then stored at a common area called as staging area.
PHASE 2 : DATA PROCESSING / MANIPULATION OF DATA
The integrated data is still not ready for visualization because the data needs processing before it can be presented. This data is pre-processed or cleaned. For example, missing values or redundant values are removed from the data set. After the data is cleaned, business rules are applied to the data and it is transformed into presentable data. This data is then loaded into the Data Warehouse.
PHASE 3 : DATA PRESENTATION / VISUALIZATION
So once the data is loaded and processed now it can be visualized much better with use of various visualizations that Power BI has to offer. Use of reports, dashboards help one represent data in more intuitive manner. These visuals, reports help business end users to take business decisions based on the insights.

BUILDING BLOCKS / COMPONENTS OF POWER BI

The basic building blocks/components of Power BI are the following:

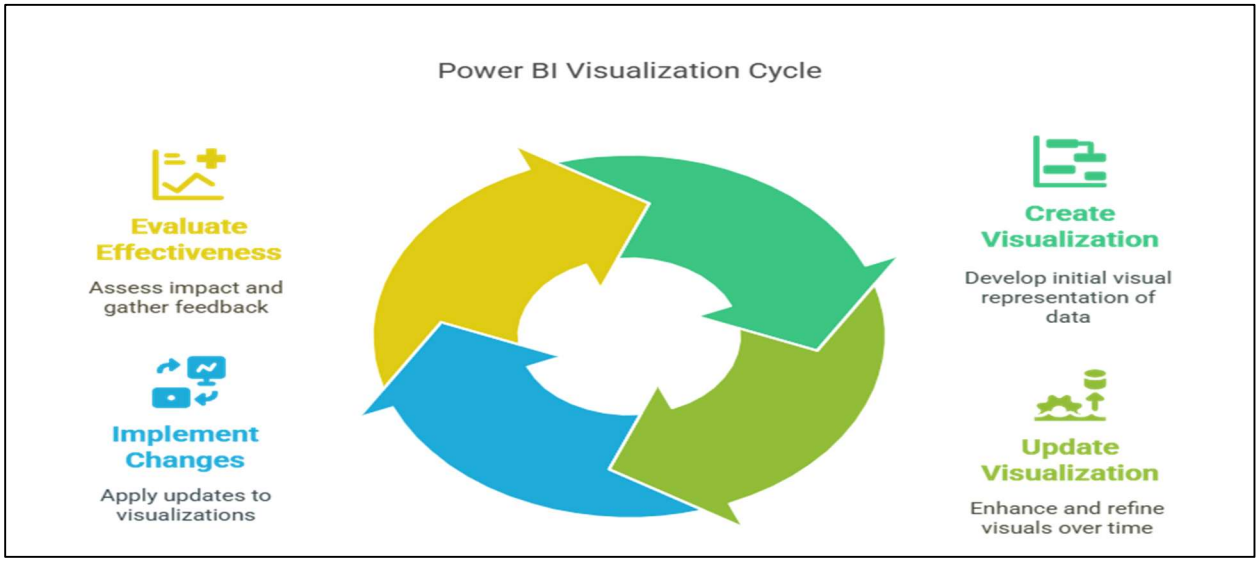
- Visualizations
- Datasets
- Reports
- Dashboards

Note: Each component in Power BI plays a crucial role in transforming raw data into actionable insights. Datasets serve as the foundation, visualizations help interpret data, reports organize insights across multiple visuals, and dashboards provide a high-level summary for quick decision-making.



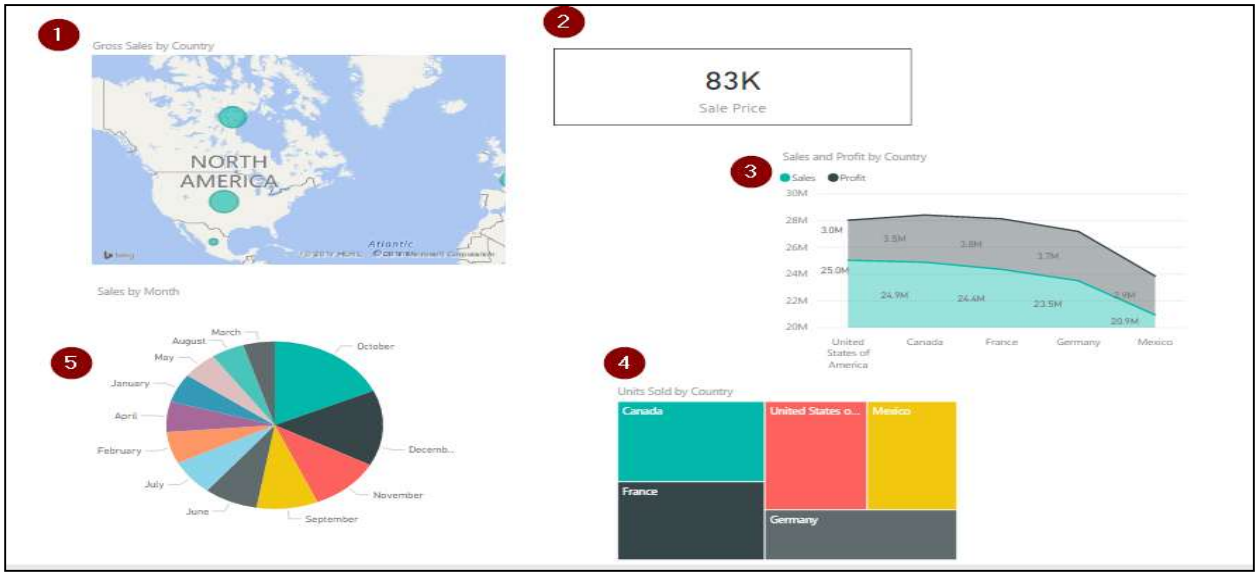
VISUALIZATIONS

A visual representation of data is called visualization. For example, a chart or a graph can be used to represent data visually. Power BI gives you different visualization types, which keep getting updated with time. Following image shows some commonly used visualizations:



Types of Visualization:

1. Map representation
2. Card visualization
3. Stacked area chart
4. Tree map
5. Pie chart



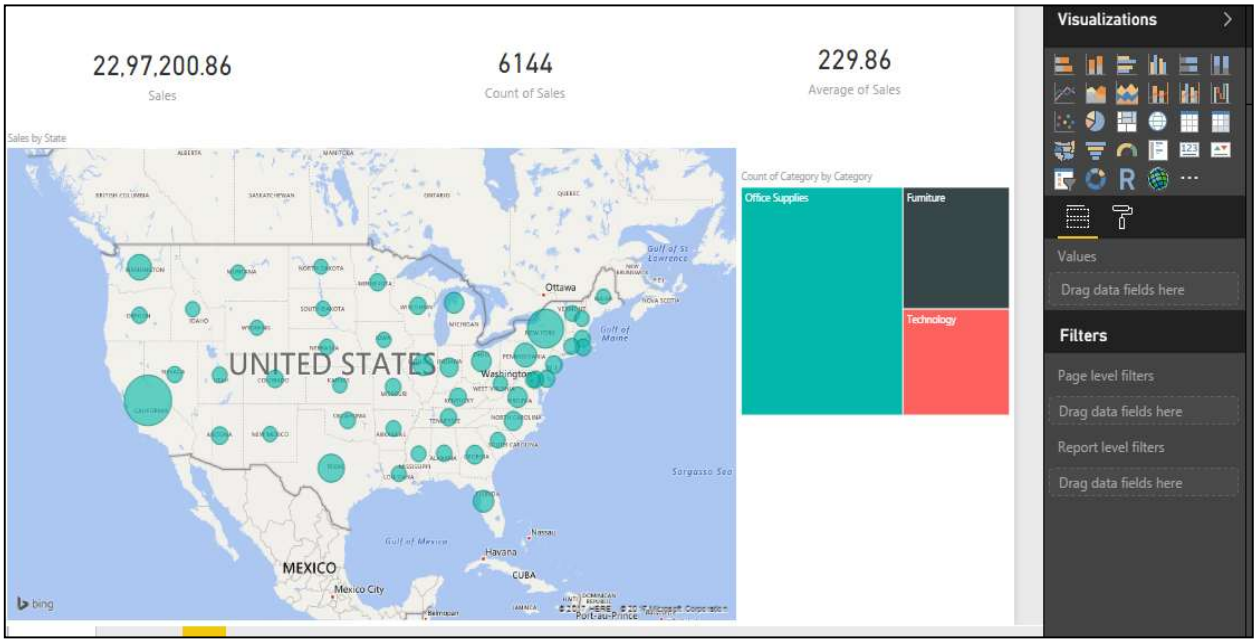
Visualizations can be simple or they can be visually complex. However, visualization aims at presenting data in such a way that it gives you more insights in the context, which is otherwise difficult to discern from simple data files.

DATASETS

- We know that data-set is nothing but a collection of data or information. Power BI harnesses this data to create visualizations. It can be a simple data set or a combination of many different sources, which can be filtered and combined to provide a different data set altogether.
- For example, you can pull together data from many different sources like different database fields, an excel table, and online results of some email campaign to create the data set
- With the data set ready, you are free to create visualizations and display different portions of that data set in different ways, and with this, you gain insights.

REPORT

- A collection of visualizations that appear together on one or more pages is a report in Power BI. It is a collection of items that are related to one another.
- You can create visualizations, on multiple different pages if necessary, and arrange them in a way that suits your story best. The image below shows a sample report.

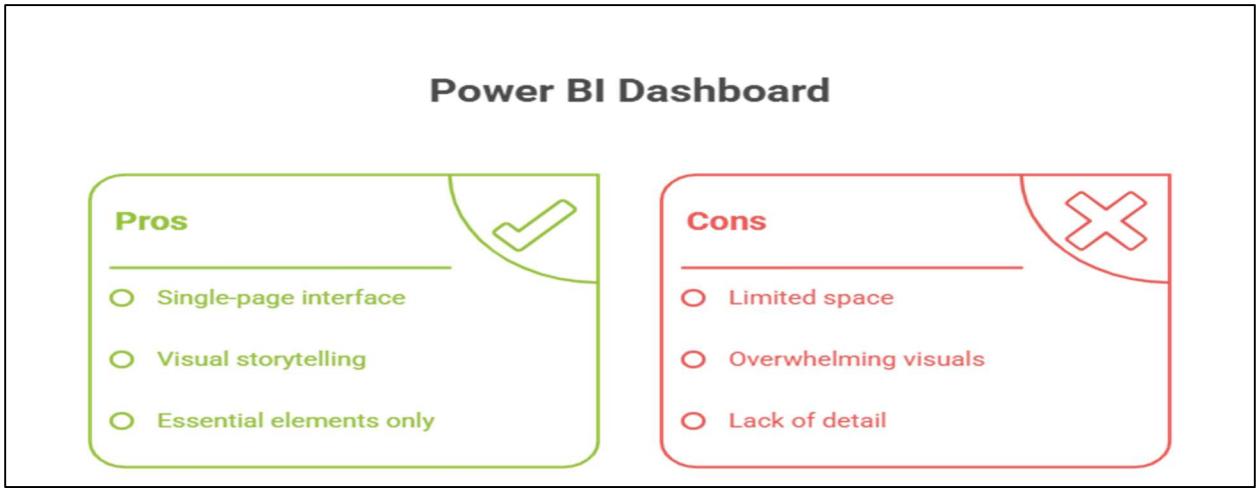


DASHBOARD

- A Power BI dashboard is a single page interface. It is 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.

- Power BI gives you the freedom to move or arrange tiles, so you can present the data the way you want to, even while you're creating a report or a dashboard. You can make the tiles bigger, change their height or width, and snuggle them up to other tiles the way you want.

- So this was about Power BI's building blocks, now I am going to take this Power BI tutorial a step further with a demonstration of creating a simple report using Power BI. However, there are few prerequisites to get started. First of all you need a 'Power BI Desktop' installed on your system, this is an interface where you can create reports. It can be downloaded for free.



Note: A Power BI Dashboard provides a single-page view of key insights using interactive visuals. It helps users monitor performance and make data-driven decisions quickly. Well-designed dashboards present only the most relevant and impactful information.

QUESTIONS

1.What is Power BI and how does it help organizations in decision-making?
2.Explain the three main stages in the architecture of Power BI.
3.What is the role of Data Processing in Power BI, and which tool is used for data transformation?
4.Differentiate between Reports and Dashboards in Power BI.
5.What is the difference between Power BI Desktop and Power BI Service?

SUMMARY

Power BI is a comprehensive **business analytics platform developed by Microsoft** that empowers organizations to visualize their data, discover insights, and make informed business decisions. It offers a suite of tools and services that connect to various data sources, transform raw information into meaningful insights, and present it in the form of interactive visualizations, dashboards, and reports.

Architecture of Power BI

1. **Data Integration:**
   In this phase, data is extracted from multiple internal or external sources—such as Excel files, databases, web APIs, or cloud applications—and combined into a unified format. This step ensures all relevant data is available for analysis. The collected data is temporarily stored in a staging area before transformation.
2. **Data Processing:**
   Once the data is integrated, it must be cleaned and transformed. This involves handling missing values, removing duplicates, correcting errors, and applying business rules to make the data consistent and usable. Power BI uses the **Power Query Editor** to perform these transformations efficiently.
3. **Data Visualization:**
   After data preparation, users can build visual reports and dashboards using **Power BI Desktop** or **Power BI Service**. These visualizations convert numbers into actionable insights, helping decision-makers quickly interpret trends and relationships within the data.

Core Components of Power BI

- **Visualizations:** Graphical representations such as charts, maps, cards, and gauges that help communicate data effectively.
- **Datasets:** The core data collections used for analysis, which can come from single or multiple sources.
- **Reports:** Multi-page documents containing related visualizations that provide detailed insights into specific aspects of a dataset.
- **Dashboards:** Single-page, interactive summaries (often called canvases) that display key metrics and KPIs using pinned visuals from reports.

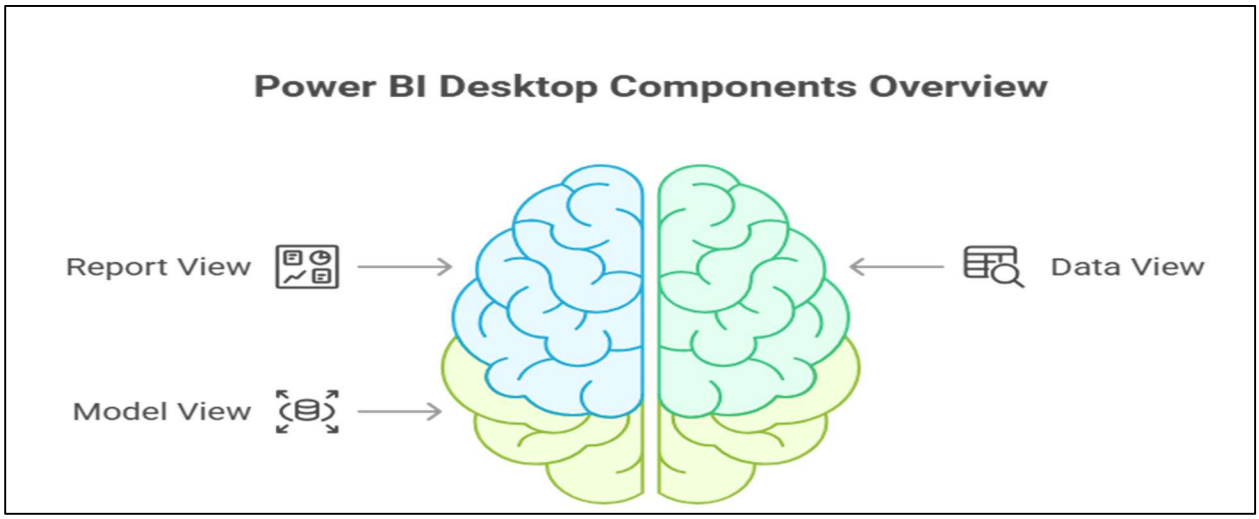
Power BI Interfaces and Ecosystem

- **Power BI Desktop:** A desktop application used for building data models, creating reports, and performing transformations.
- **Power BI Service (Cloud):** An online SaaS platform for sharing, publishing, and collaborating on reports and dashboards.

CHAPTER 2: POWER BI DESKTOP COMPONENTS

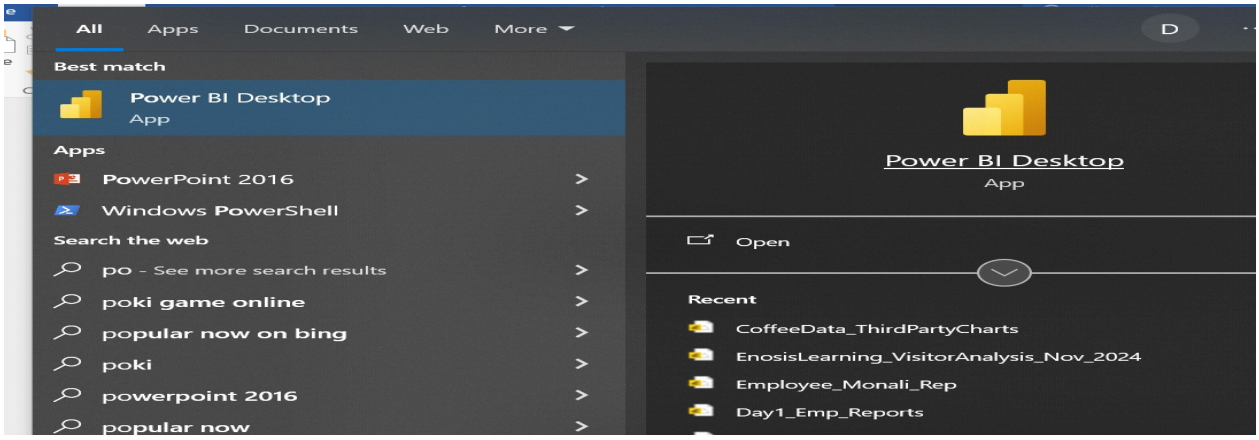
COMPONENTS OF POWER BI DESKTOP

In this chapter we will explore different components of Power BI Desktop

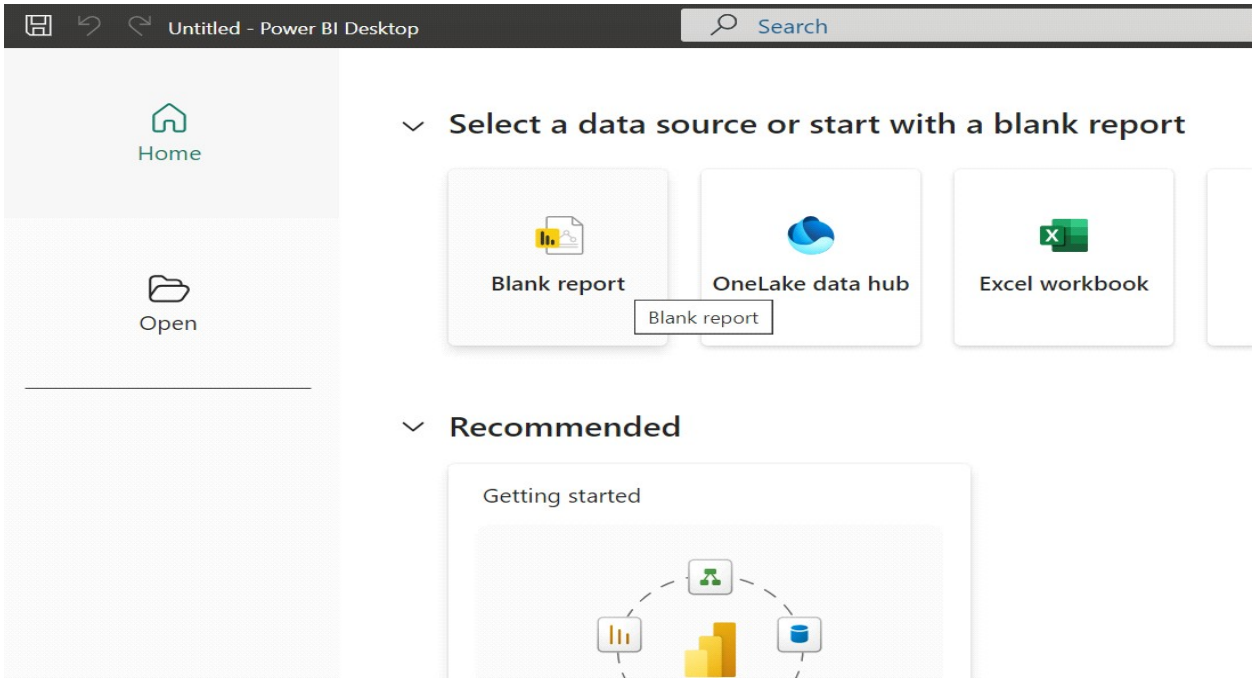


CREATING THE FIRST REPORT USING POWER BI DESKTOP

STEP 1 : OPEN POWER BI DESKTOP AND CREATE A NEW POWER BI REPORT

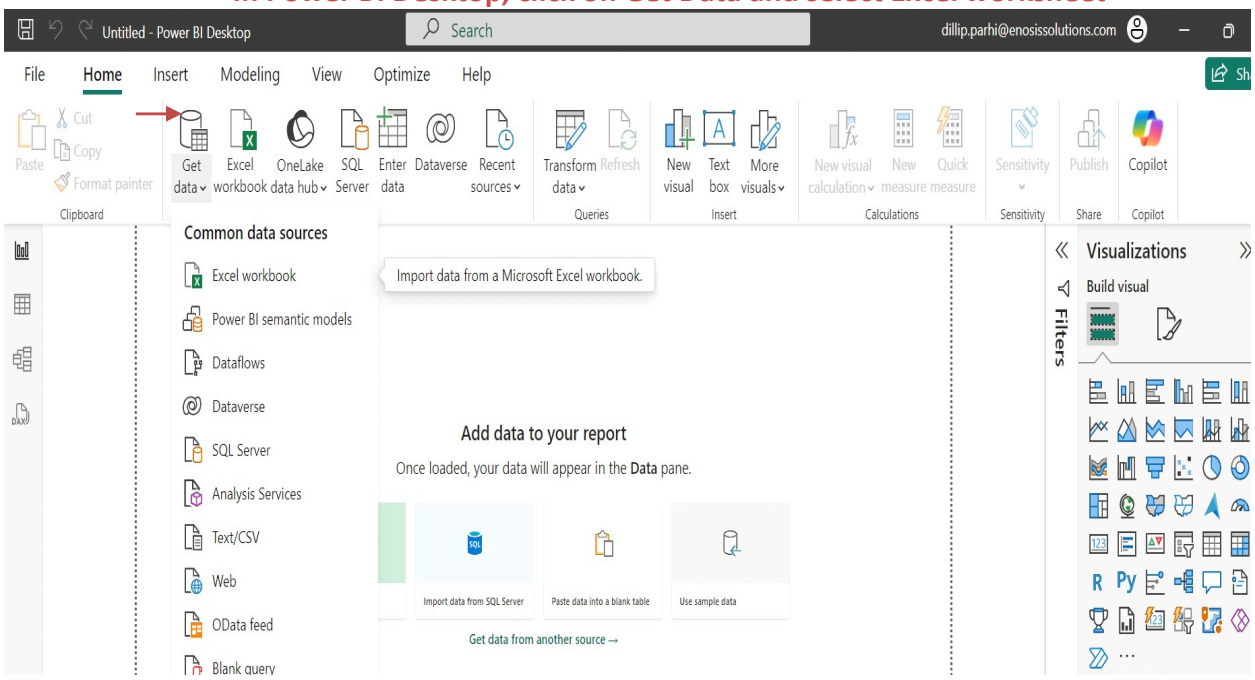


- **Click on New Report in the Power BI Desktop IDE(Editor)**

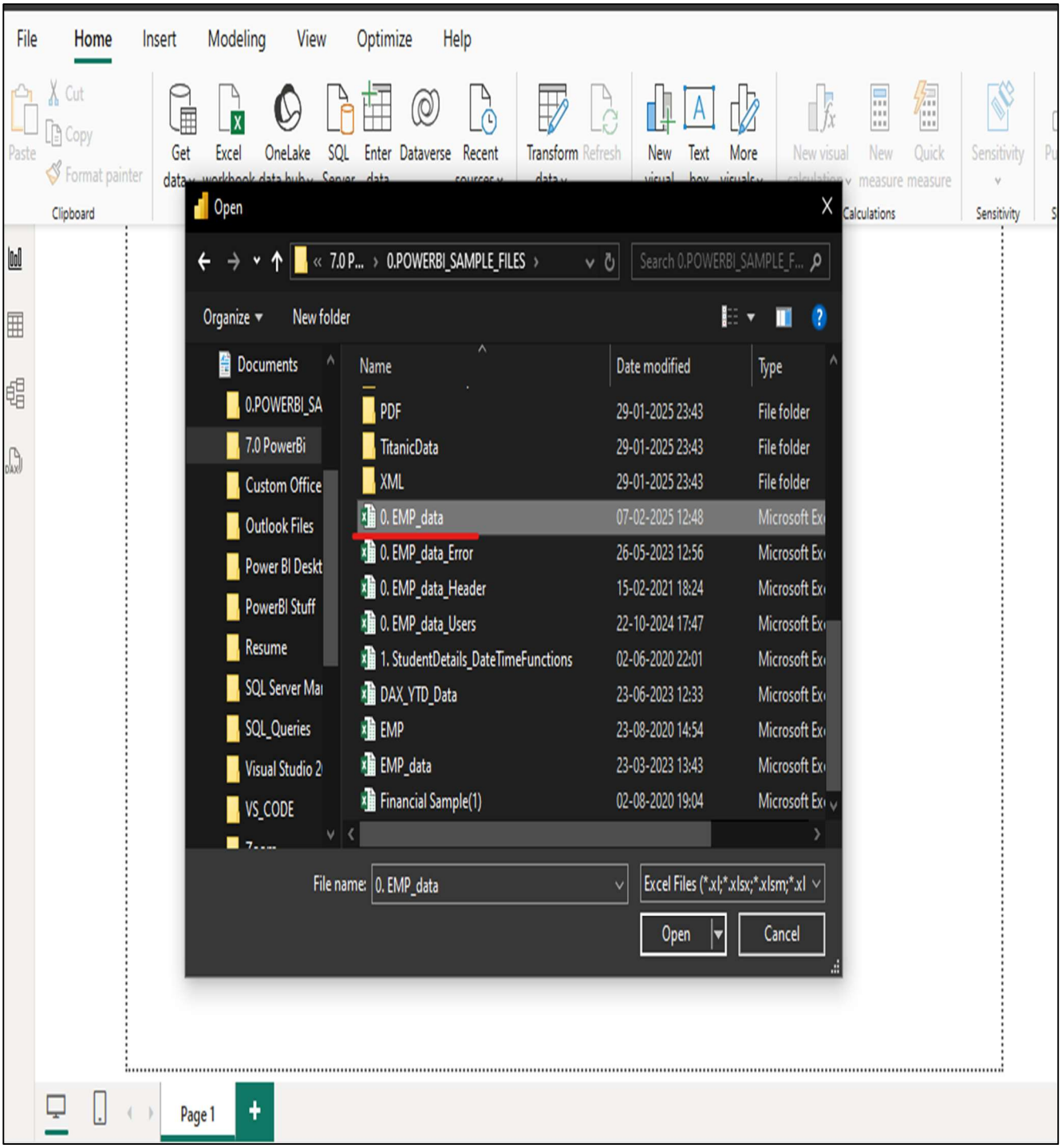


STEP 2 : IMPORT DATA AND LOAD THE DATA INTO POWER BI

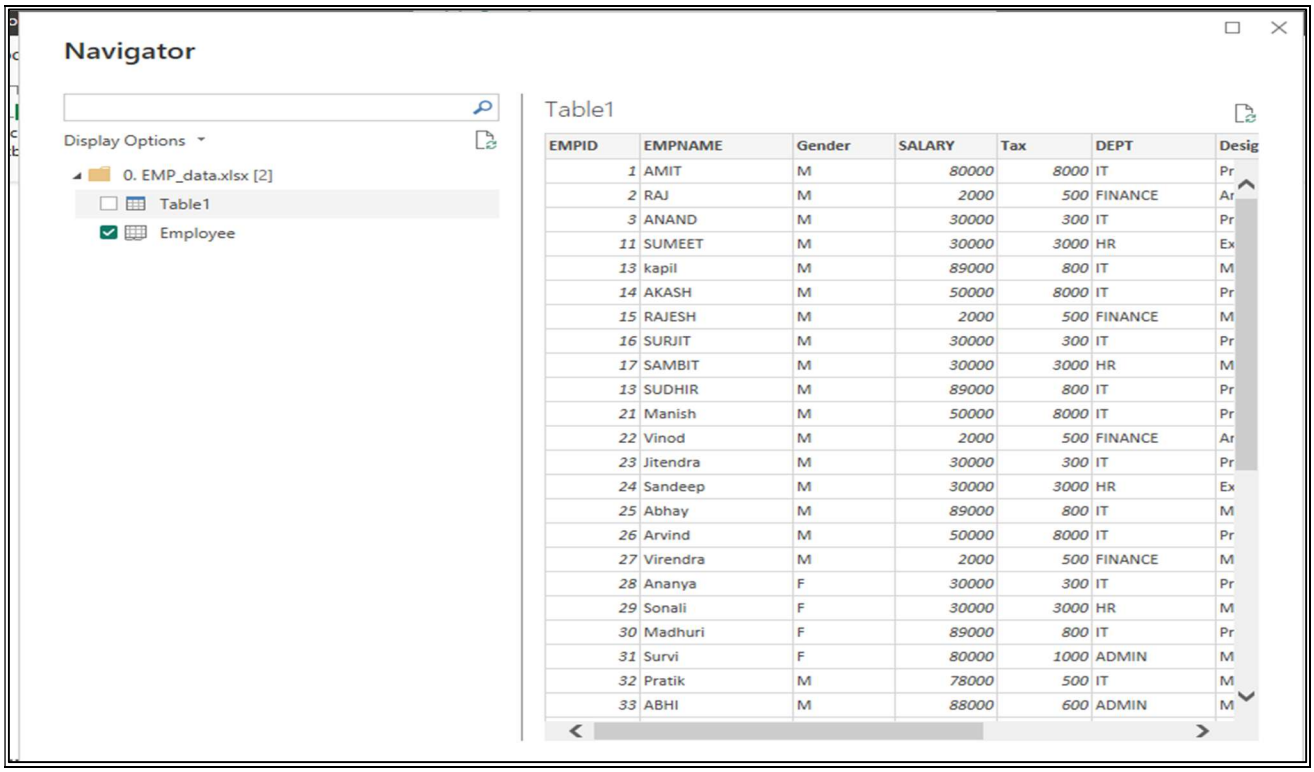
In Power BI Desktop, click on Get Data and select Excel worksheet



Browse to the path and select the file (Excel file)

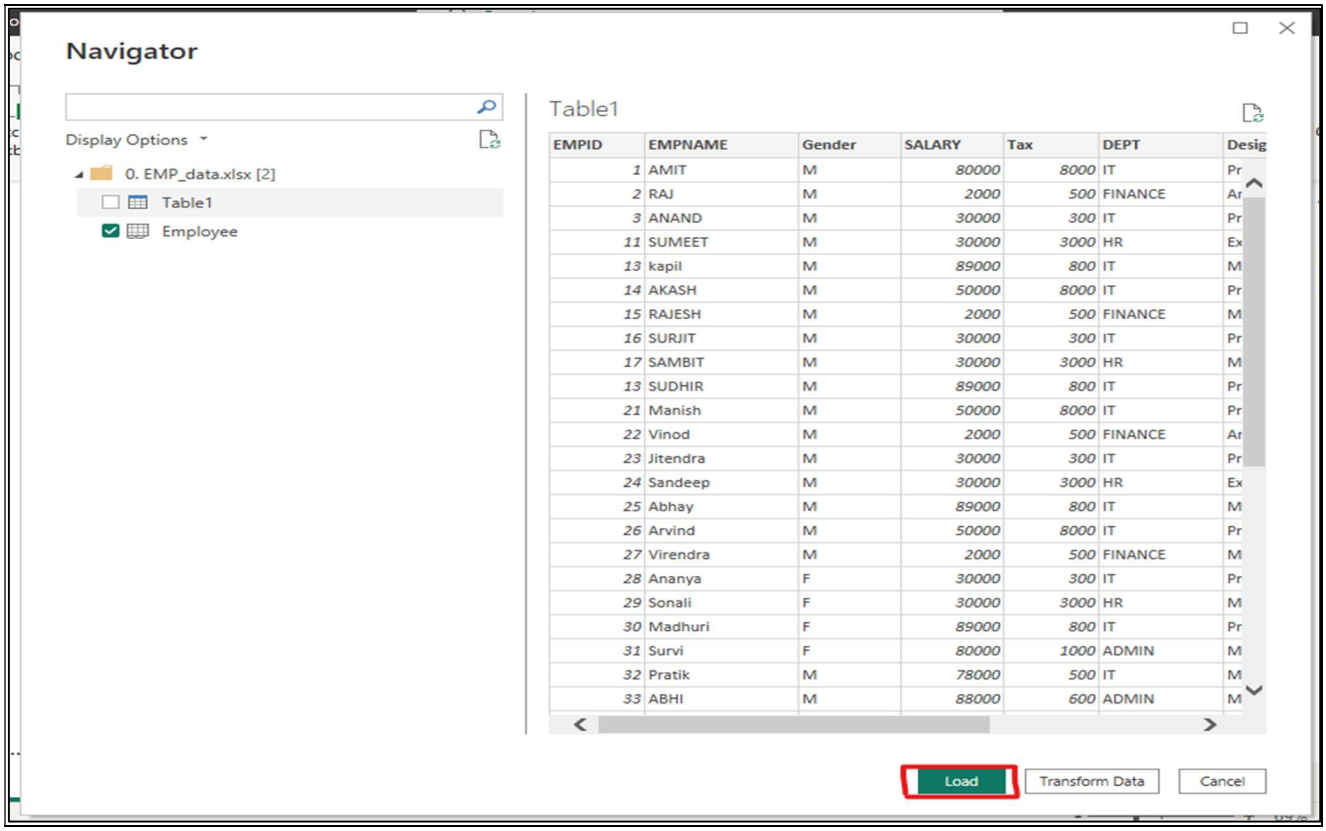


- **Select the sheet from the excel path**
- It will display the sheets present in the Excel, sheet the Employee sheet data



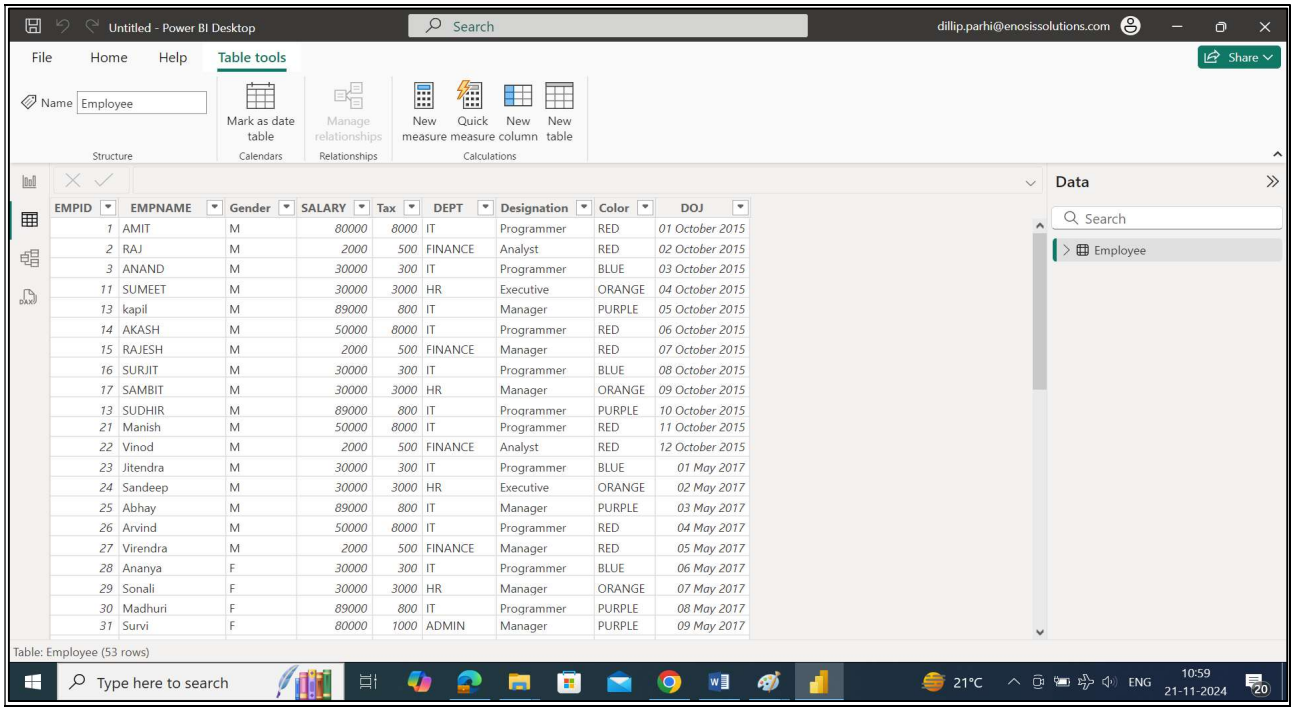
EMPID	EMPNAME	Gender	SALARY	Tax	DEPT	Desig
1	AMIT	M	80000	8000	IT	Pr
2	RAJ	M	2000	500	FINANCE	Ar
3	ANAND	M	30000	300	IT	Pr
11	SUMEET	M	30000	3000	HR	Ex
13	kapil	M	89000	800	IT	M
14	AKASH	M	50000	8000	IT	Pr
15	RAJESH	M	2000	500	FINANCE	M
16	SURJIT	M	30000	300	IT	Pr
17	SAMBIT	M	30000	3000	HR	M
13	SUDHIR	M	89000	800	IT	Pr
21	Manish	M	50000	8000	IT	Pr
22	Vinod	M	2000	500	FINANCE	Ar
23	Jitendra	M	30000	300	IT	Pr
24	Sandeep	M	30000	3000	HR	Ex
25	Abhay	M	89000	800	IT	M
26	Arvind	M	50000	8000	IT	Pr
27	Virendra	M	2000	500	FINANCE	M
28	Ananya	F	30000	300	IT	Pr
29	Sonali	F	30000	3000	HR	M
30	Madhuri	F	89000	800	IT	Pr
31	Survi	F	80000	1000	ADMIN	M
32	Pratik	M	78000	500	IT	M
33	ABHI	M	88000	600	ADMIN	M

View the data and Click on Load to load the data into Power BI



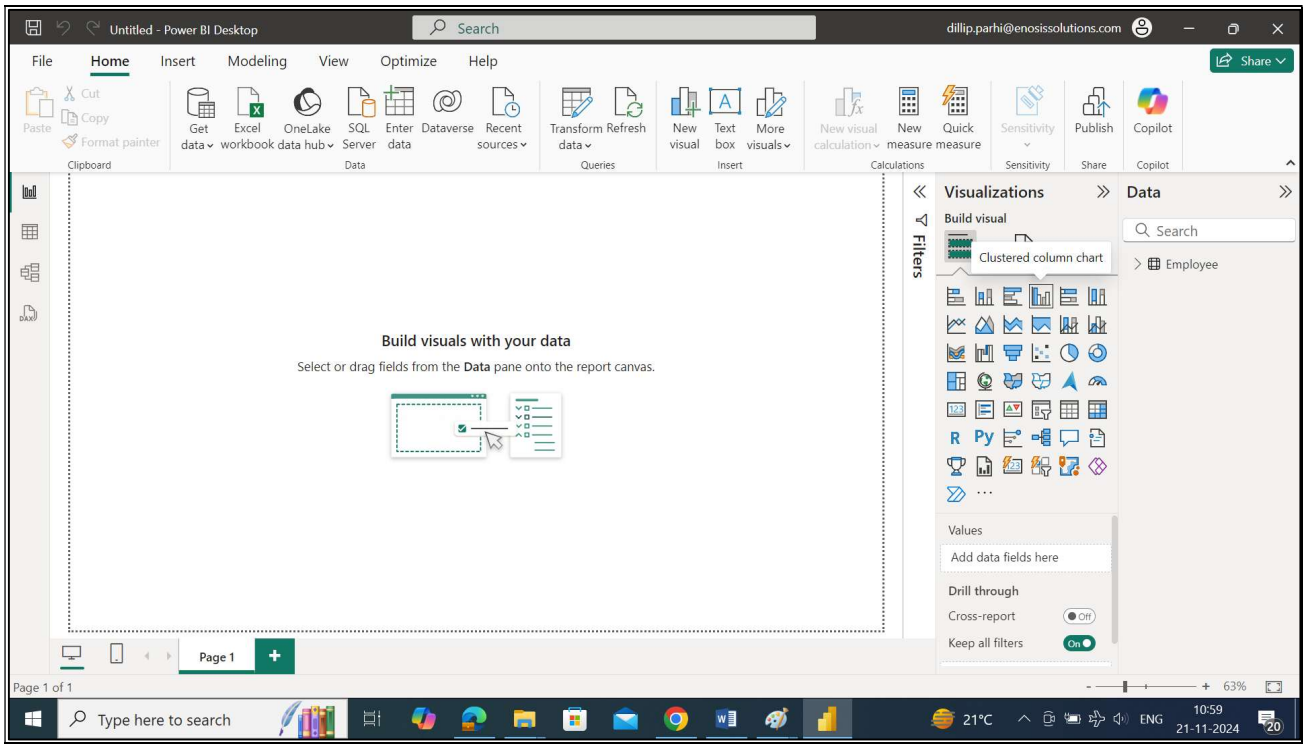
EMPID	EMPNAME	Gender	SALARY	Tax	DEPT	Desig
1	AMIT	M	80000	8000	IT	Pr
2	RAJ	M	2000	500	FINANCE	Ar
3	ANAND	M	30000	300	IT	Pr
11	SUMEET	M	30000	3000	HR	Ex
13	kapil	M	89000	800	IT	M
14	AKASH	M	50000	8000	IT	Pr
15	RAJESH	M	2000	500	FINANCE	M
16	SURJIT	M	30000	300	IT	Pr
17	SAMBIT	M	30000	3000	HR	M
13	SUDHIR	M	89000	800	IT	Pr
21	Manish	M	50000	8000	IT	Pr
22	Vinod	M	2000	500	FINANCE	Ar
23	Jitendra	M	30000	300	IT	Pr
24	Sandeep	M	30000	3000	HR	Ex
25	Abhay	M	89000	800	IT	M
26	Arvind	M	50000	8000	IT	Pr
27	Virendra	M	2000	500	FINANCE	M
28	Ananya	F	30000	300	IT	Pr
29	Sonali	F	30000	3000	HR	M
30	Madhuri	F	89000	800	IT	Pr
31	Survi	F	80000	1000	ADMIN	M
32	Pratik	M	78000	500	IT	M
33	ABHI	M	88000	600	ADMIN	M

STEP 3 : VIEW THE DATA IN POWER BI BY GOING TO THE DATA SECTION IN POWERBI

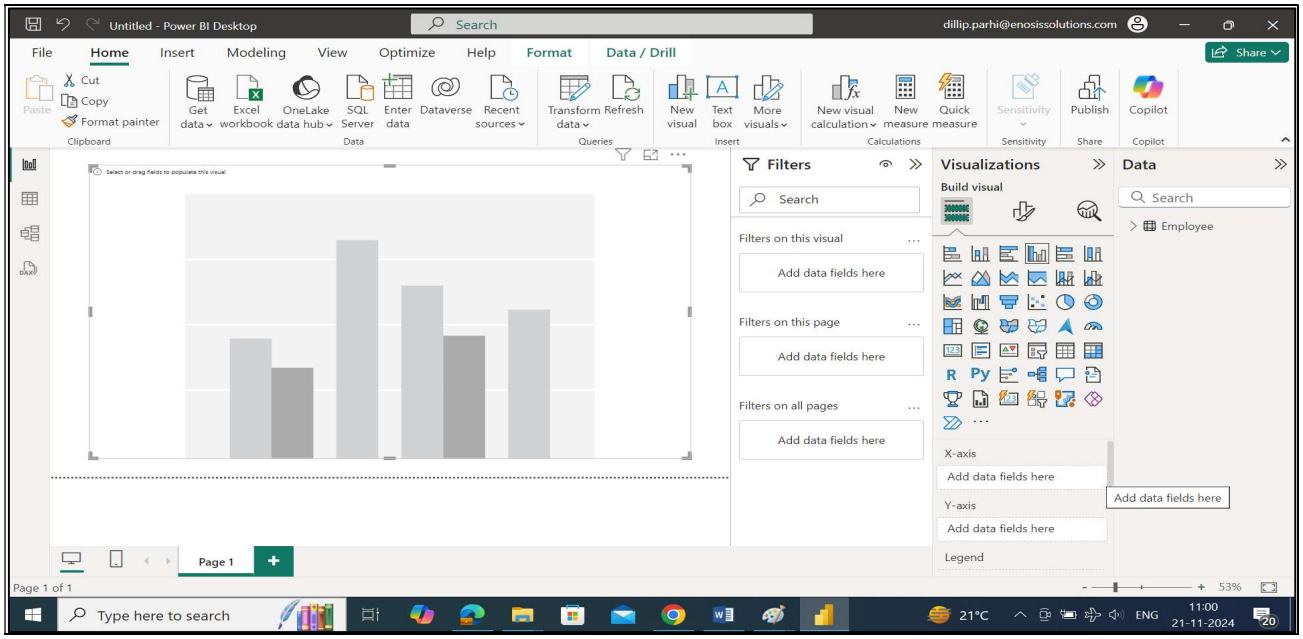


EMPID	EMPNAME	Gender	SALARY	Tax	DEPT	Designation	Color	DOJ
1	AMIT	M	80000	8000	IT	Programmer	RED	01 October 2015
2	RAJ	M	2000	500	FINANCE	Analyst	RED	02 October 2015
3	ANAND	M	30000	300	IT	Programmer	BLUE	03 October 2015
11	SUMEET	M	30000	3000	HR	Executive	ORANGE	04 October 2015
13	kapil	M	89000	800	IT	Manager	PURPLE	05 October 2015
14	AKASH	M	50000	8000	IT	Programmer	RED	06 October 2015
15	RAJESH	M	2000	500	FINANCE	Manager	RED	07 October 2015
16	SURJIT	M	30000	300	IT	Programmer	BLUE	08 October 2015
17	SAMBIT	M	30000	3000	HR	Manager	ORANGE	09 October 2015
13	SUDHIR	M	89000	800	IT	Programmer	PURPLE	10 October 2015
21	Manish	M	50000	8000	IT	Programmer	RED	11 October 2015
22	Vinod	M	2000	500	FINANCE	Analyst	RED	12 October 2015
23	Jitendra	M	30000	300	IT	Programmer	BLUE	01 May 2017
24	Sandeep	M	30000	3000	HR	Executive	ORANGE	02 May 2017
25	Abhay	M	89000	800	IT	Manager	PURPLE	03 May 2017
26	Arvind	M	50000	8000	IT	Programmer	RED	04 May 2017
27	Virendra	M	2000	500	FINANCE	Manager	RED	05 May 2017
28	Ananya	F	30000	300	IT	Programmer	BLUE	06 May 2017
29	Sonali	F	30000	3000	HR	Manager	ORANGE	07 May 2017
30	Madhuri	F	89000	800	IT	Programmer	PURPLE	08 May 2017
31	Survi	F	80000	1000	ADMIN	Manager	PURPLE	09 May 2017

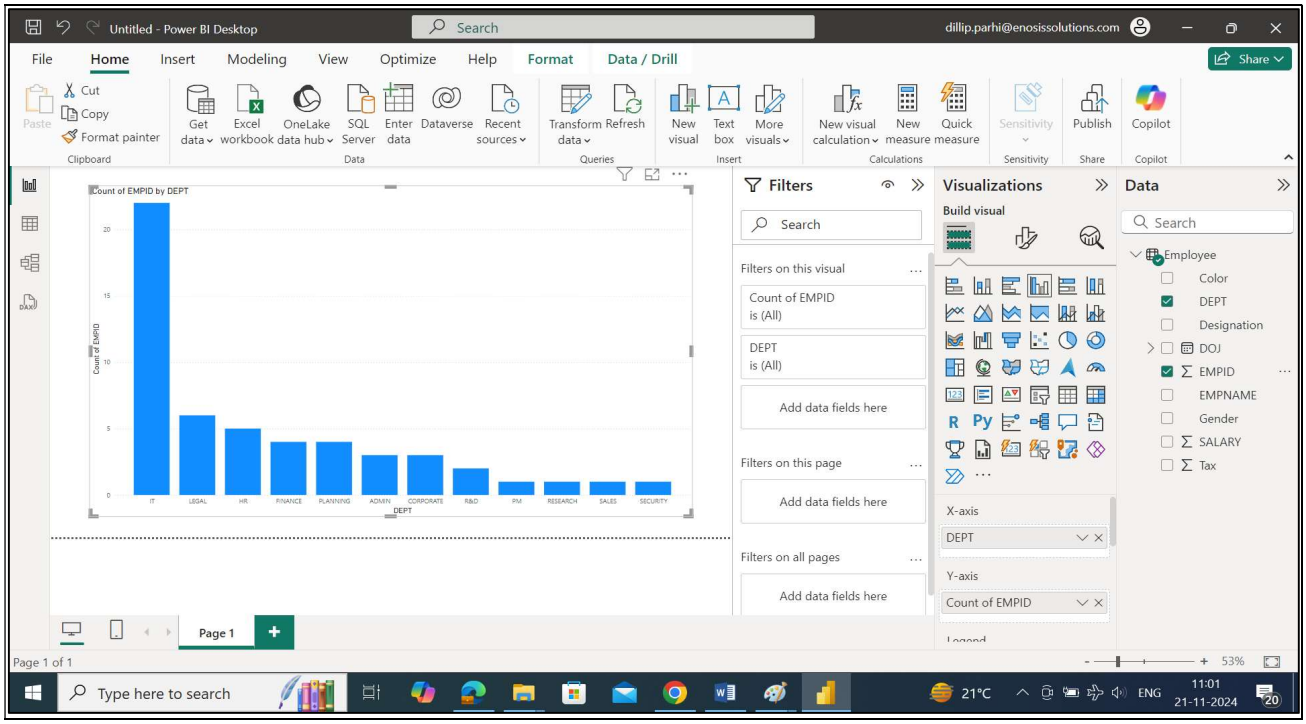
STEP 4 : ADD VISUALIZATION FROM THE VISUALIZATION PANE



- Drag the Visualization from the Visualization Pane to Power BI Page

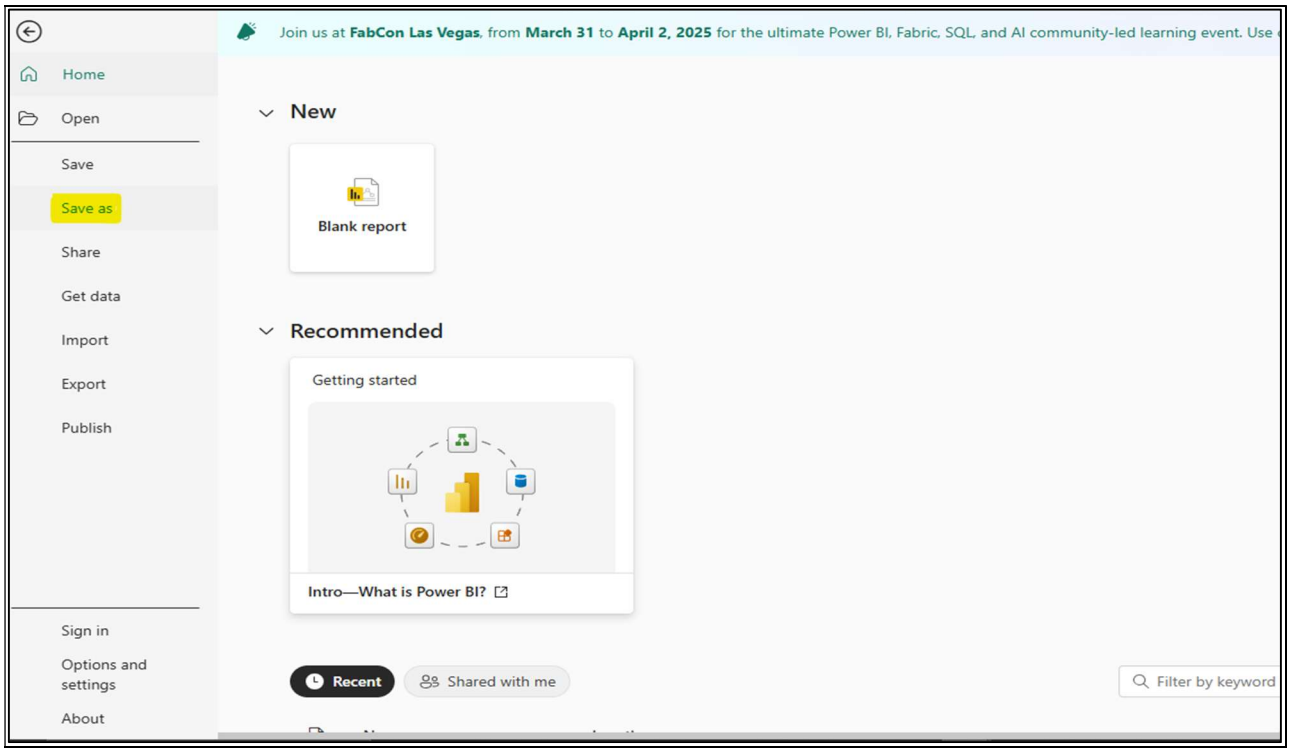


Bind the x- axis and y –axis with the fields /columns data from Data Section.
In this example we have selected the following columns in x axis and y-axis
X- AXIS : DEPT
Y – AXIS : COUNT(EMPID) . Y – axis always apply aggregation by default.

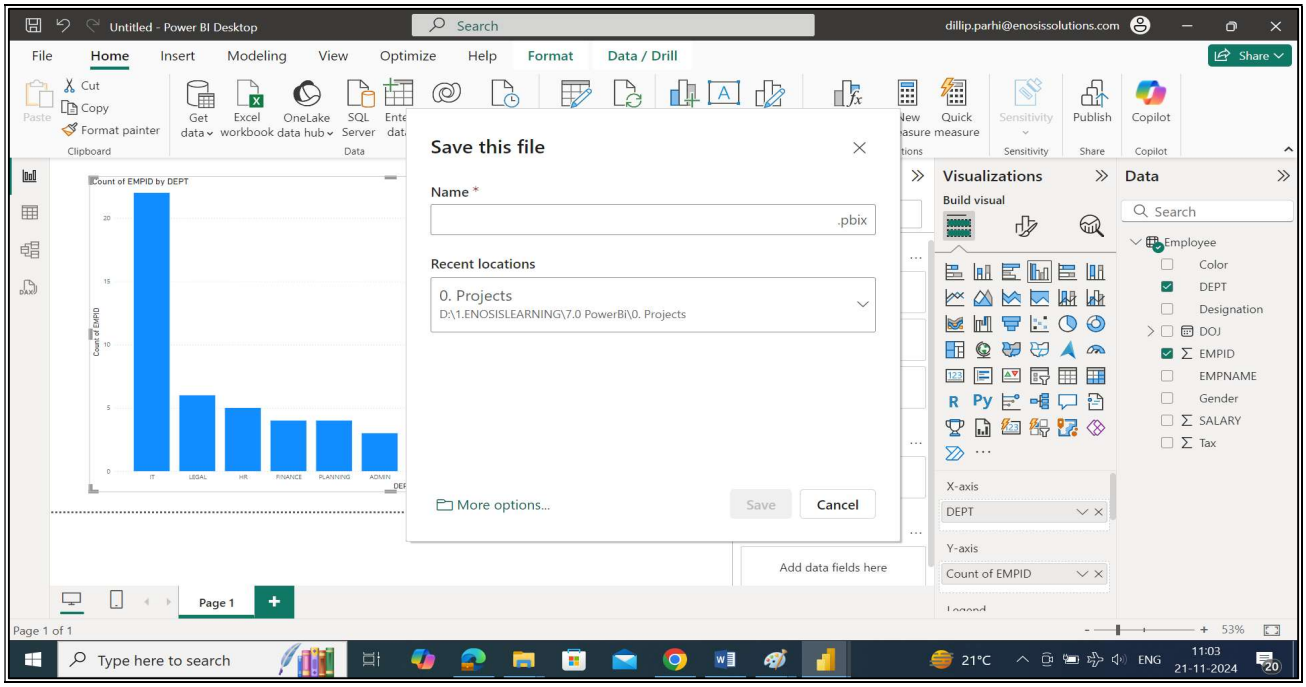


- The data selected from given data column, it will get binded in the visualization.

STEP 5 : SAVE THE REPORT IN A PARTICULAR DRIVE OR LOCATION BY CLICKING ON SAVE FROM THE MENU.

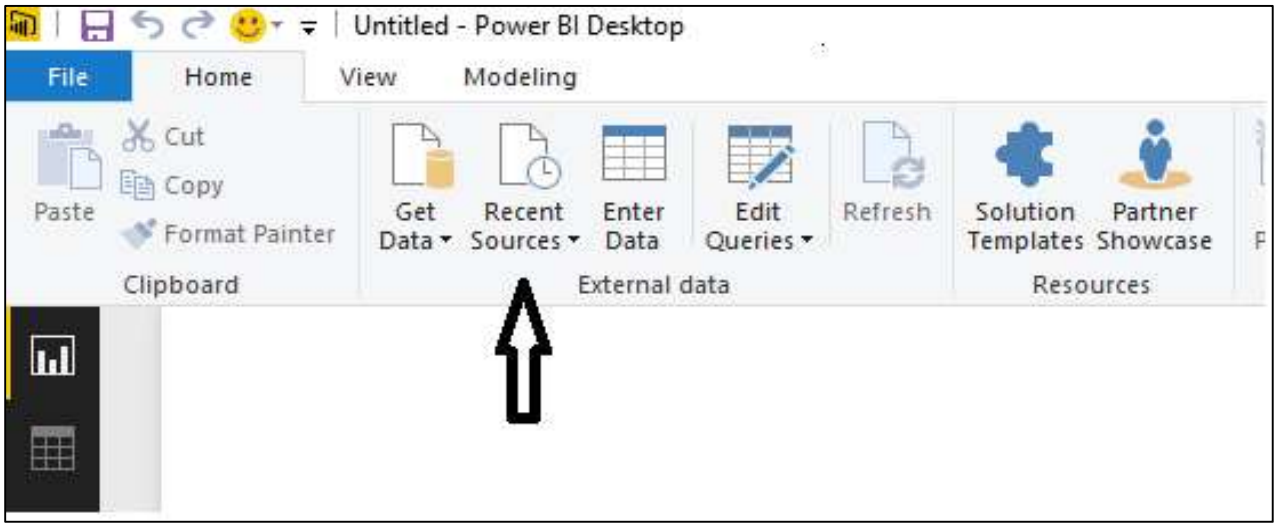


Define the Location and Name of the .PBIX File to Save it.



RECENT SOURCES

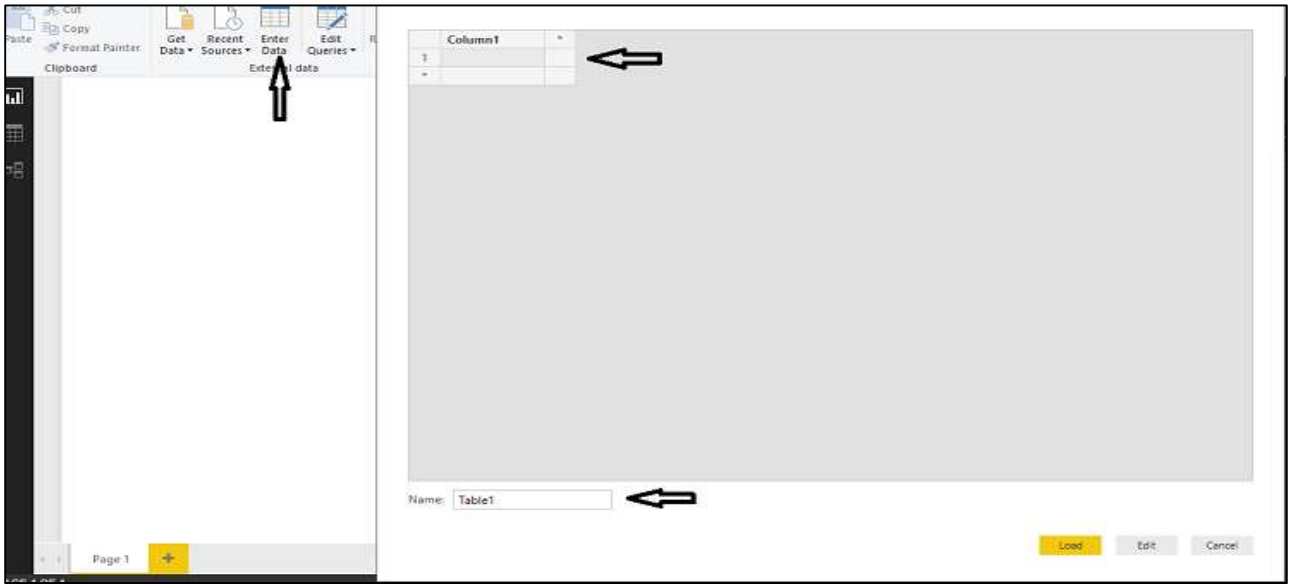
This will show all recent connects made by recent power bi reports.



Note: Power BI Desktop provides an intuitive interface to connect, transform, and visualize data effectively. It allows users to import data from multiple sources, build interactive visuals, and create insightful reports. Each step—from data loading to visualization—helps uncover meaningful patterns and trends. Saving the report as a .pbix file ensures your work can be edited or published later. Mastering these steps builds the foundation for professional Power BI report development.

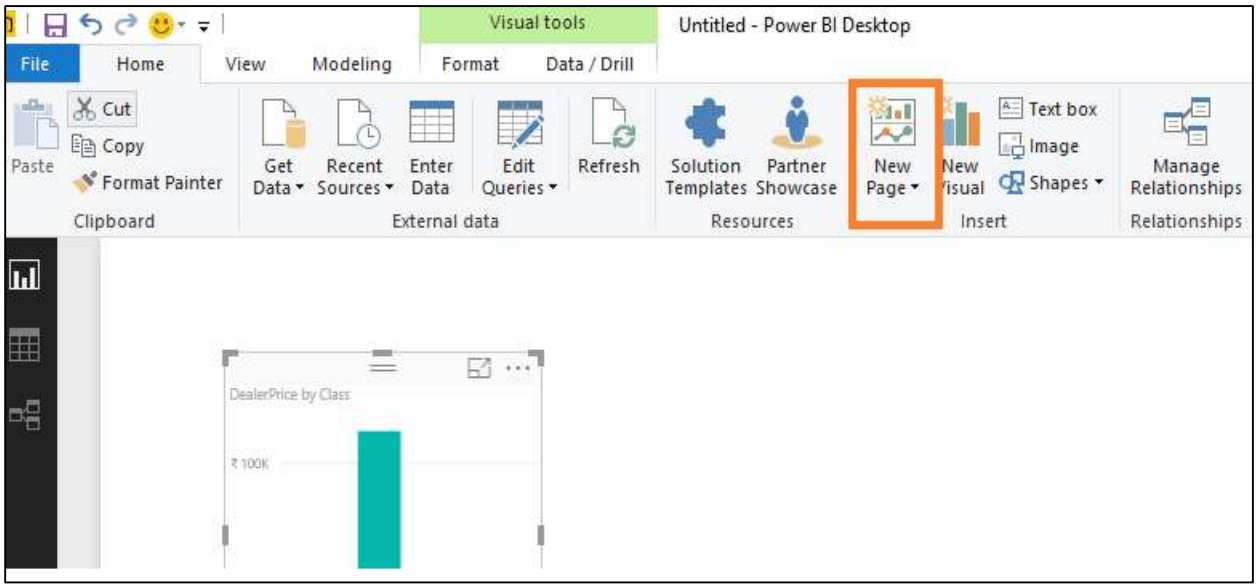
ENTER DATA

This allows to enter data and create your own table in power-bi.



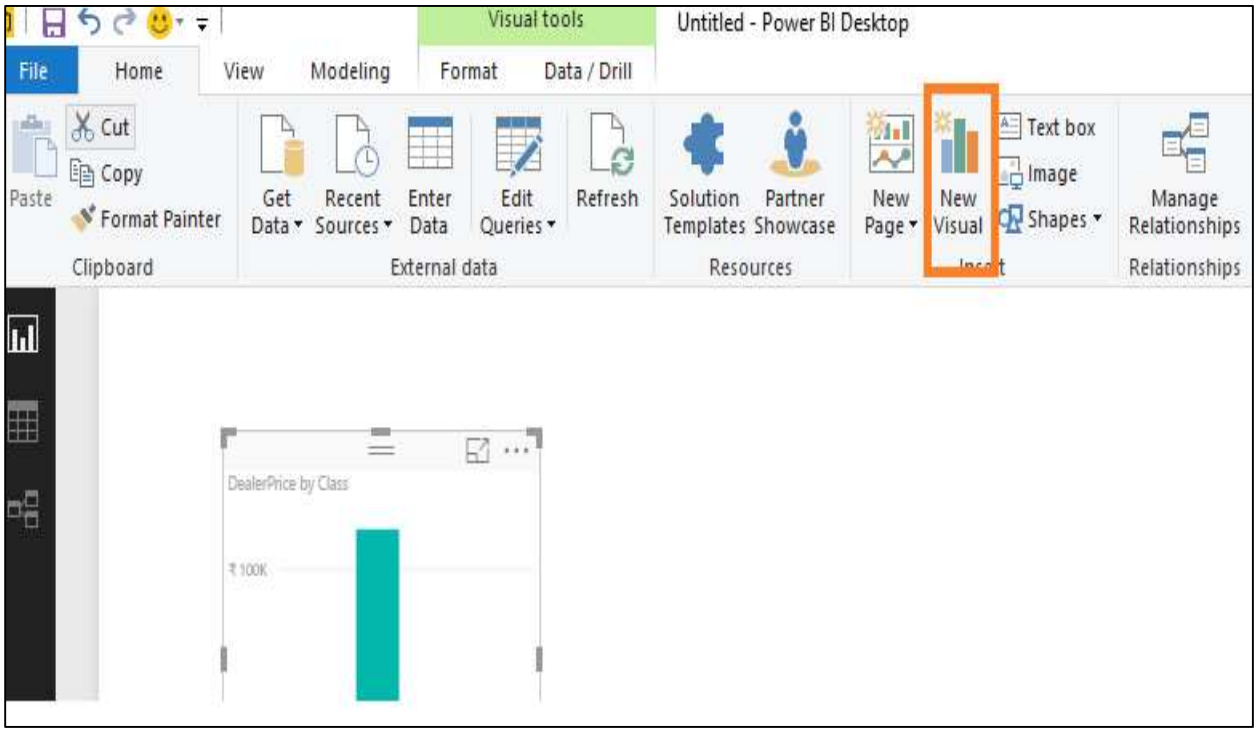
NEW PAGE

This will add a new page to the Power BI Projects



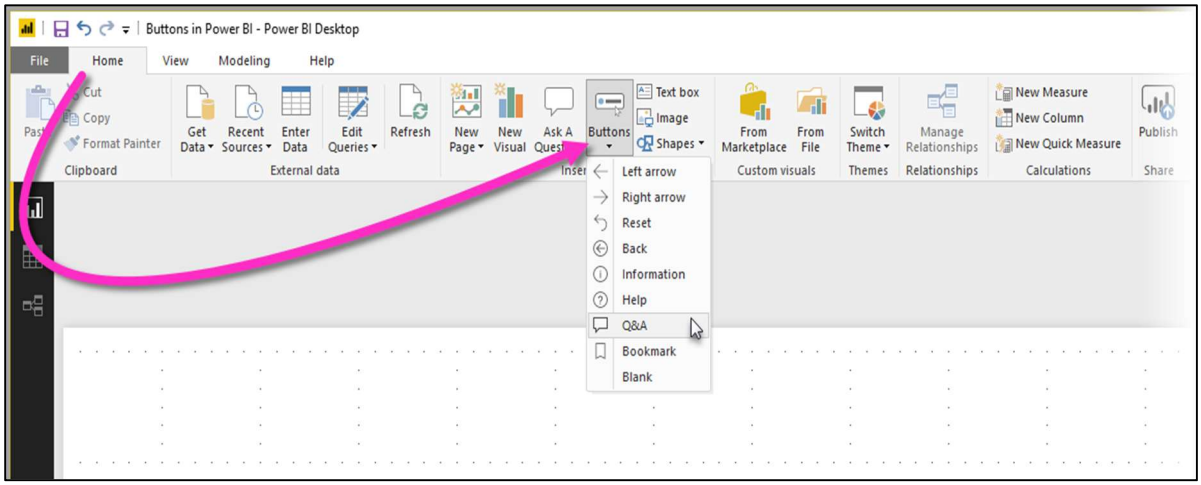
NEW VISUAL

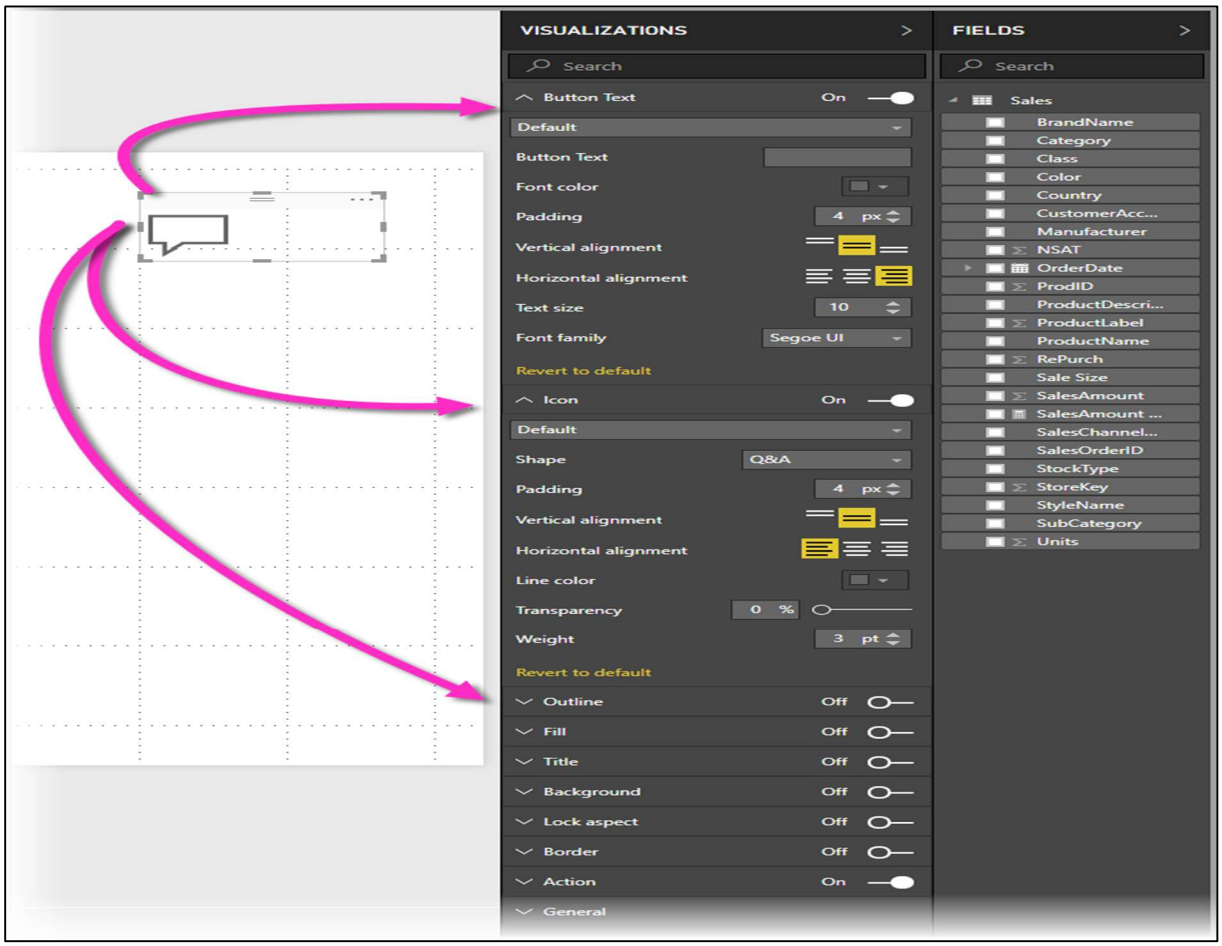
This will add a new visualization component in the below page.



BUTTONS

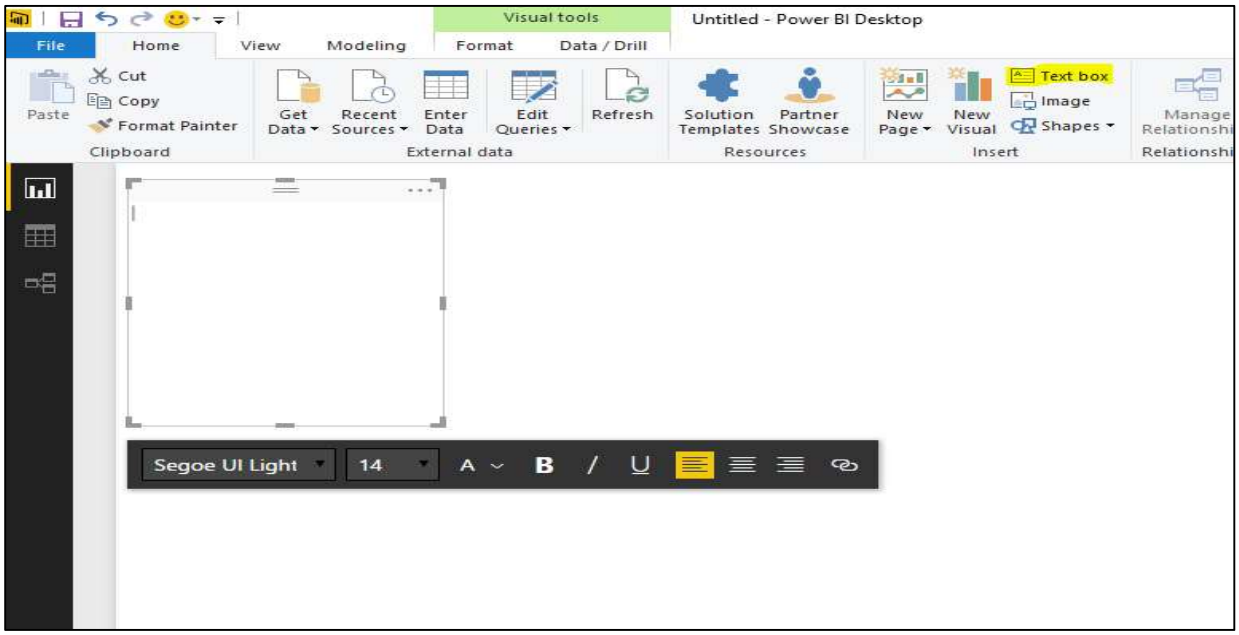
This will add buttons like Left Arrow ,Right Arrow. This buttons can be used for navigations etc.





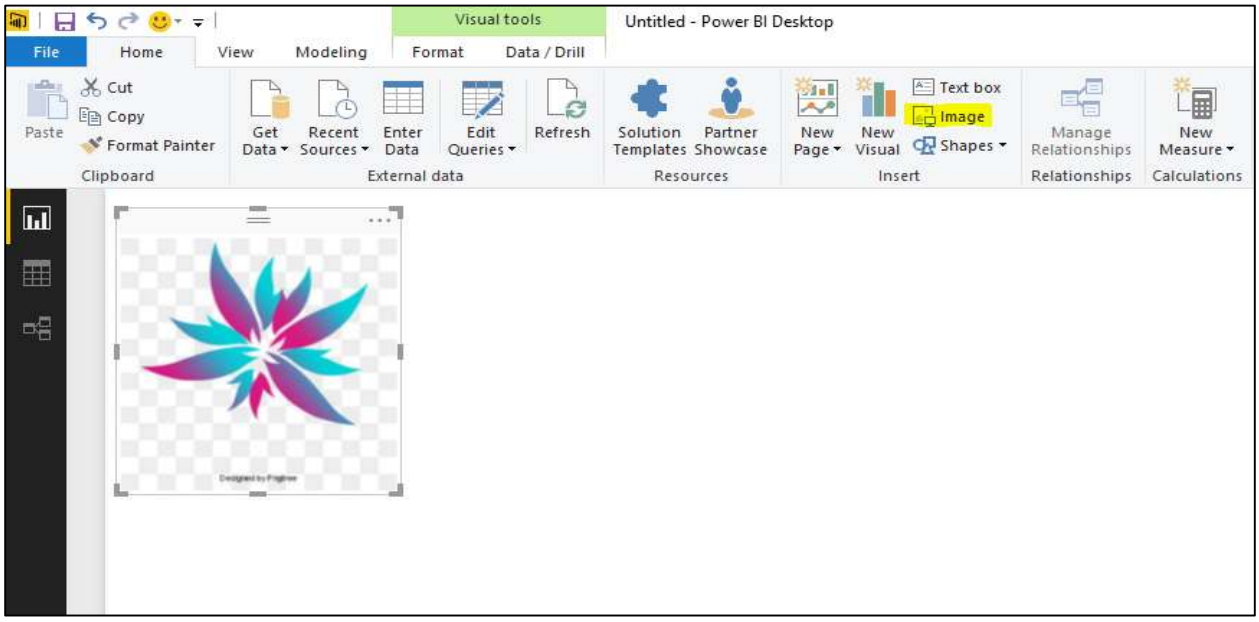
TEXTBOX

This will add a textbox in the Power BI Page



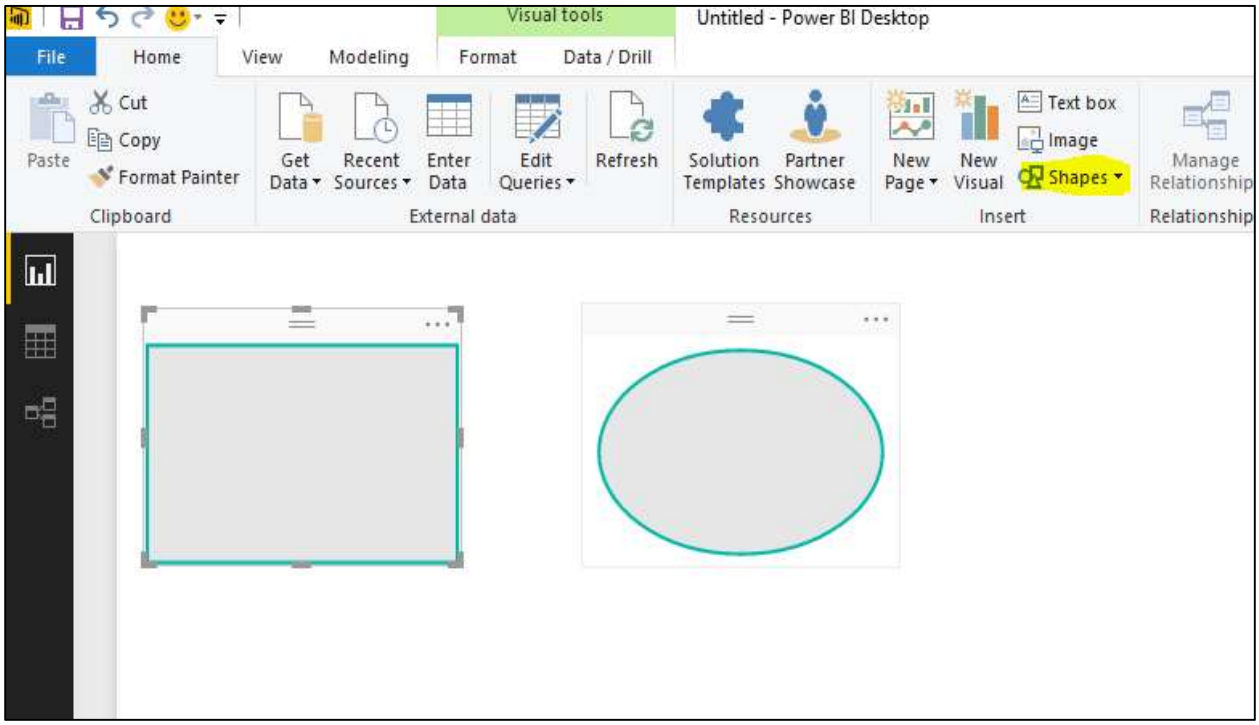
IMAGES

This will add images into the PowerBI Report Page. You can add different external images and show it in Power BI.



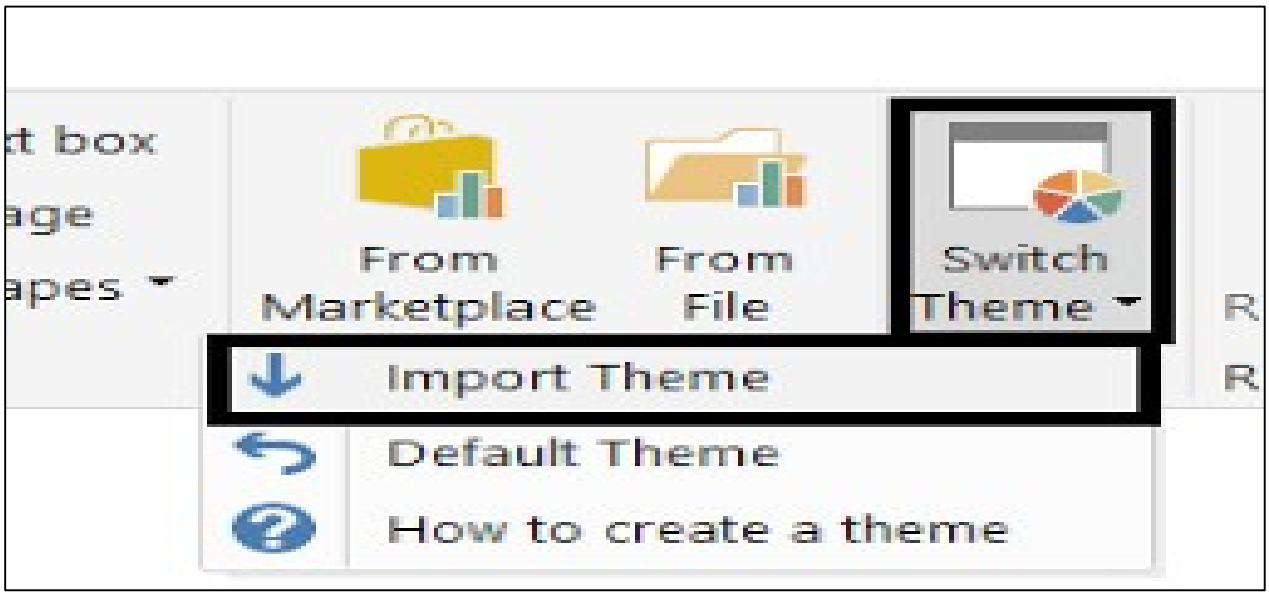
SHAPES

This will add shapes to the Power BI Report Page



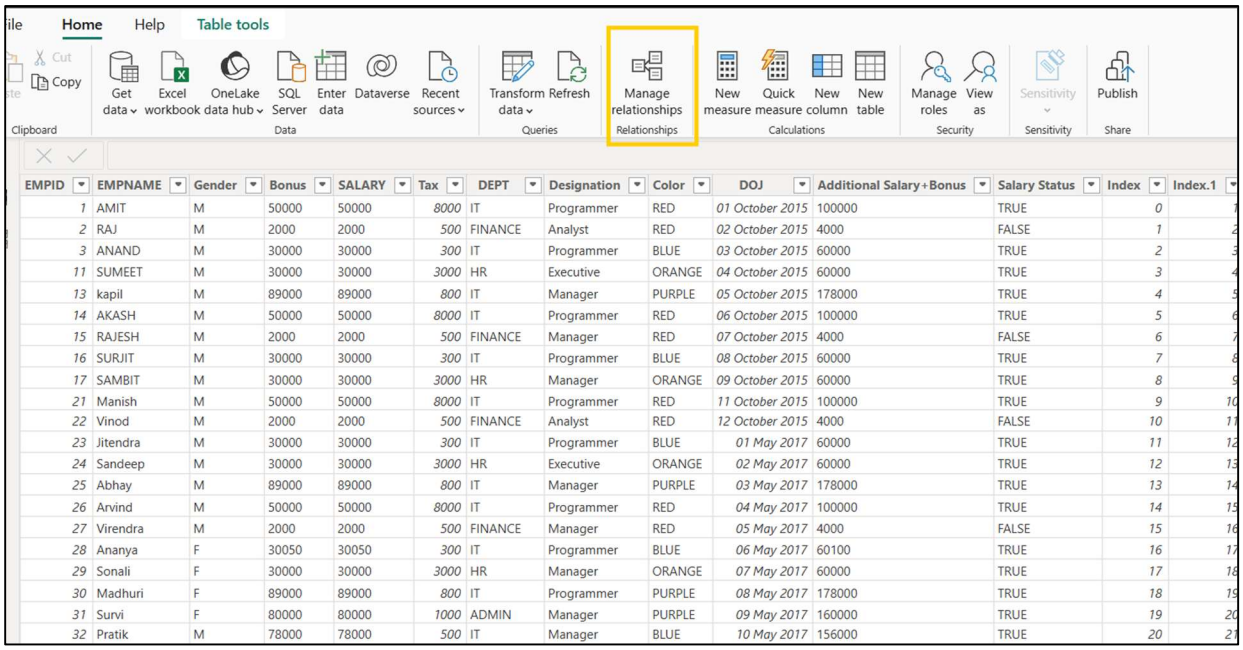
SWITCH THEME

The Switch Theme sections will be used to add different theme to the Power BI Reports. The colors of charts will change based upon the selection of schemes.



MANAGE RELATIONSHIP

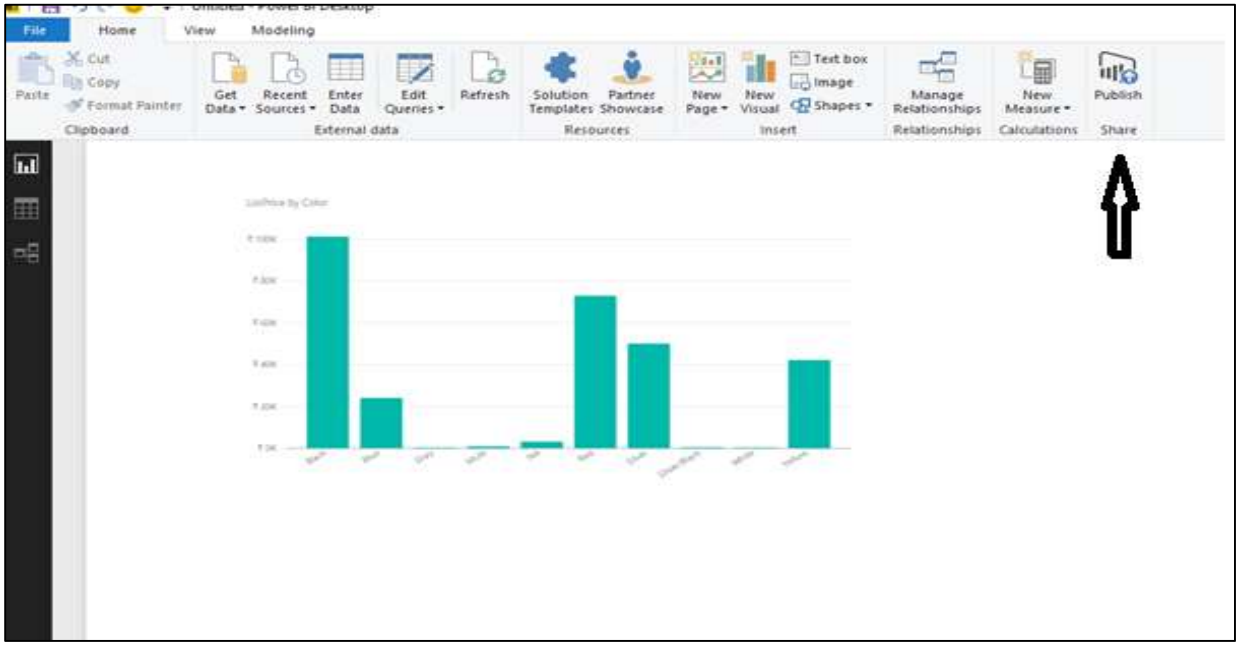
The section is used to show the existing relationship between the tables of the Datasets. We can add or edit the existing relationship of the tables.



EMPID	EMPNAME	Gender	Bonus	SALARY	Tax	DEPT	Designation	Color	DOJ	Additional Salary+Bonus	Salary Status	Index	Index.1
1	AMIT	M	50000	50000	8000	IT	Programmer	RED	01 October 2015	100000	TRUE	0	1
2	RAJ	M	2000	2000	500	FINANCE	Analyst	RED	02 October 2015	4000	FALSE	1	2
3	ANAND	M	30000	30000	300	IT	Programmer	BLUE	03 October 2015	60000	TRUE	2	3
11	SUMEET	M	30000	30000	3000	HR	Executive	ORANGE	04 October 2015	60000	TRUE	3	4
13	kapil	M	89000	89000	800	IT	Manager	PURPLE	05 October 2015	178000	TRUE	4	5
14	AKASH	M	50000	50000	8000	IT	Programmer	RED	06 October 2015	100000	TRUE	5	6
15	RAJESH	M	2000	2000	500	FINANCE	Manager	RED	07 October 2015	4000	FALSE	6	7
16	SURJIT	M	30000	30000	300	IT	Programmer	BLUE	08 October 2015	60000	TRUE	7	8
17	SAMBIT	M	30000	30000	3000	HR	Manager	ORANGE	09 October 2015	60000	TRUE	8	9
21	Manish	M	50000	50000	8000	IT	Programmer	RED	11 October 2015	100000	TRUE	9	10
22	Vinod	M	2000	2000	500	FINANCE	Analyst	RED	12 October 2015	4000	FALSE	10	11
23	Jitendra	M	30000	30000	300	IT	Programmer	BLUE	01 May 2017	60000	TRUE	11	12
24	Sandeep	M	30000	30000	3000	HR	Executive	ORANGE	02 May 2017	60000	TRUE	12	13
25	Abhay	M	89000	89000	800	IT	Manager	PURPLE	03 May 2017	178000	TRUE	13	14
26	Arvind	M	50000	50000	8000	IT	Programmer	RED	04 May 2017	100000	TRUE	14	15
27	Virendra	M	2000	2000	500	FINANCE	Manager	RED	05 May 2017	4000	FALSE	15	16
28	Ananya	F	30050	30050	300	IT	Programmer	BLUE	06 May 2017	60100	TRUE	16	17
29	Sonali	F	30000	30000	3000	HR	Manager	ORANGE	07 May 2017	60000	TRUE	17	18
30	Madhuri	F	89000	89000	800	IT	Programmer	PURPLE	08 May 2017	178000	TRUE	18	19
31	Survi	F	80000	80000	1000	ADMIN	Manager	PURPLE	09 May 2017	160000	TRUE	19	20
32	Pratik	M	78000	78000	500	IT	Manager	BLUE	10 May 2017	156000	TRUE	20	21

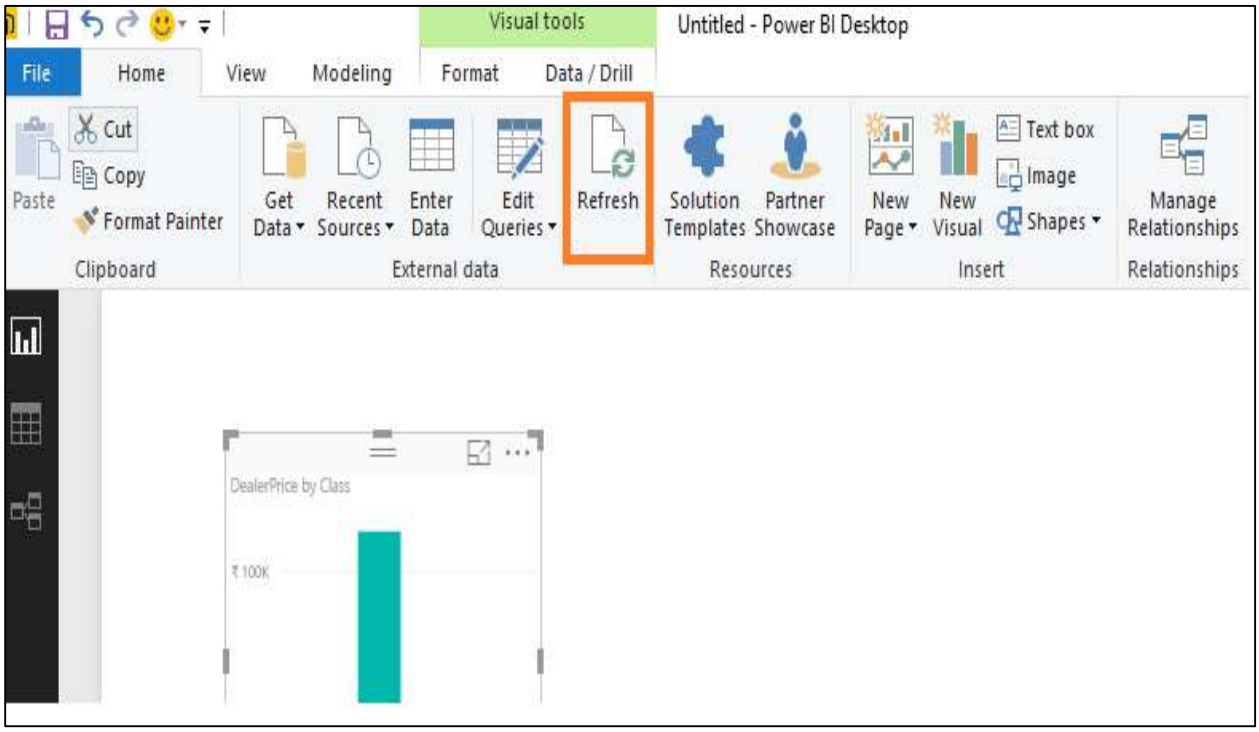
PUBLISH

This will publish the power bi reports into the Power BI Cloud Server



REFRESH

This will refresh the page



Note: Power BI Desktop offers various tools to enhance report creation and presentation. Options like Enter Data, New Page, and New Visual help build and organize reports efficiently. You can add buttons, text, images, and shapes to make reports more interactive and visually appealing. Features such as Switch Theme and Manage Relationships ensure consistency and accurate data connections. Finally, the Publish option allows you to share your reports securely through the Power BI Service.

QUESTIONS

1.What is the purpose of Power BI Desktop in the Power BI ecosystem?
2.Explain the functions of Report View, Data View, and Model View in Power BI Desktop.
3.What is the role of the Visualization Pane and Fields Pane in Power BI Desktop?
4.How does the Manage Relationships feature help in creating accurate reports in Power BI Desktop?
5.What is the difference between the Publish and Refresh functionalities in Power BI Desktop?

SUMMARY

Power BI Desktop is the core application of the Power BI suite, designed for **data modeling, transformation, and visualization**. It provides a comprehensive environment where users can connect to multiple data sources, clean and shape data, build relationships between tables, and design interactive reports and dashboards—all from a single platform.

Interface and Views

1. **Report View** – The default workspace where users design dashboards using charts, visuals, and KPIs. It acts as the canvas for report creation.
2. **Data View** – Displays all imported datasets in tabular form. It allows users to inspect data, add calculated columns, and perform quick validation checks.
3. **Model View** – Displays the relationships between multiple tables in the form of a data model diagram. It helps define relationships (one-to-many, many-to-one, etc.) and maintain referential integrity across datasets.

Key Components

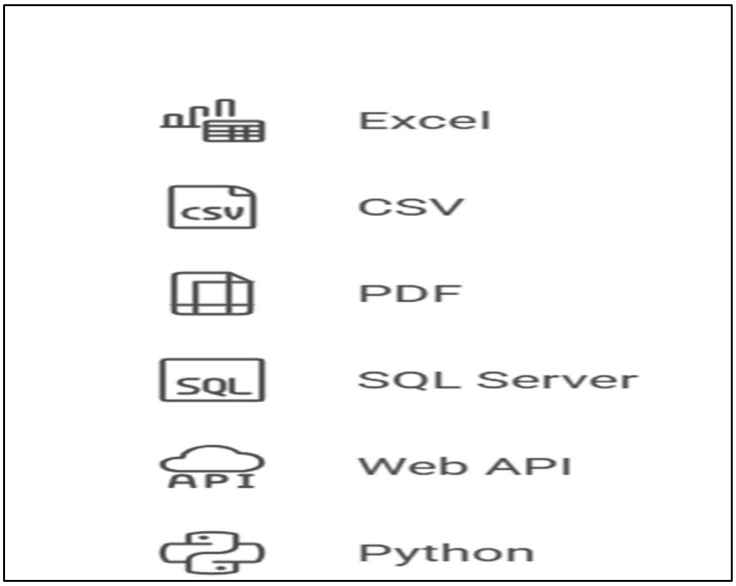
- **Get Data:** The starting point to connect Power BI Desktop to various data sources such as Excel, SQL Server, CSV, Web APIs, and cloud services.
- **Visualization Pane:** Provides a collection of built-in chart types (bar, line, pie, map, etc.) and tools for customization. Users can drag and drop fields to create visuals easily.
- **Fields Pane:** Lists all available tables and columns from the data model. These can be used to build visuals or apply filters.
- **Filters Pane:** Used to add visual-level, page-level, or report-level filters to control data displayed in charts.

Additional Functionalities

- **Manage Relationships:** Allows users to create or modify relationships between tables, ensuring accurate data joins and interactions in reports.
- **Switch Theme:** Enables quick application of color palettes or corporate branding to maintain consistency across visuals.
- **Buttons, Text Boxes, Images, and Shapes:** Enhance the presentation and interactivity of reports—useful for storytelling, navigation, and design aesthetics.
- **Publish:** With a single click, reports can be published from Power BI Desktop to the **Power BI Service**, making them accessible to stakeholders.
- **Refresh:** Updates data connections and re-imports the latest information from connected sources.

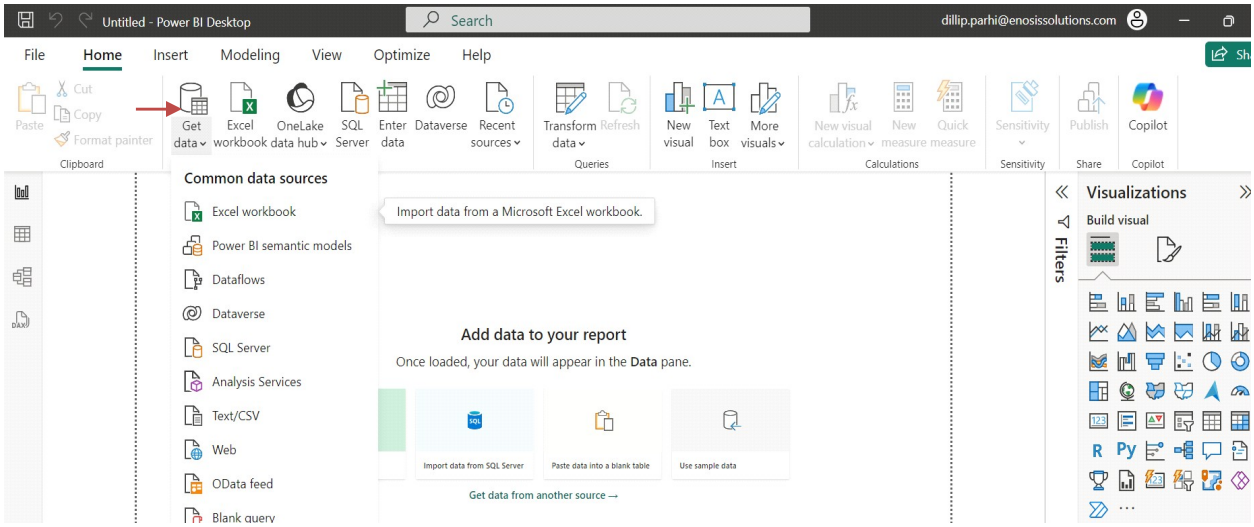
CHAPTER 3 : POWER BI DATA SOURCES

In this chapter we will explore important data sources of Power BI Desktop

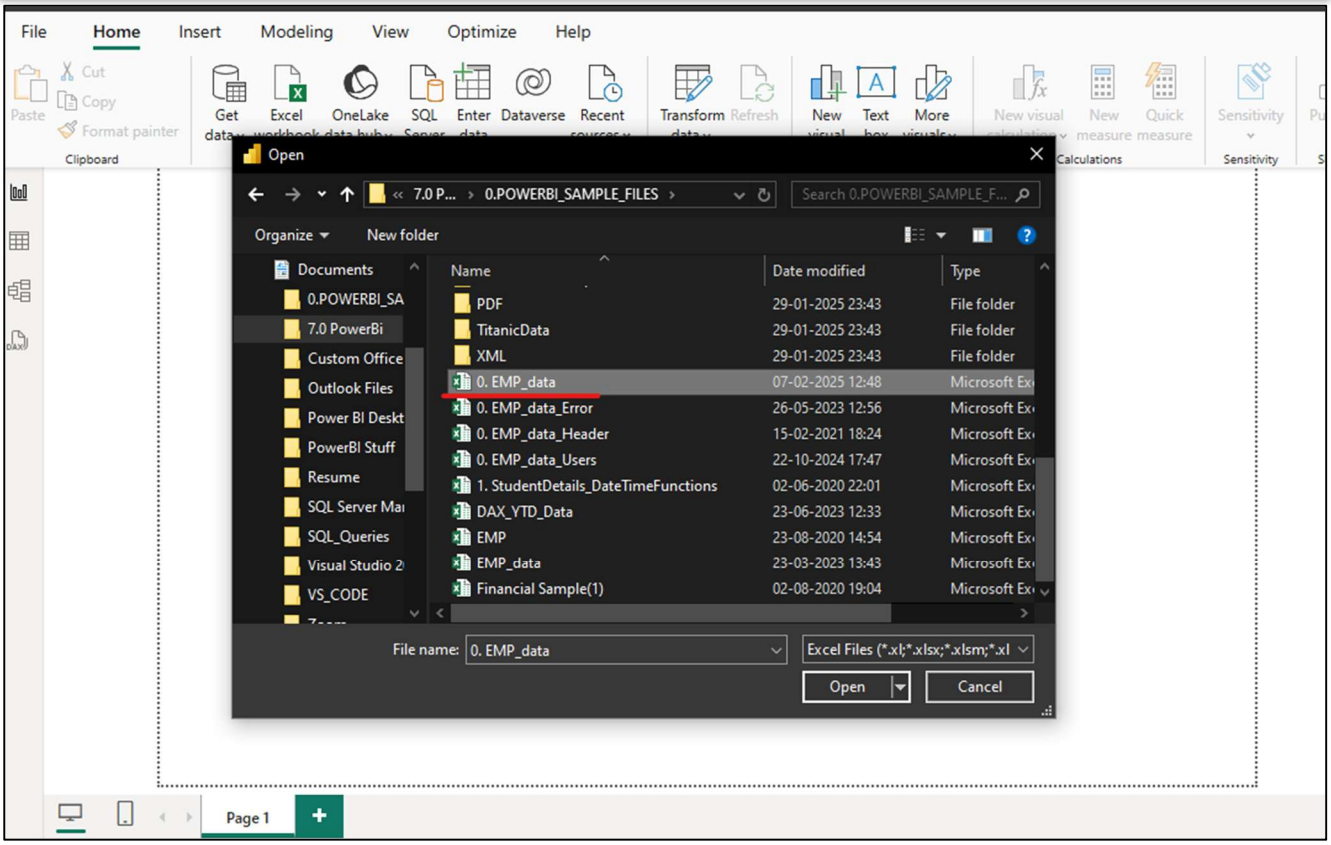


CONNECTING WITH EXCEL

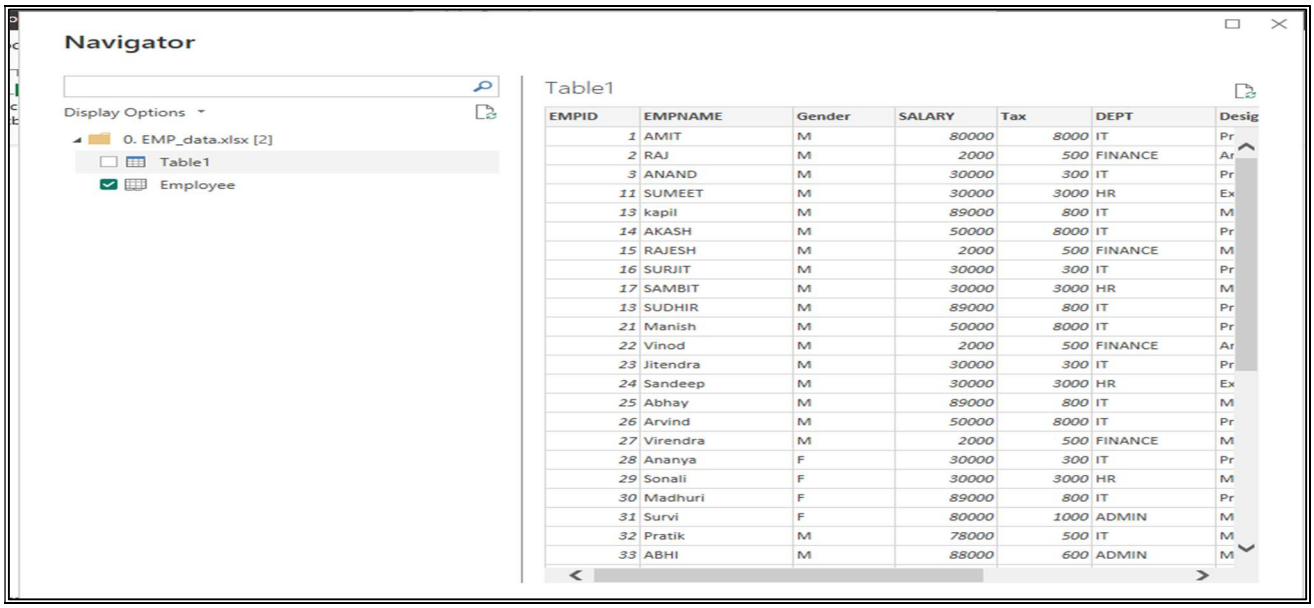
The example below explains the process loading Excel file in Power BI.



STEP 2 : BROWSE TO THE PATH AND SELECT THE FILE (EXCEL FILE)

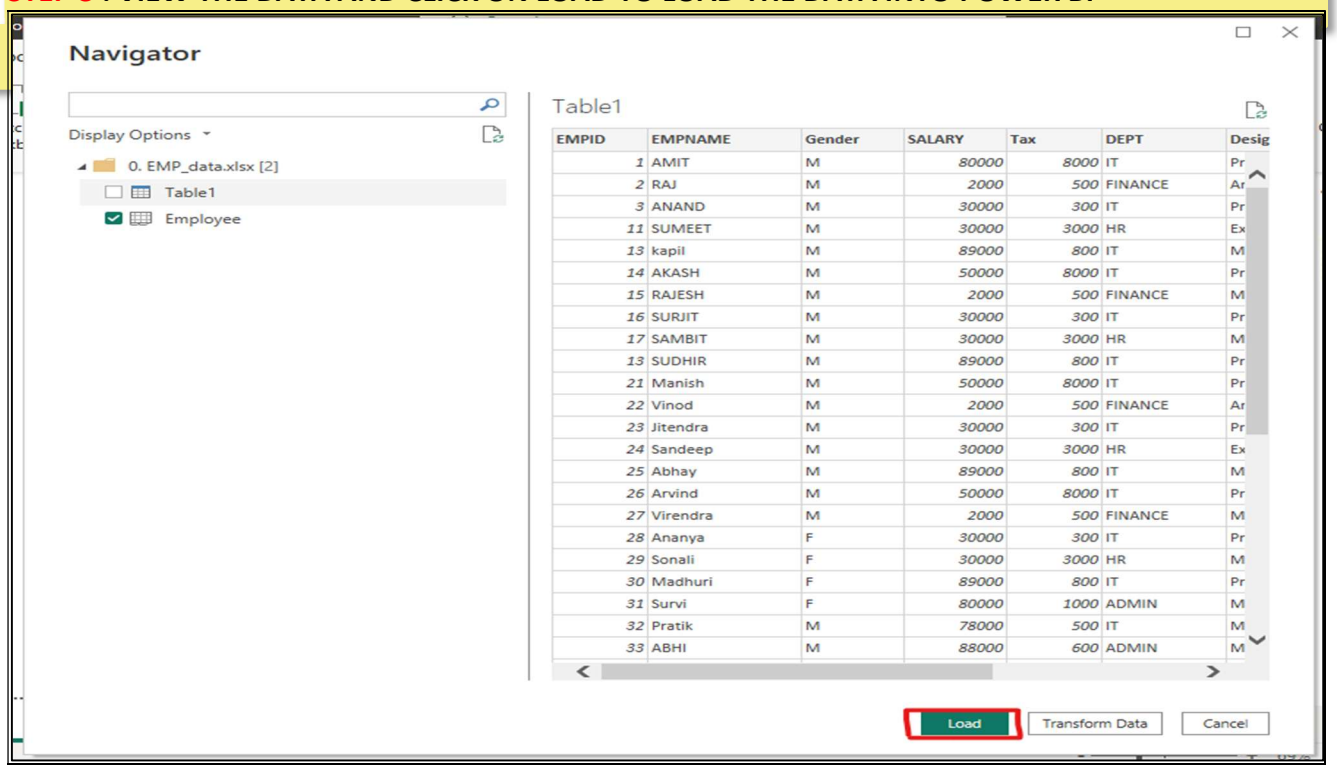


- **Select the sheet from the excel path**
- It will display the sheets present in the Excel, sheet the Employee sheet data



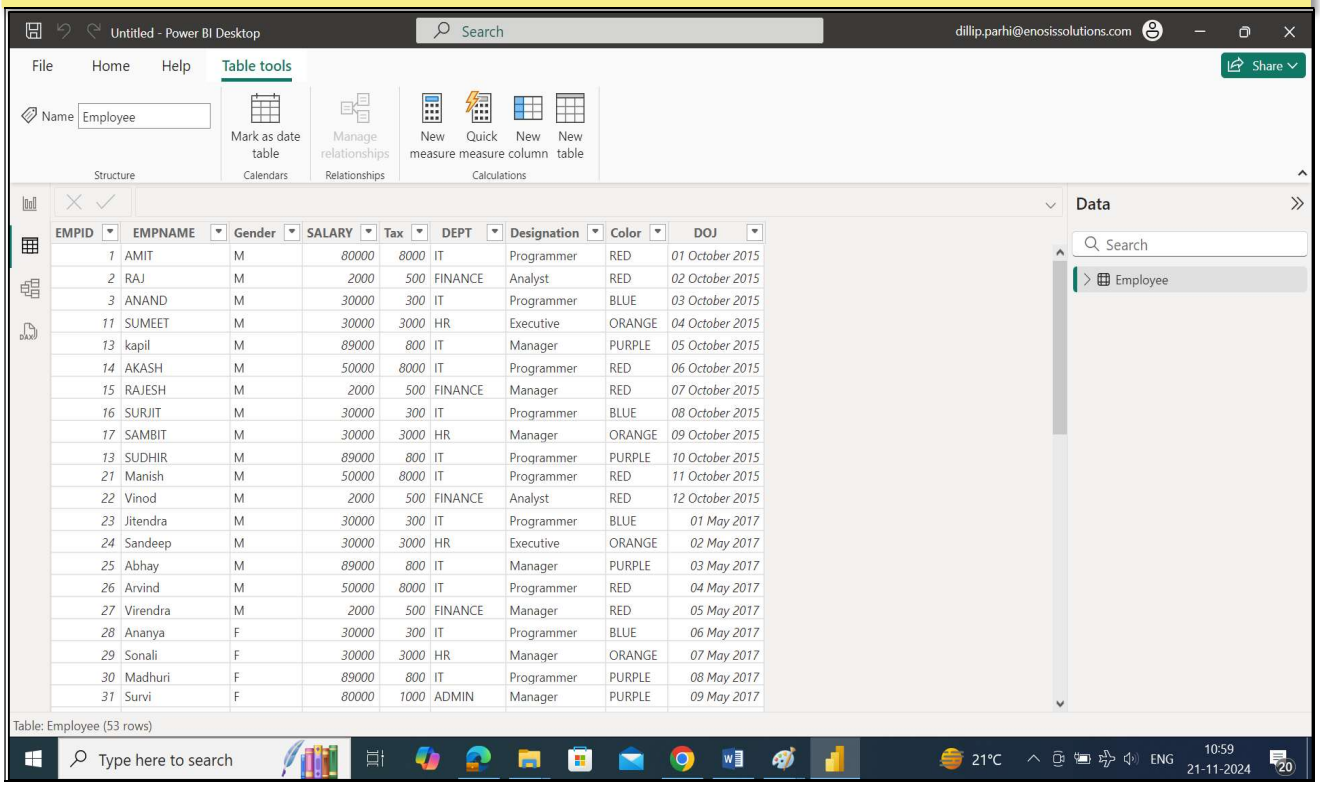
EMPID	EMPNAME	Gender	SALARY	Tax	DEPT	Desig
1	AMIT	M	80000	8000	IT	Pr
2	RAJ	M	2000	500	FINANCE	Ar
3	ANAND	M	30000	300	IT	Pr
11	SUMEET	M	30000	3000	HR	Ex
13	kapil	M	89000	800	IT	M
14	AKASH	M	50000	8000	IT	Pr
15	RAJESH	M	2000	500	FINANCE	M
16	SURJIT	M	30000	300	IT	Pr
17	SAMBIT	M	30000	3000	HR	M
13	SUDHIR	M	89000	800	IT	Pr
21	Manish	M	50000	8000	IT	Pr
22	Vinod	M	2000	500	FINANCE	Ar
23	Jitendra	M	30000	300	IT	Pr
24	Sandeep	M	30000	3000	HR	Ex
25	Abhay	M	89000	800	IT	M
26	Arvind	M	50000	8000	IT	Pr
27	Virendra	M	2000	500	FINANCE	M
28	Ananya	F	30000	300	IT	Pr
29	Sonali	F	30000	3000	HR	M
30	Madhuri	F	89000	800	IT	Pr
31	Survi	F	80000	1000	ADMIN	M
32	Pratik	M	78000	500	IT	M
33	ABHI	M	88000	600	ADMIN	M

STEP 3 : VIEW THE DATA AND CLICK ON LOAD TO LOAD THE DATA INTO POWER BI



EMPID	EMPNAME	Gender	SALARY	Tax	DEPT	Desig
1	AMIT	M	80000	8000	IT	Pr
2	RAJ	M	2000	500	FINANCE	Ar
3	ANAND	M	30000	300	IT	Pr
11	SUMEET	M	30000	3000	HR	Ex
13	kapil	M	89000	800	IT	M
14	AKASH	M	50000	8000	IT	Pr
15	RAJESH	M	2000	500	FINANCE	M
16	SURJIT	M	30000	300	IT	Pr
17	SAMBIT	M	30000	3000	HR	M
13	SUDHIR	M	89000	800	IT	Pr
21	Manish	M	50000	8000	IT	Pr
22	Vinod	M	2000	500	FINANCE	Ar
23	Jitendra	M	30000	300	IT	Pr
24	Sandeep	M	30000	3000	HR	Ex
25	Abhay	M	89000	800	IT	M
26	Arvind	M	50000	8000	IT	Pr
27	Virendra	M	2000	500	FINANCE	M
28	Ananya	F	30000	300	IT	Pr
29	Sonali	F	30000	3000	HR	M
30	Madhuri	F	89000	800	IT	Pr
31	Survi	F	80000	1000	ADMIN	M
32	Pratik	M	78000	500	IT	M
33	ABHI	M	88000	600	ADMIN	M

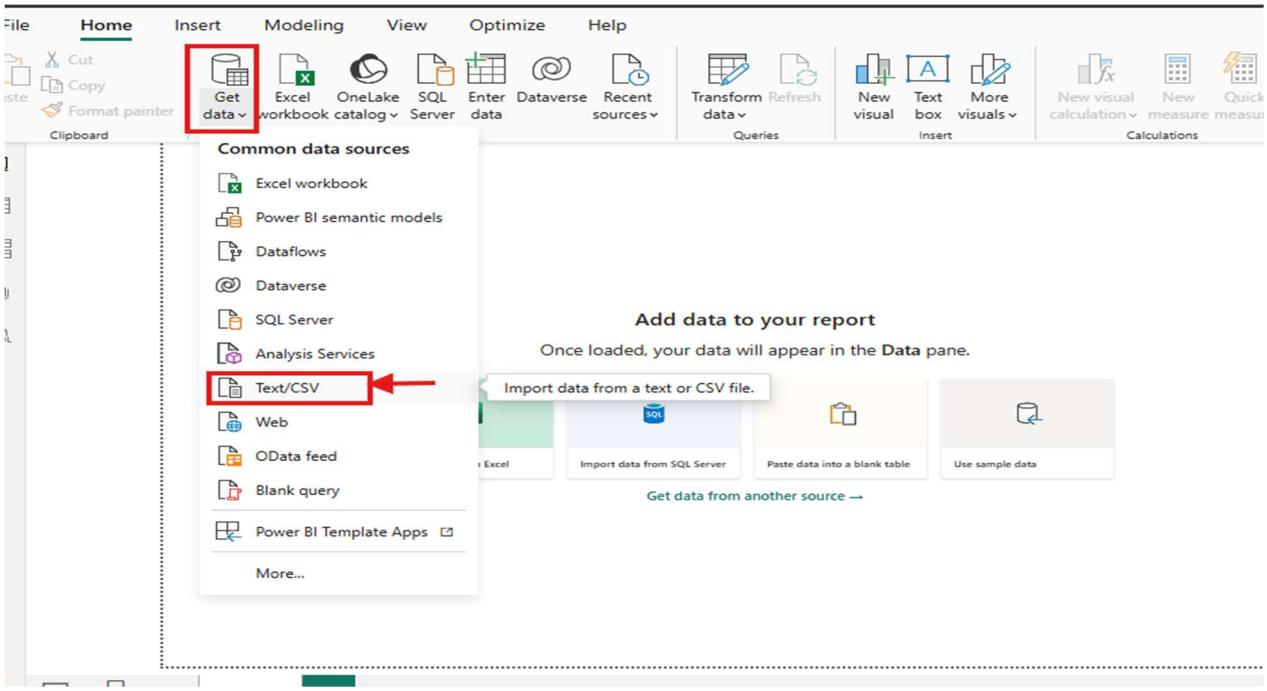
STEP 4 : SUCCESSFULLY LOADED EXCEL FILE OF EMP DATA IN POWER BI



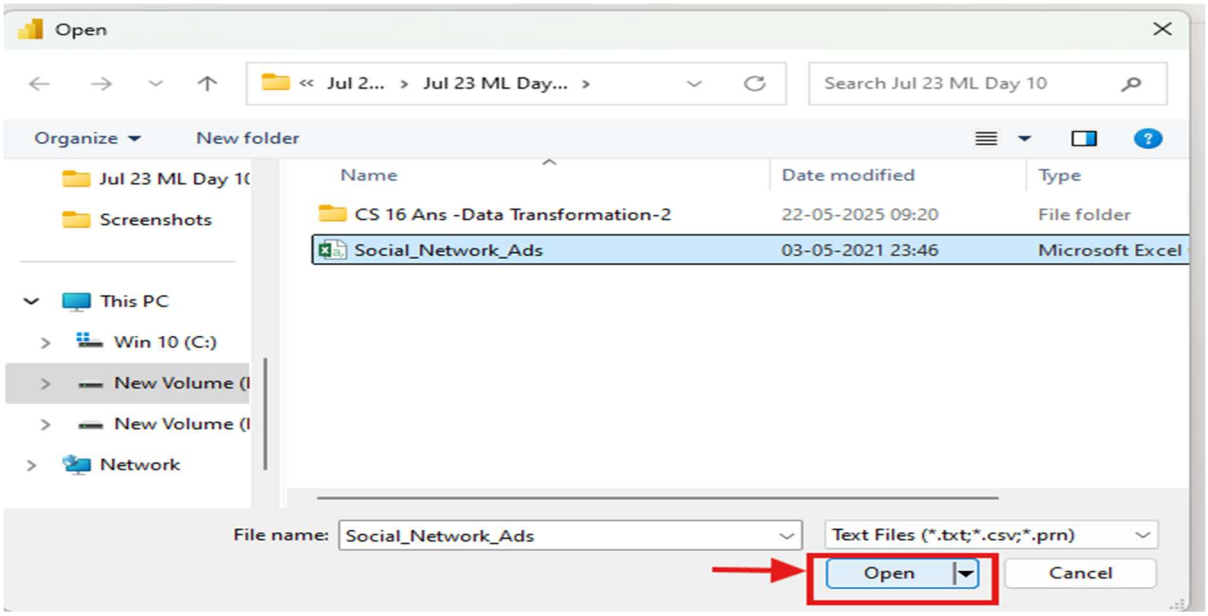
EMPID	EMPNAME	Gender	SALARY	Tax	DEPT	Designation	Color	DOJ
1	AMIT	M	80000	8000	IT	Programmer	RED	01 October 2015
2	RAJ	M	2000	500	FINANCE	Analyst	RED	02 October 2015
3	ANAND	M	30000	300	IT	Programmer	BLUE	03 October 2015
11	SUMEET	M	30000	3000	HR	Executive	ORANGE	04 October 2015
13	kapil	M	89000	800	IT	Manager	PURPLE	05 October 2015
14	AKASH	M	50000	8000	IT	Programmer	RED	06 October 2015
15	RAJESH	M	2000	500	FINANCE	Manager	RED	07 October 2015
16	SURJIT	M	30000	300	IT	Programmer	BLUE	08 October 2015
17	SAMBIT	M	30000	3000	HR	Manager	ORANGE	09 October 2015
13	SUDHIR	M	89000	800	IT	Programmer	PURPLE	10 October 2015
21	Manish	M	50000	8000	IT	Programmer	RED	11 October 2015
22	Vinod	M	2000	500	FINANCE	Analyst	RED	12 October 2015
23	Jitendra	M	30000	300	IT	Programmer	BLUE	01 May 2017
24	Sandeep	M	30000	3000	HR	Executive	ORANGE	02 May 2017
25	Abhay	M	89000	800	IT	Manager	PURPLE	03 May 2017
26	Arvind	M	50000	8000	IT	Programmer	RED	04 May 2017
27	Virendra	M	2000	500	FINANCE	Manager	RED	05 May 2017
28	Ananya	F	30000	300	IT	Programmer	BLUE	06 May 2017
29	Sonali	F	30000	3000	HR	Manager	ORANGE	07 May 2017
30	Madhuri	F	89000	800	IT	Programmer	PURPLE	08 May 2017
31	Survi	F	80000	1000	ADMIN	Manager	PURPLE	09 May 2017

STEP 1: IN **HOME->GET DATA** ,THEN SELECT EXCEL WORKBOOK FROM VARIOUS DATA SOURCES

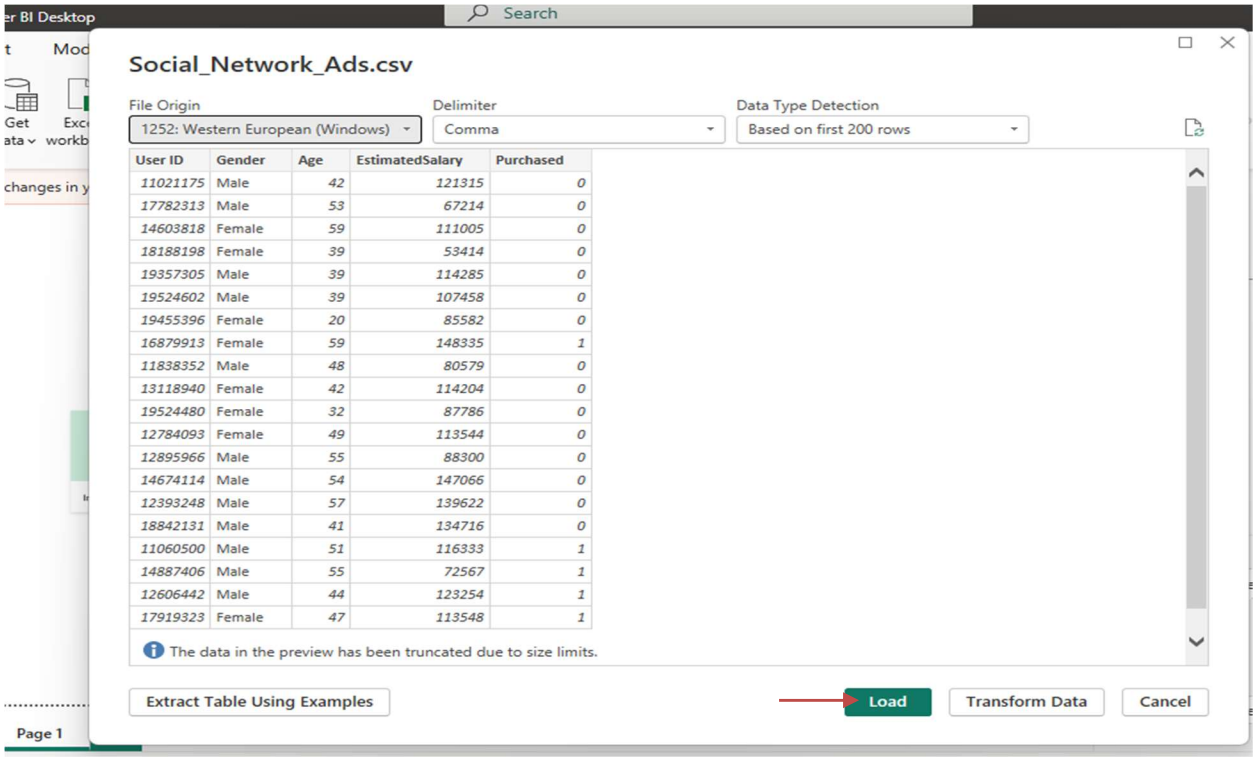
STEP 1 : OPEN HOME TAB -> GET DATA. CLICK ON TEXT/CSV



STEP 2 : BROWSE YOUR COMPUTER AND SELECT THE CSV FILE YOU WANT TO IMPORT. CLICK OPEN.

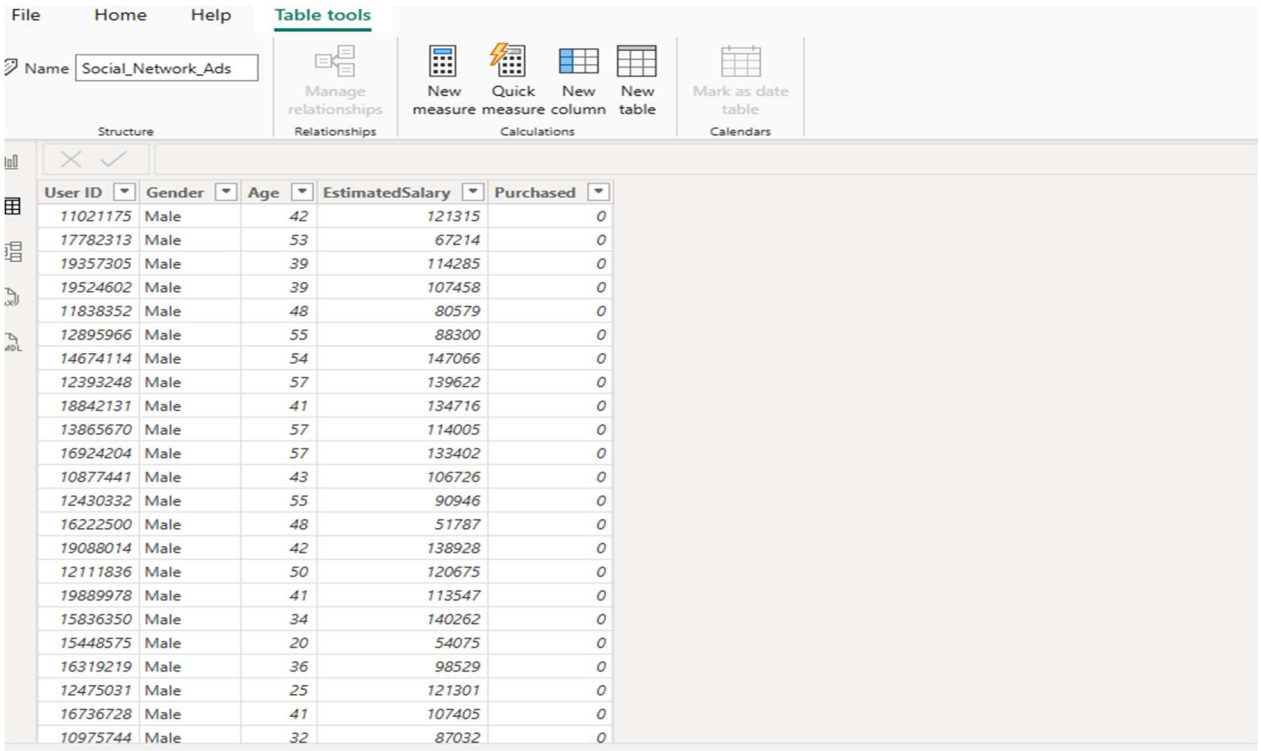


STEP3 : A PREVIEW WINDOW WILL APPEAR SHOWING SAMPLE ROWS FROM YOUR CSV FILE. CLICK LOAD → DIRECTLY LOAD DATA INTO POWER BI.



User ID	Gender	Age	EstimatedSalary	Purchased
11021175	Male	42	121315	0
17782313	Male	53	67214	0
14603818	Female	59	111005	0
18188198	Female	39	53414	0
19357305	Male	39	114285	0
19524602	Male	39	107458	0
19455396	Female	20	85582	0
16879913	Female	59	148335	1
11838352	Male	48	80579	0
13118940	Female	42	114204	0
19524480	Female	32	87786	0
12784093	Female	49	113544	0
12895966	Male	55	88300	0
14674114	Male	54	147066	0
12393248	Male	57	139622	0
18842131	Male	41	134716	0
11060500	Male	51	116333	1
14887406	Male	55	72567	1
12606442	Male	44	123254	1
17919323	Female	47	113548	1

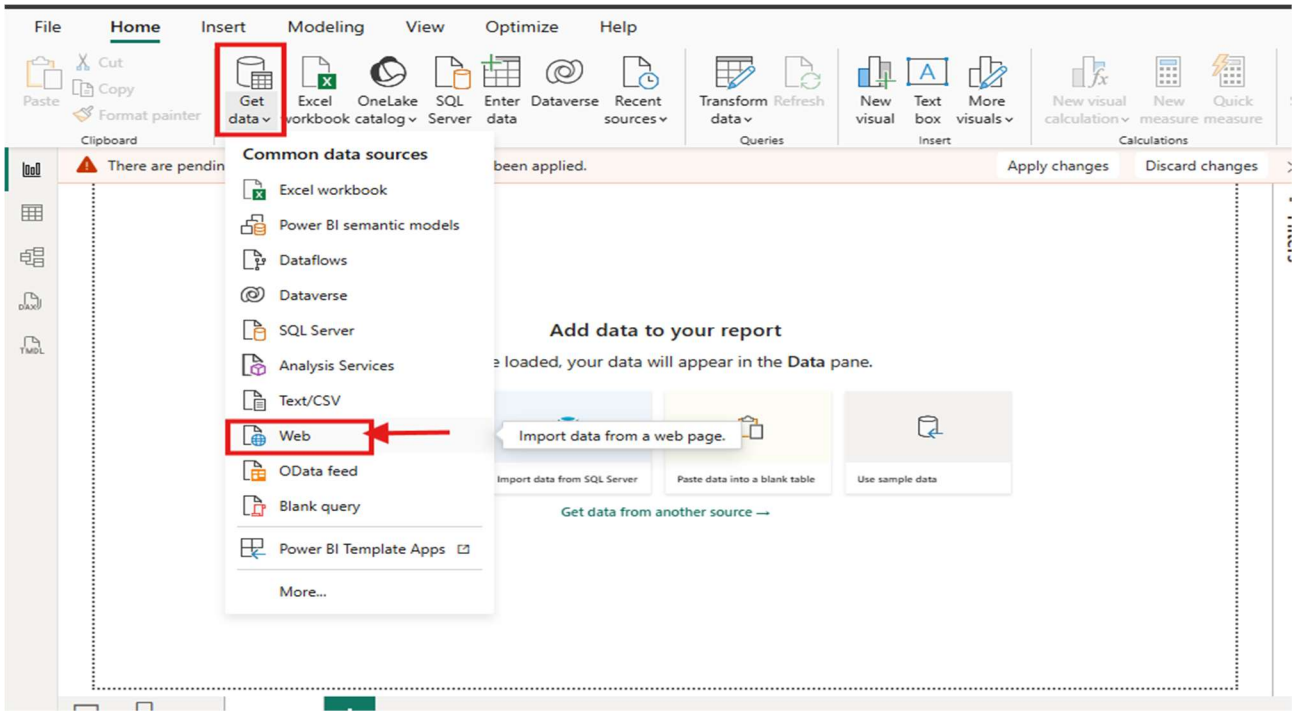
STEP 4 : THE CSV DATA IS NOW IMPORTED INTO POWER BI.



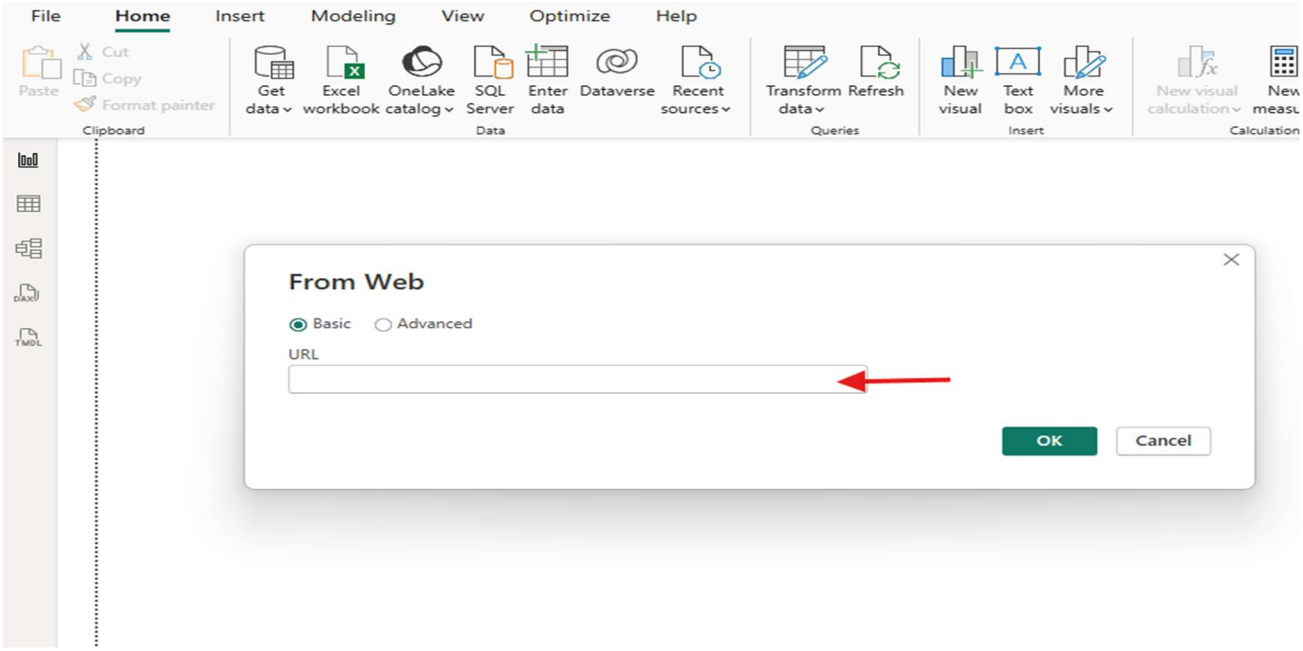
User ID	Gender	Age	EstimatedSalary	Purchased
11021175	Male	42	121315	0
17782313	Male	53	67214	0
19357305	Male	39	114285	0
19524602	Male	39	107458	0
11838352	Male	48	80579	0
12895966	Male	55	88300	0
14674114	Male	54	147066	0
12393248	Male	57	139622	0
18842131	Male	41	134716	0
13865670	Male	57	114005	0
16924204	Male	57	133402	0
10877441	Male	43	106726	0
12430332	Male	55	90946	0
16222500	Male	48	51787	0
19088014	Male	42	138928	0
12111836	Male	50	120675	0
19889978	Male	41	113547	0
15836350	Male	34	140262	0
15448575	Male	20	54075	0
16319219	Male	36	98529	0
12475031	Male	25	121301	0
16736728	Male	41	107405	0
10975744	Male	32	87032	0

CONNECTING WITH WEB

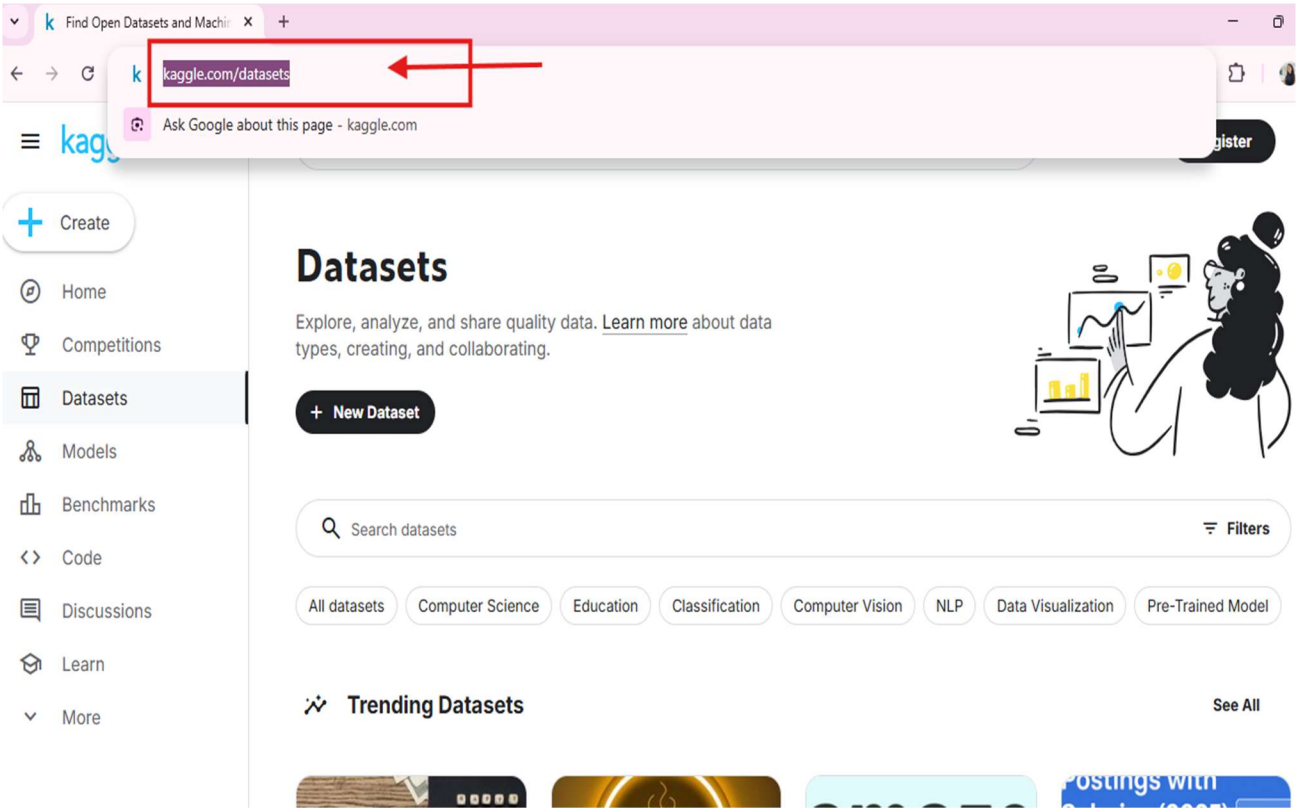
STEP 1 : GO TO THE **HOME TAB** → CLICK **GET DATA** → CHOOSE **WEB**.



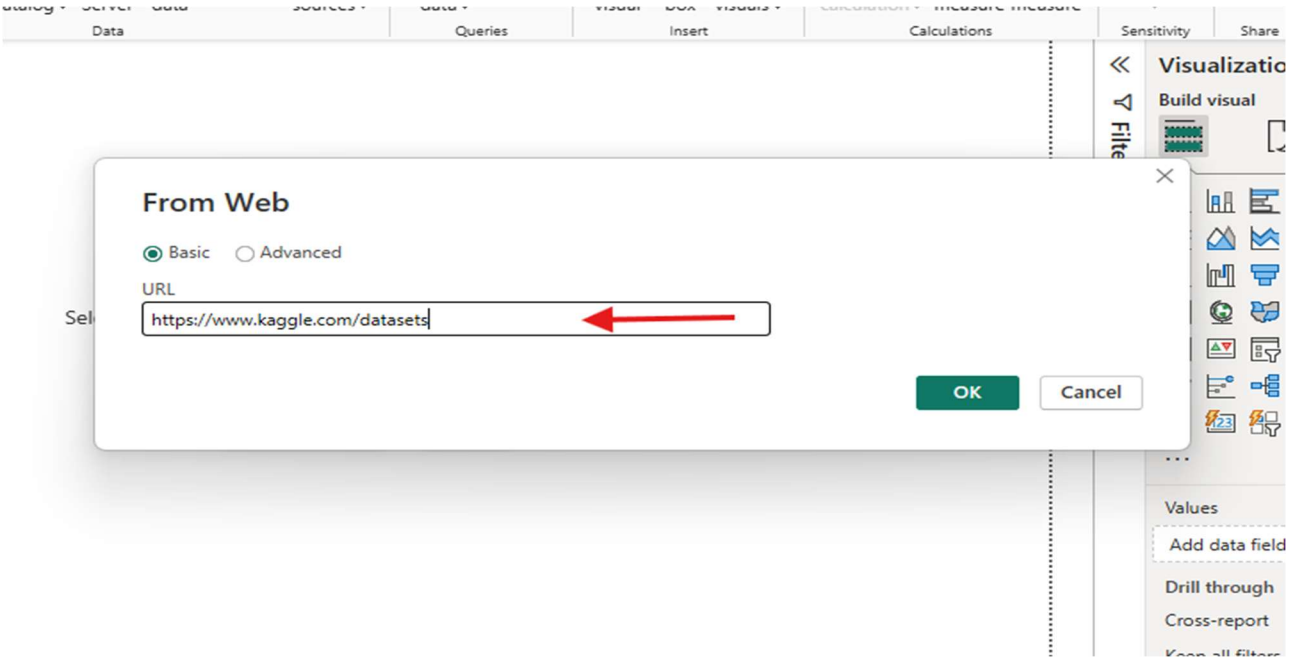
STEP 2 : A DIALOG BOX WILL APPEAR



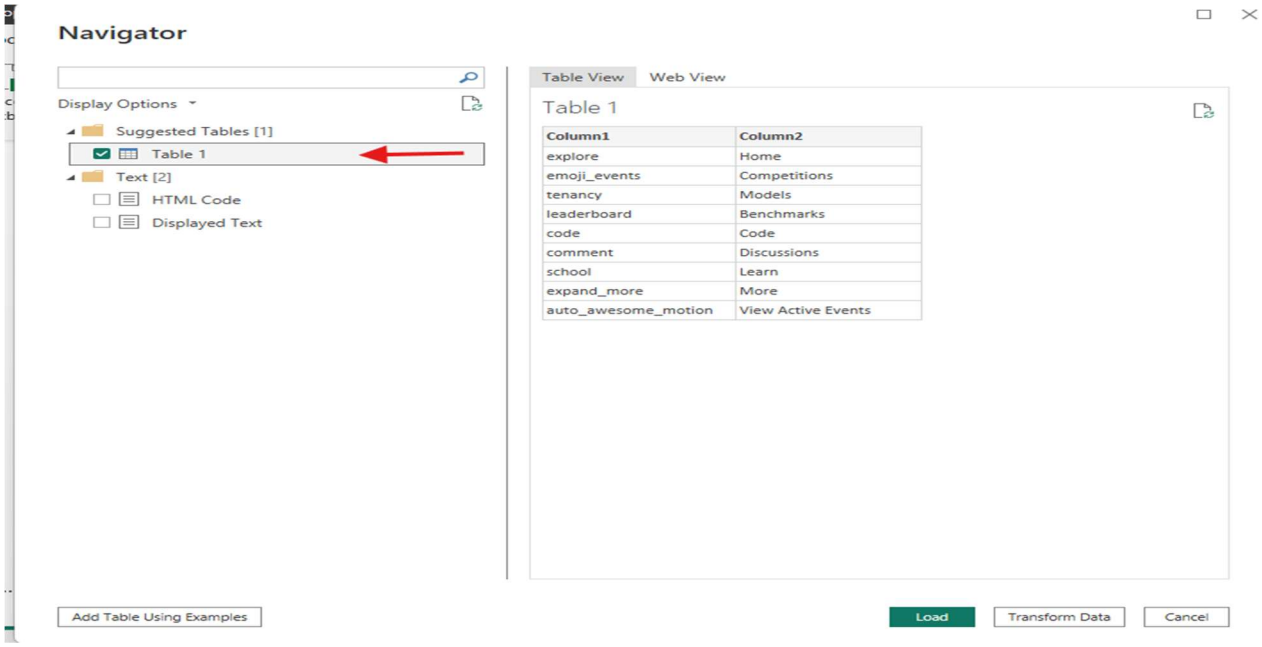
STEP 3 : OPEN ANY BROWSER AND SEARCH FOR THE DATASET YOU WANT TO LOAD INTO THE POWER BI



STEP 4 : PASTE THE WEBSITE URL THAT YOU HAVE COPIED .IF THE DATA IS PUBLIC, JUST CLICK OK.



STEP 5 : POWER BI WILL FETCH THE WEB DATA AND SHOW A PREVIEW WINDOW. CLICK LOAD → IMPORT DATA DIRECTLY.

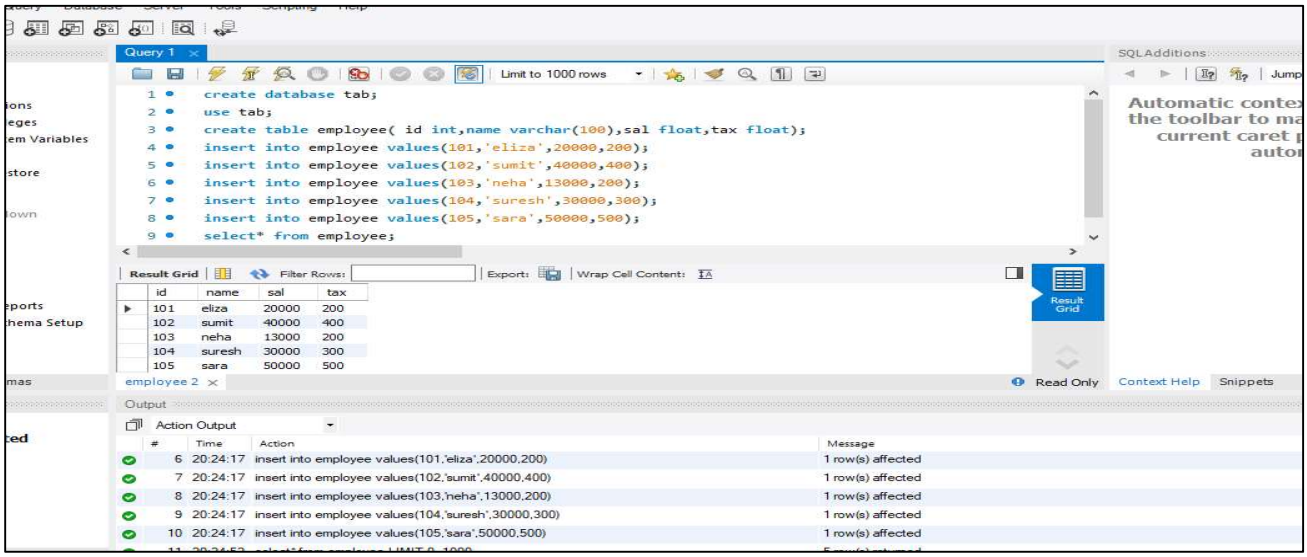


CONNECTING WITH MYSQL

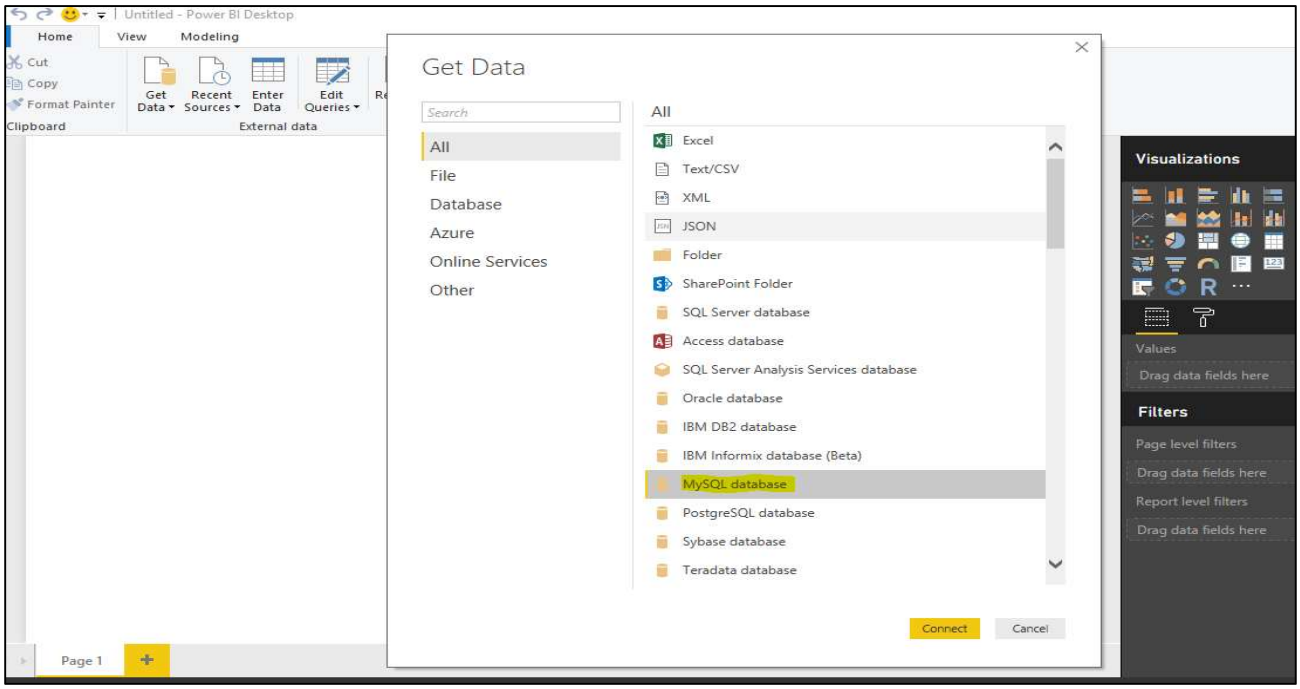
In Power BI, go to **Get Data → SQL Server**, enter your **server/database details**, choose authentication (Windows/SQL), and select **Import** or **DirectQuery**. Then pick your tables, load them, and start building reports.

EXAMPLE : HOW TO CONNECT MYSQL WITH POWER-BI

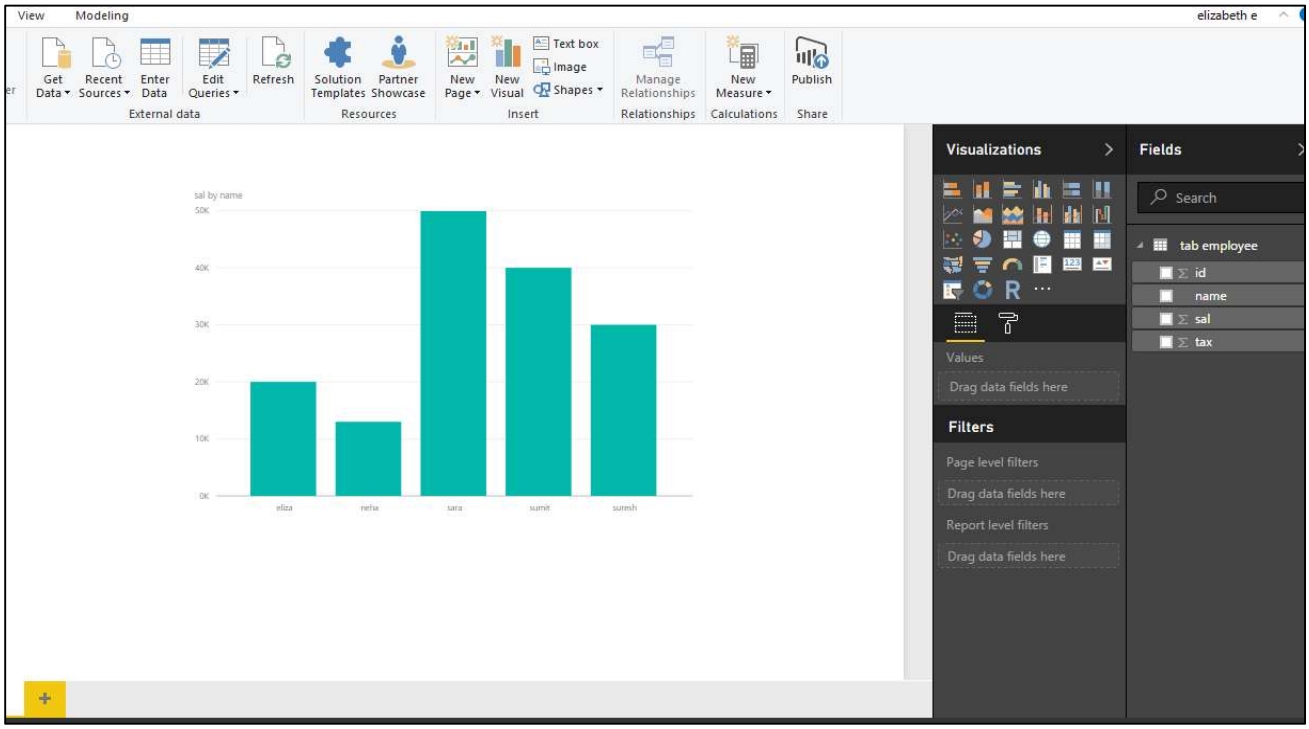
STEP 1: CREATE TABLE IN MYSQL



STEP 2: IMPORT DATA FROM MYSQL DATABASE USING GETDATA



STEP 3: CREATE VISUALIZATION

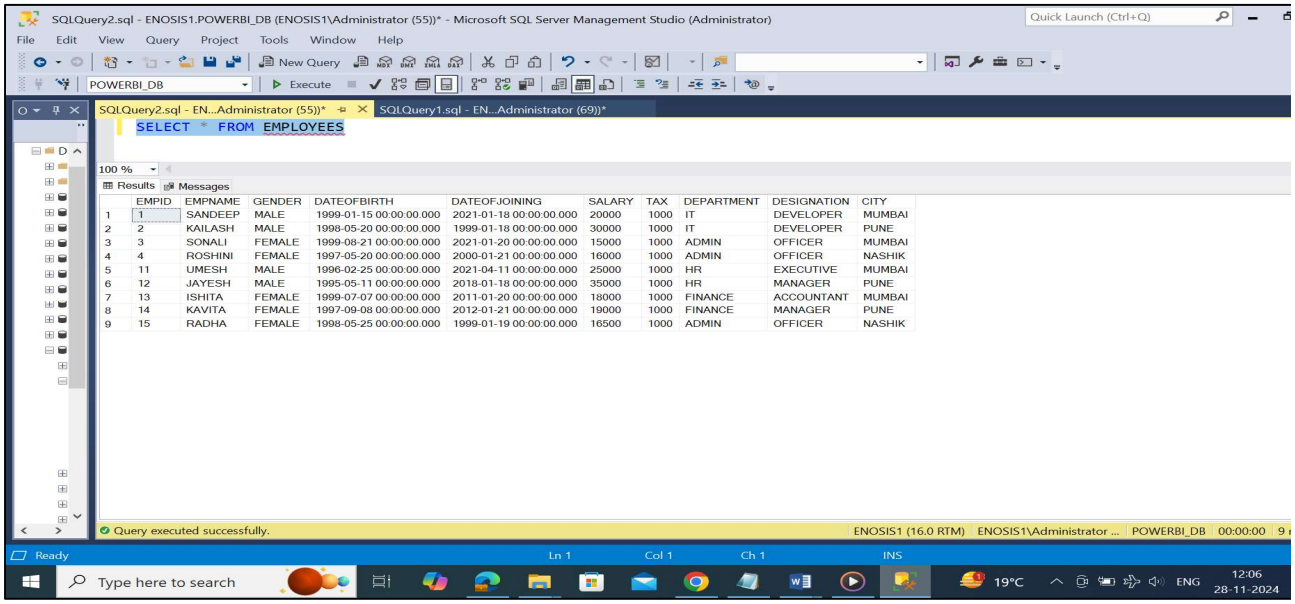


CONNECTING WITH SQL SERVER

In the example below, we will explore how to connect with SQL SERVER.

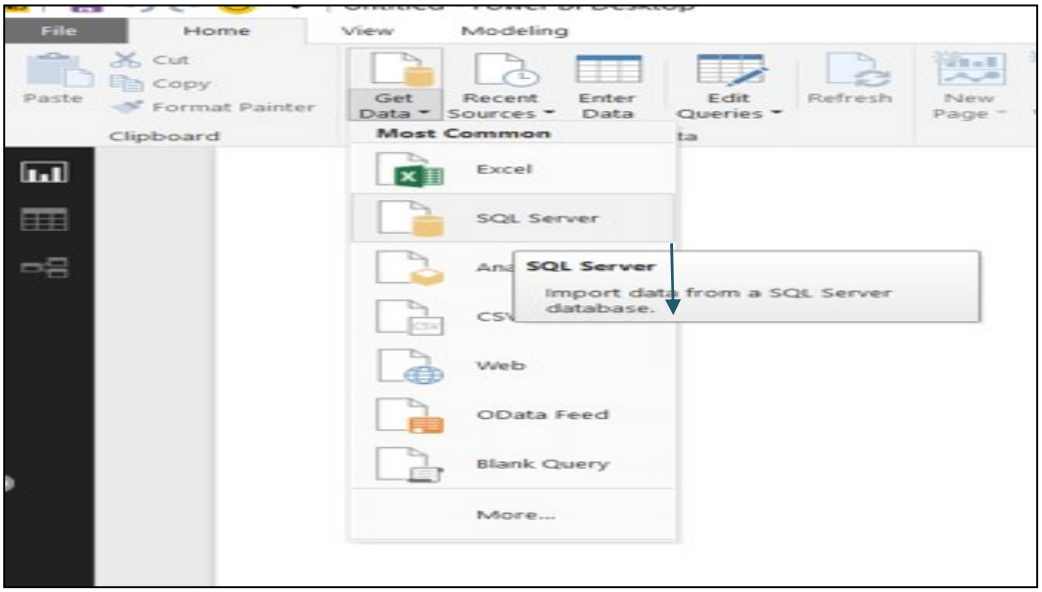
We will first run the following SQL script in a SQL database. It creates an employee table and adds few records in table.

```
CREATE TABLE EMPLOYEES
(
EMPID INT, EMPNAME VARCHAR(100), GENDER VARCHAR(10),
DATEOFBIRTH DATETIME, DATEOFJOINING DATETIME, SALARY FLOAT, TAX FLOAT,
DEPARTMENT VARCHAR(100), DESIGNATION VARCHAR(100), CITY VARCHAR(100)
)
SELECT * FROM EMPLOYEES
INSERT INTO EMPLOYEES VALUES(1,'SANDEEP','MALE','01/15/1999','01/18/2021',20000,1000,'IT','DEVELOPER','MUMBAI')
INSERT INTO EMPLOYEES VALUES(2,'KAILASH','MALE','05/20/1998','01/18/1999',30000,1000,'IT','DEVELOPER','PUNE')
INSERT INTO EMPLOYEES VALUES(3,'SONALI','FEMALE','08/21/1999','01/20/2021',15000,1000,'ADMIN','OFFICER','MUMBAI')
INSERT INTO EMPLOYEES VALUES(4,'ROSHINI','FEMALE','05/20/1997','01/21/2000',16000,1000,'ADMIN','OFFICER','NASHIK')
INSERT INTO EMPLOYEES VALUES(11,'UMESH','MALE','02/25/1996','04/11/2021',25000,1000,'HR','EXECUTIVE','MUMBAI')
INSERT INTO EMPLOYEES VALUES(12,'JAYESH','MALE','05/11/1995','01/18/2018',35000,1000,'HR','MANAGER','PUNE')
INSERT INTO EMPLOYEES
VALUES(13,'ISHITA','FEMALE','07/07/1999','01/20/2011',18000,1000,'FINANCE','ACCOUNTANT','MUMBAI')
INSERT INTO EMPLOYEES VALUES(14,'KAVITA','FEMALE','09/08/1997','01/21/2012',19000,1000,'FINANCE','MANAGER','PUNE')
INSERT INTO EMPLOYEES VALUES(15,'RADHA','FEMALE','05/25/1998','01/19/1999',16500,1000,'ADMIN','OFFICER','NASHIK')
```

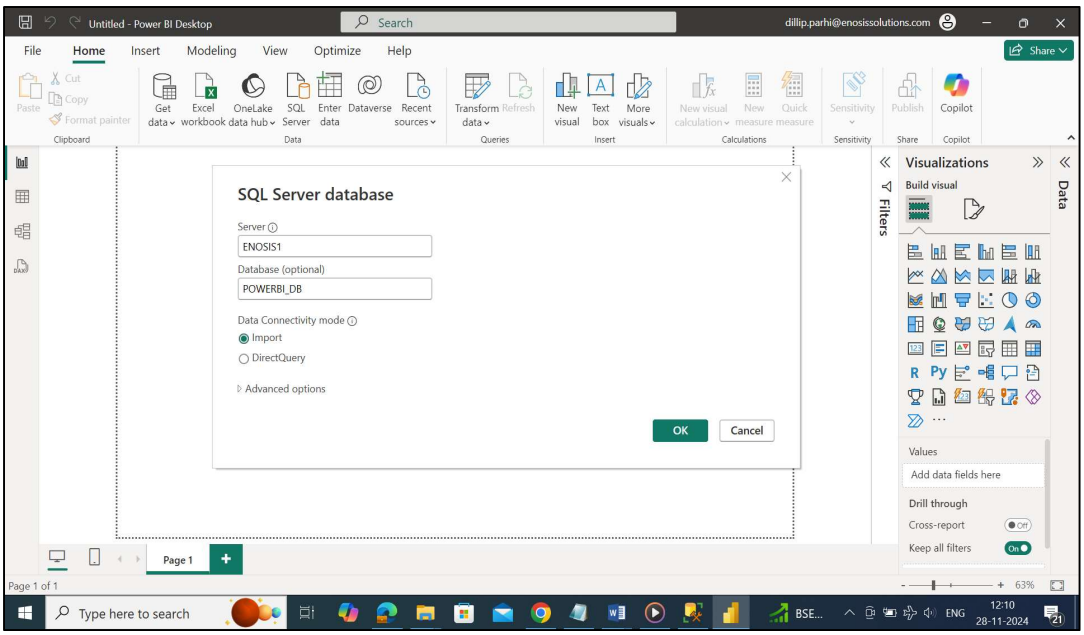


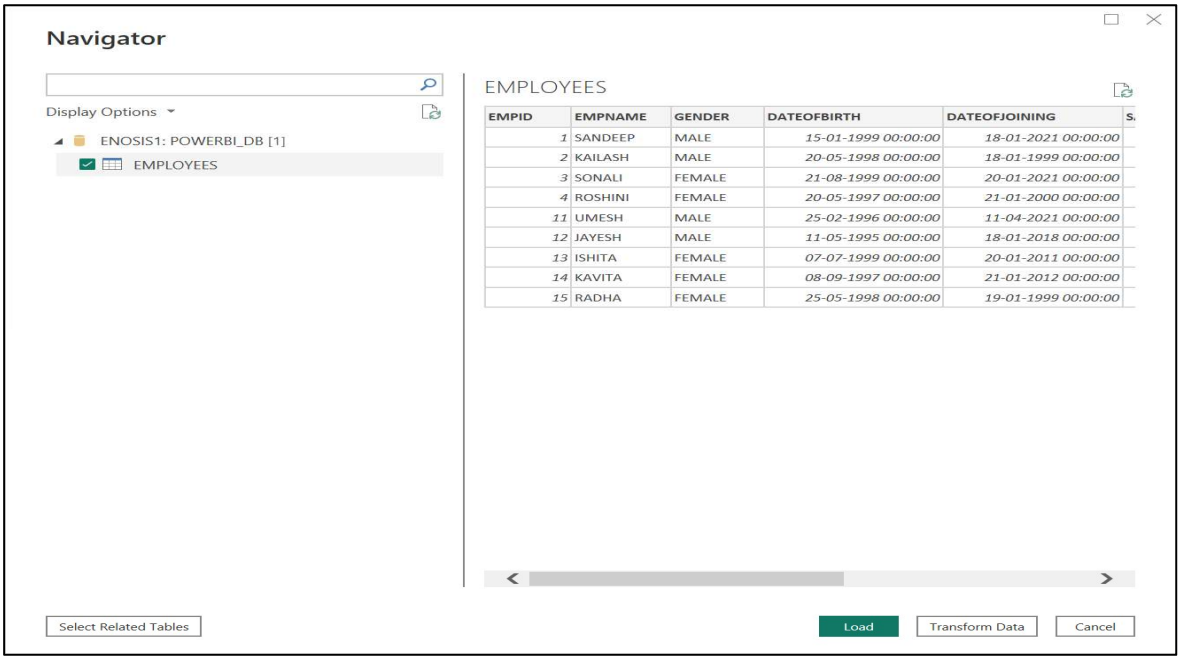
STEP 1: CONNECT WITH SQL SERVER FROM POWER BI DESKTOP

- To create a report, we need to bring our SQL SERVER data into Power BI desktop. Open Power BI desktop and click on *'Get Data'* and select *'SQL server'* as the data source.



STEP 2: CONNECT TO DATABASE IN POWER BI, AND THEN SELECT THE EMPLOYEE TABLE

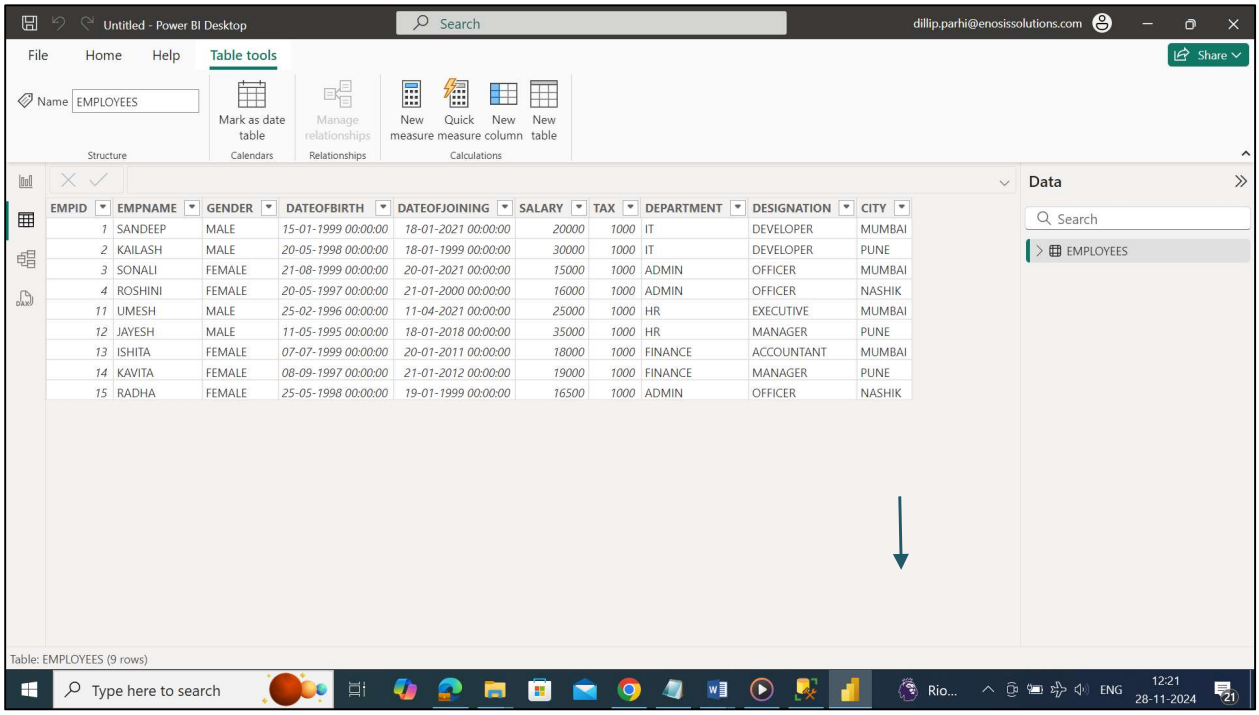




EMPID	EMPNAME	GENDER	DATEOFBIRTH	DATEOFJOINING
1	SANDEEP	MALE	15-01-1999 00:00:00	18-01-2021 00:00:00
2	KAILASH	MALE	20-05-1998 00:00:00	18-01-1999 00:00:00
3	SONALI	FEMALE	21-08-1999 00:00:00	20-01-2021 00:00:00
4	ROSHINI	FEMALE	20-05-1997 00:00:00	21-01-2000 00:00:00
11	UMESH	MALE	25-02-1996 00:00:00	11-04-2021 00:00:00
12	JAYESH	MALE	11-05-1995 00:00:00	18-01-2018 00:00:00
13	ISHITA	FEMALE	07-07-1999 00:00:00	20-01-2011 00:00:00
14	KAVITA	FEMALE	08-09-1997 00:00:00	21-01-2012 00:00:00
15	RADHA	FEMALE	25-05-1998 00:00:00	19-01-1999 00:00:00

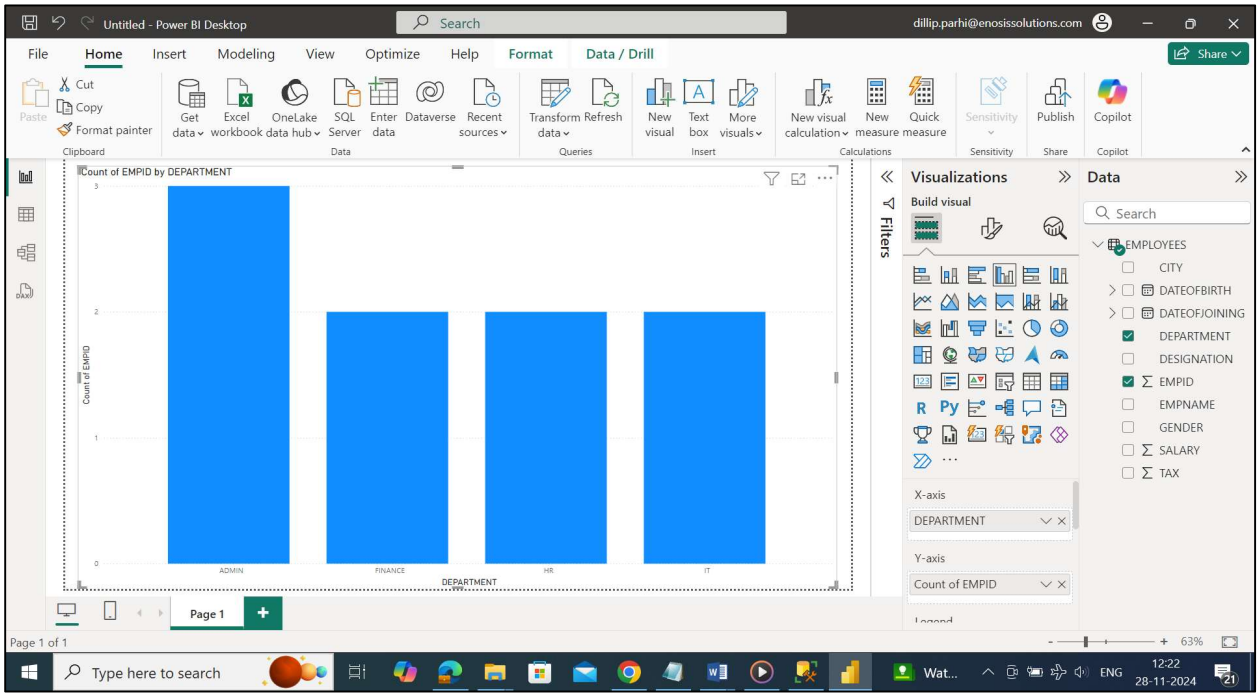
- Click on Load Button

STEP 3: AFTER THAT THE DATA IS LOADED FROM SQL SERVER INTO POWER BI APPLICATION



EMPID	EMPNAME	GENDER	DATEOFBIRTH	DATEOFJOINING	SALARY	TAX	DEPARTMENT	DESIGNATION	CITY
1	SANDEEP	MALE	15-01-1999 00:00:00	18-01-2021 00:00:00	20000	1000	IT	DEVELOPER	MUMBAI
2	KAILASH	MALE	20-05-1998 00:00:00	18-01-1999 00:00:00	30000	1000	IT	DEVELOPER	PUNE
3	SONALI	FEMALE	21-08-1999 00:00:00	20-01-2021 00:00:00	15000	1000	ADMIN	OFFICER	MUMBAI
4	ROSHINI	FEMALE	20-05-1997 00:00:00	21-01-2000 00:00:00	16000	1000	ADMIN	OFFICER	NASHIK
11	UMESH	MALE	25-02-1996 00:00:00	11-04-2021 00:00:00	25000	1000	HR	EXECUTIVE	MUMBAI
12	JAYESH	MALE	11-05-1995 00:00:00	18-01-2018 00:00:00	35000	1000	HR	MANAGER	PUNE
13	ISHITA	FEMALE	07-07-1999 00:00:00	20-01-2011 00:00:00	18000	1000	FINANCE	ACCOUNTANT	MUMBAI
14	KAVITA	FEMALE	08-09-1997 00:00:00	21-01-2012 00:00:00	19000	1000	FINANCE	MANAGER	PUNE
15	RADHA	FEMALE	25-05-1998 00:00:00	19-01-1999 00:00:00	16500	1000	ADMIN	OFFICER	NASHIK

STEP 4: NOW WE CAN CREATE A POWER BI REPORT USING VARIOUS VISUALIZATION COMPONENTS.



Note: Power BI supports connecting to multiple data sources such as Excel, CSV, Web, MySQL, and SQL Server. Each connection type allows users to import, transform, and visualize data seamlessly within Power BI Desktop. Establishing proper connections ensures accurate and up-to-date reporting. It enables integration of diverse data formats into a unified model for analysis. Understanding these connections is essential for building dynamic and reliable Power BI reports.

QUESTIONS

1.What are the common data sources supported by Power BI?
2.Explain the steps involved in connecting a data source in Power BI.
3.What is the difference between Import Mode and DirectQuery Mode in Power BI?
4.How does Power BI help users combine multiple data sources into a unified dataset?
5.Explain how Microsoft SQL Server and MySQL can be connected with Power BI.

SUMMARY

Power BI provides extensive connectivity to a wide range of **data sources**, allowing seamless integration between **on-premises systems and cloud-based services**. Users can connect to data from **Excel, CSV, Web, SQL Server, and MySQL**, among many others. The connection process typically involves selecting *Get Data*, choosing the source, browsing or entering a path/URL, previewing the dataset, and loading it into Power BI for visualization. Power BI supports two main connection modes: **Import Mode**, which stores a snapshot of the data for offline analysis, and **DirectQuery Mode**, which enables real-time data access directly from the source without importing. This flexibility allows users to blend multiple sources—such as spreadsheets, databases, and online feeds—into a single, unified dataset. By offering secure, fast, and versatile data integration, Power BI simplifies the process of transforming raw, scattered data into insightful, visual reports that drive business decisions.

Power BI connects seamlessly with various data sources, enabling integration across cloud and on-premises systems. Data sources include Excel, CSV, Web data, SQL Server, and MySQL.

Common connection steps:

1. Go to Get Data → Choose Source (Excel, Web, etc.).
2. Browse or enter the path/URL.
3. Preview the dataset and click Load.
4. Start creating visuals using imported data.

Examples:

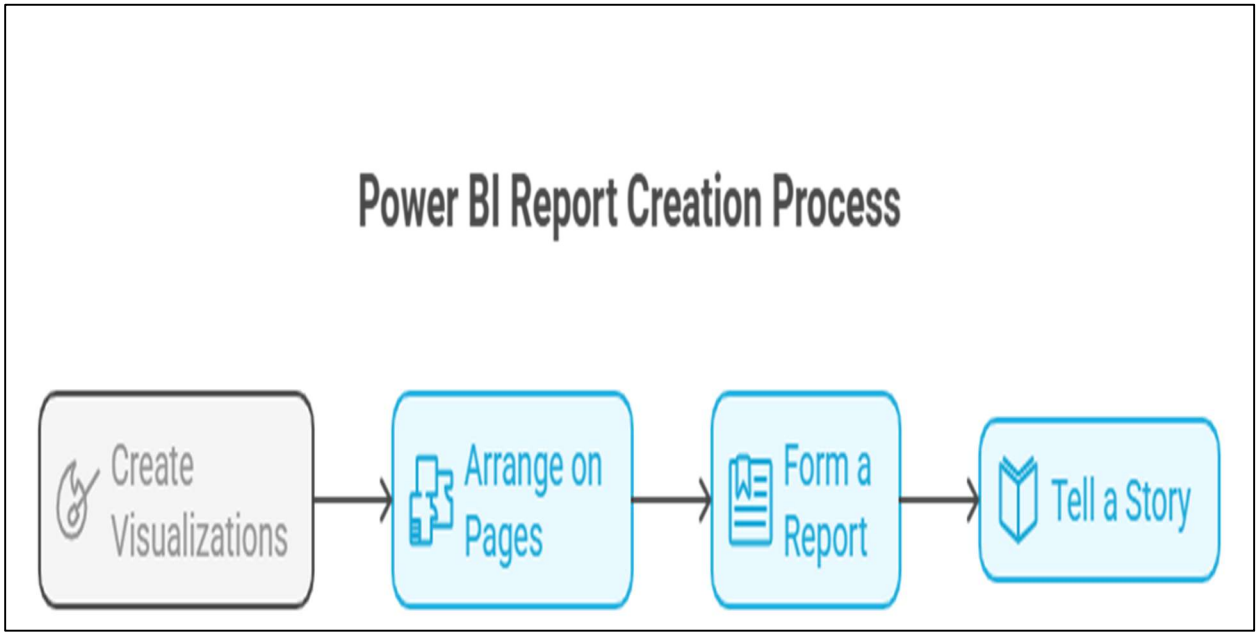
- Excel – Import worksheets directly into Power BI.
- CSV – Import text-based data with delimiters.
- Web – Connect to web-based datasets via URLs.
- SQL Server/MySQL – Connect using credentials and load tables.

Power BI supports both Import Mode (for offline analysis) and DirectQuery Mode (for real-time connections). These options give flexibility for combining multiple data sources into unified datasets for reporting.

CHAPTER 4 : POWER BI REPORTS

Power BI Tool is primarily used to create reports. We load the data from various sources and clean it and then build reports from the data. The IDE (Power BI Desktop software) is used to create visualization.

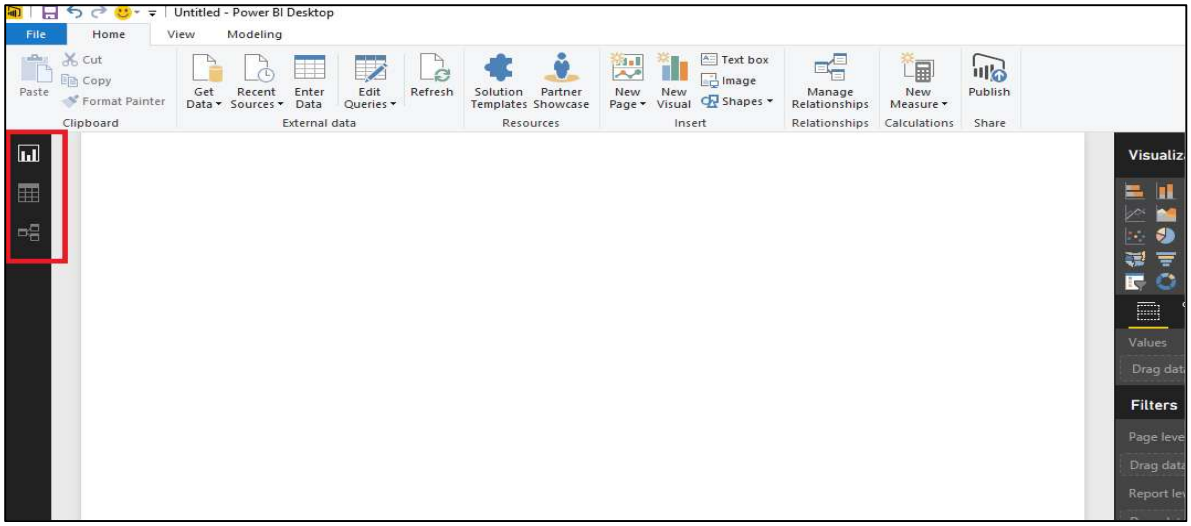
Power BI Report is PBIX file. The Report file consists of Power BI pages, where we add different types of visualization.



CREATING A REPORT USING POWER BI

Note: For this you have to first download and install Power BI Desktop

The image below shows how a Power BI Desktop's interface looks. The highlighted section in blue color, on the left panel shows the report, data and relationship workspaces. By default, the *report* workspace will open. This is where you create reports. Below the reports workspace is the data workspace which is used to see the imported data sets. Last tab is the relations tab which gives you relationship between different variables in a data set, if they are well defined. On the right side, you will see visualizations and field workspace.



STEPS FOR CREATING POWER BI REPORTS

STEP 1	OPEN POWER BI DESKTOP AND CREATE A NEW POWER BI REPORT
STEP 2	IMPORT DATA AND LOAD THE DATA INTO POWER BI
STEP 3	VIEW THE DATA
STEP 4	ADD VISUALIZATION AND BIND THE DATA WITH THE VISUALIZATION
STEP 5	SAVE THE REPORT IN A PARTICULAR DRIVE OR LOCATION

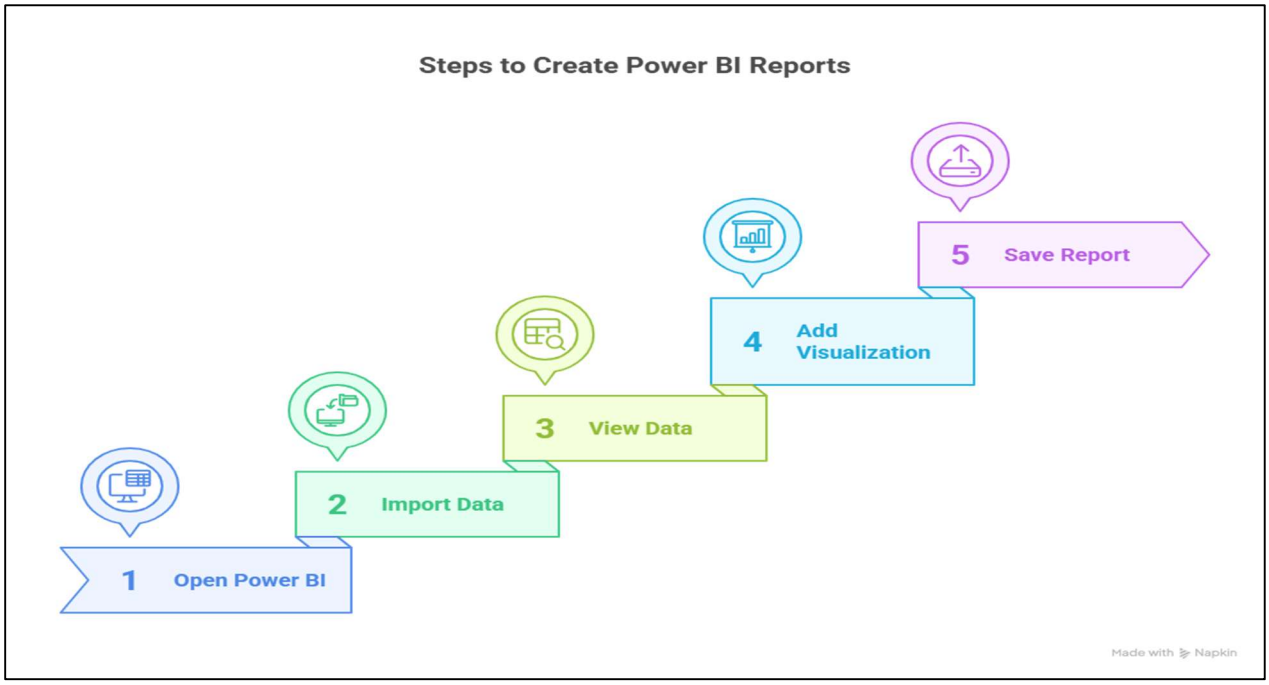
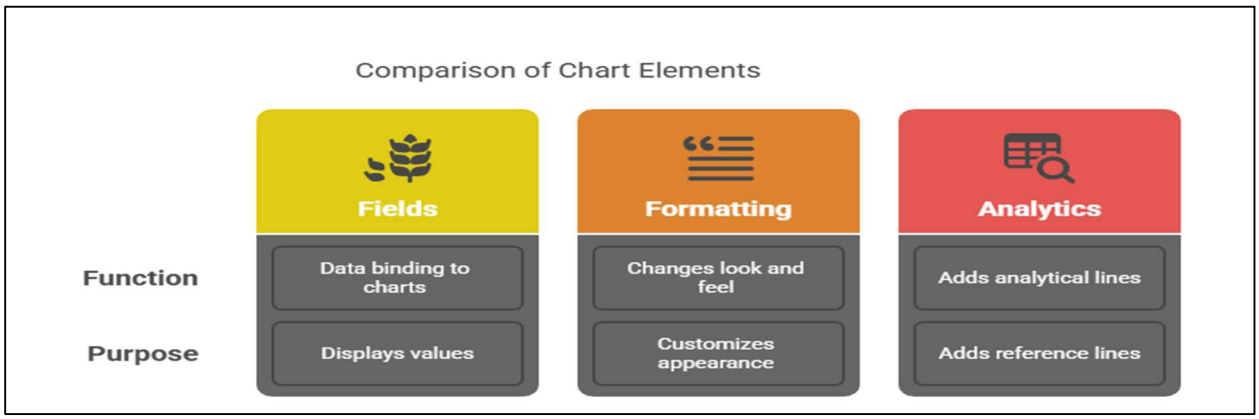


CHART FEATURES

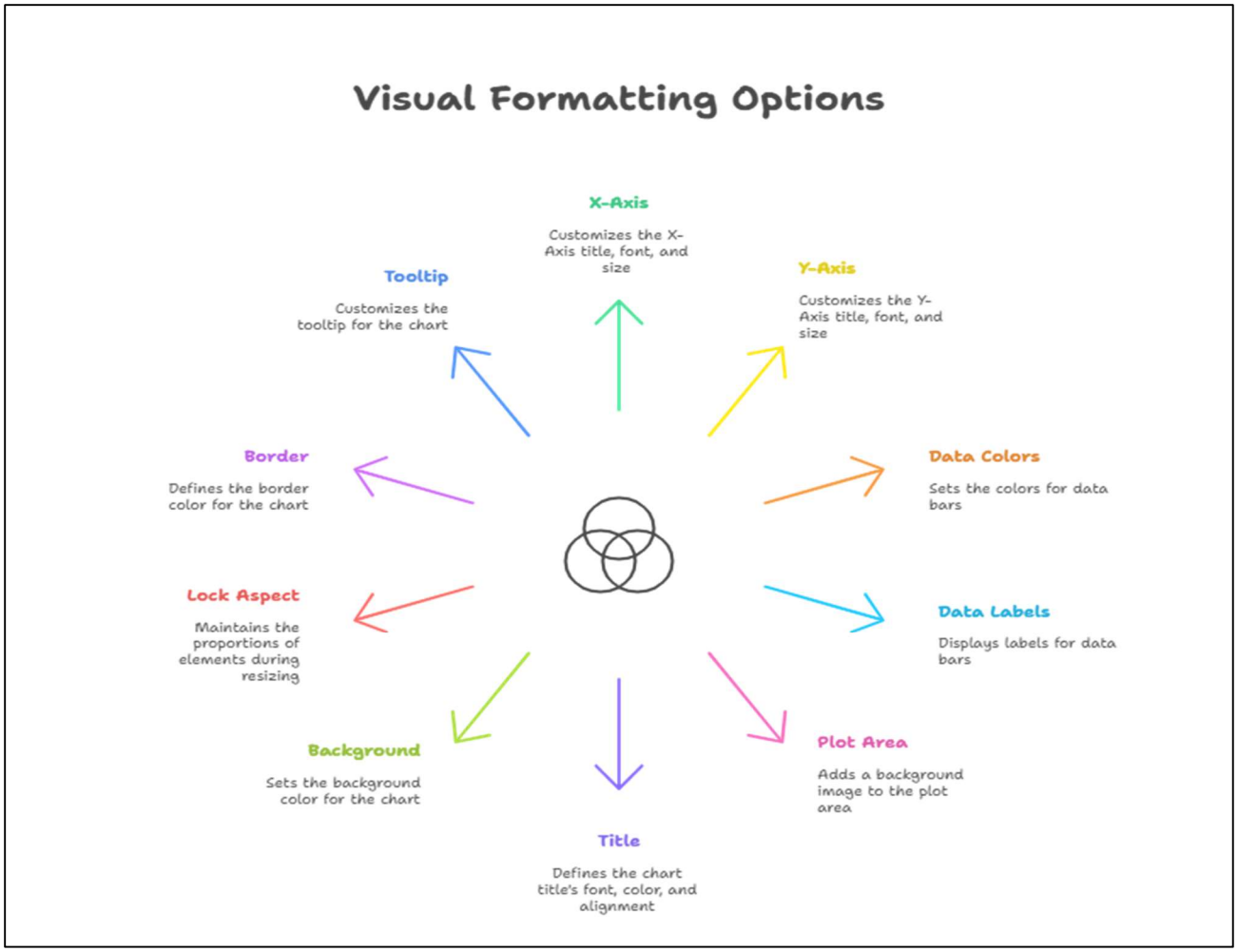
FIELDS	FORMATTING	ANALYTICS
The values of fields /columns are binded to the charts as data. Based upon the fields attached to the visualization it will display the values accordingly	The formatting option will change the look and feel of the visualization. For example, back color, forecolor, axis labels, titles can be changed using formation options	The Analytics section is used to add different analytics line to the charts like constant line, min line,max line, median line etc.



The following are the common features of Each Visualization in Power BI

FORMATS

GENERAL	Define the X-Position, Y-Position
X-AXIS	Defines the X-Axis features like x –axis title, font type, font size etc.
Y- AXIS	Defines the X-Axis features like Y –axis title, font type, font size etc.
DATA – COLORS	Show the colors of the Data bars
DATA – LABELS	Show the labels for the Data bars
PLOT AREA	Add a Image in the background
TITLE	Define the Title for the Charts. Defines the Font color, font size, alignment for the Title
BACKGROUND	Defines the Background color for the chart.
LOCK ASPECT	It locks the proportions of an element - so if you resize - it will keep the same proportions.
BORDER	Defines the border color for the chart.
TOOLTIP	Define the tooltip for the chart.

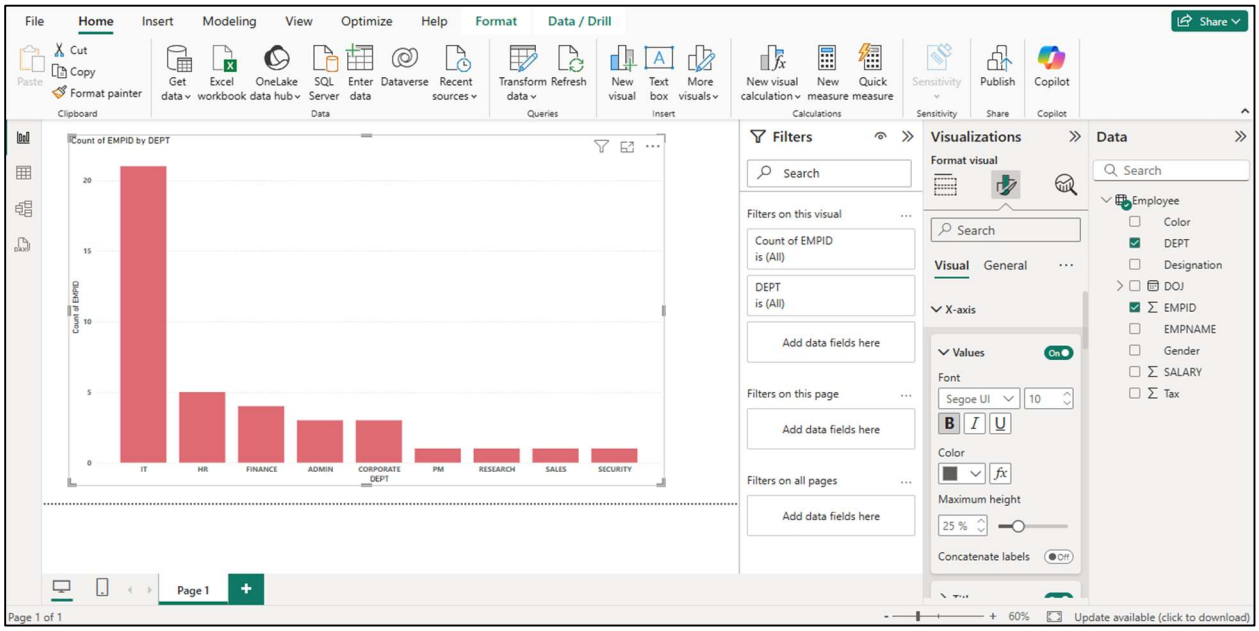


ANALYTICS

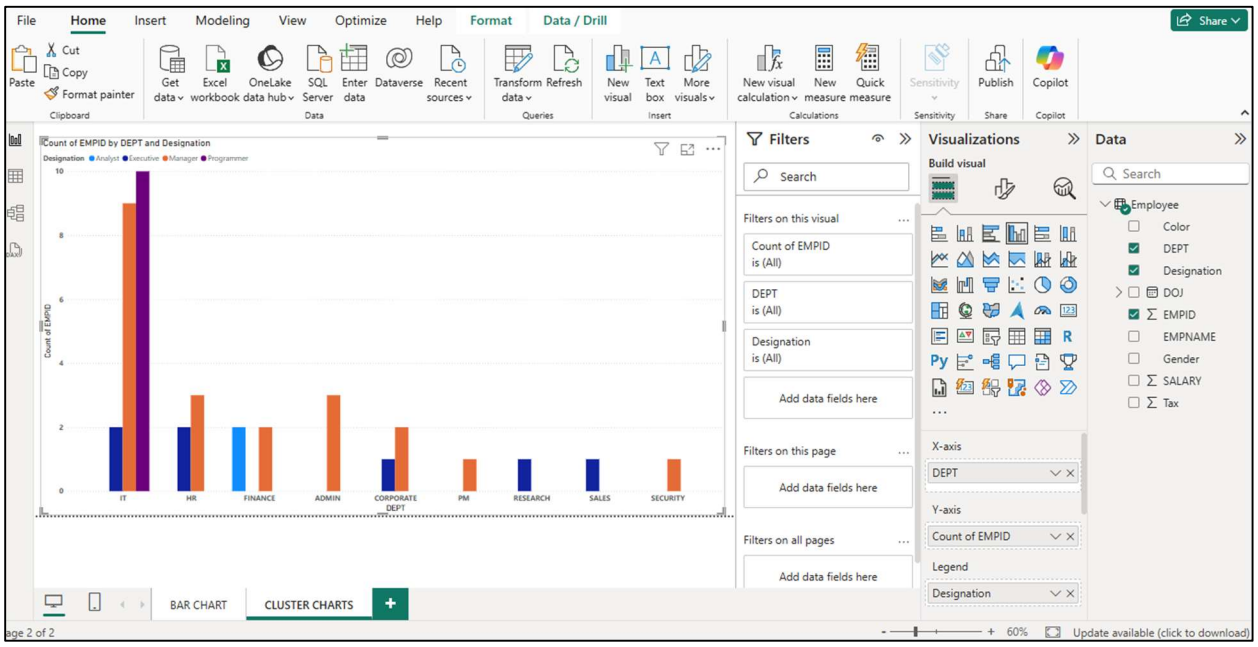
CONSTANT LINE	Constant line in the analytics pane can be used to create a reference line for a set of visualizations.
MIN LINE	Defines the minimum line for the chart
MAX LINE	Defines the maximum line for the chart.
AVERAGE LINE	Defines the average line for the chart.
MEDIAN LINE	Defines the median line for the chart.

AXIS , Y - AXIS & LEGEND

- Legends divided the X – Axis



- In the chart, lets add a Legend designation and the chart will display as shown below:

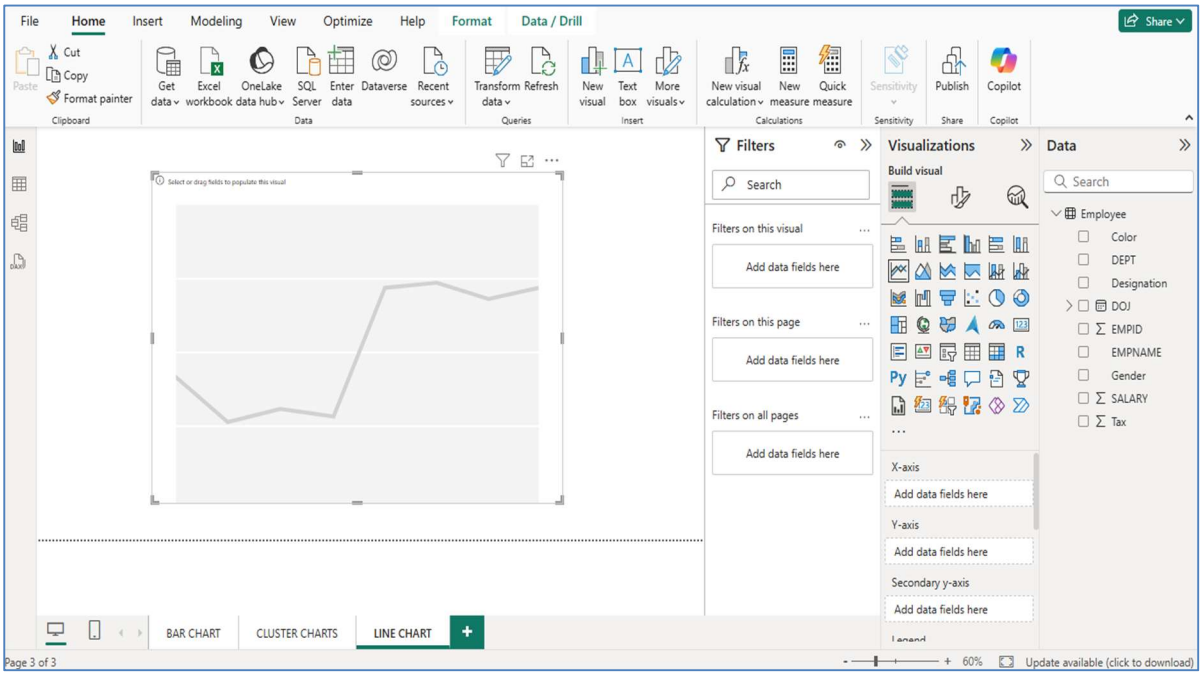


Note: Chart features in Power BI enhance data interpretation through formatting and analytics options. Users can customize visuals with colors, labels, and titles for better presentation. Analytics tools like constant, average, and median lines help highlight key insights within the data.

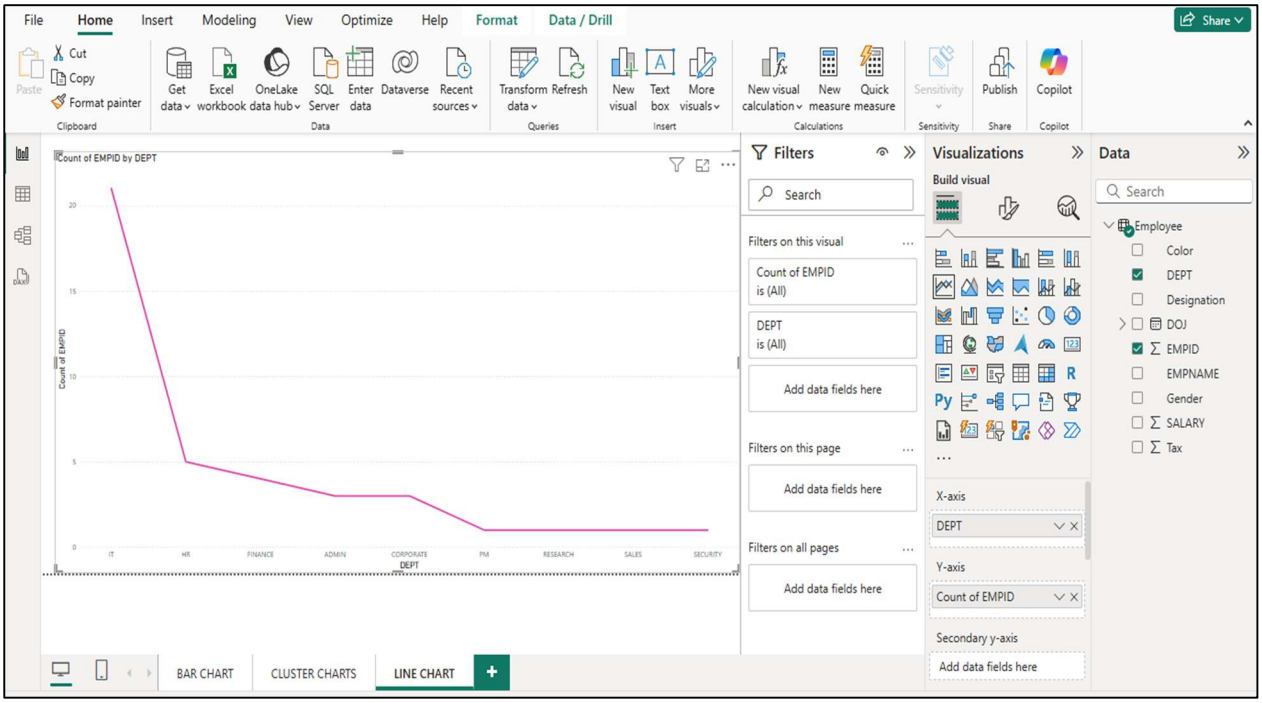
TYPES OF CHARTS/VISUALIZATIONS IN POWER BI

LINE CHART : A Line is created for each value specified.

- Go to the visualization pane and add the Line Chart from the pane.

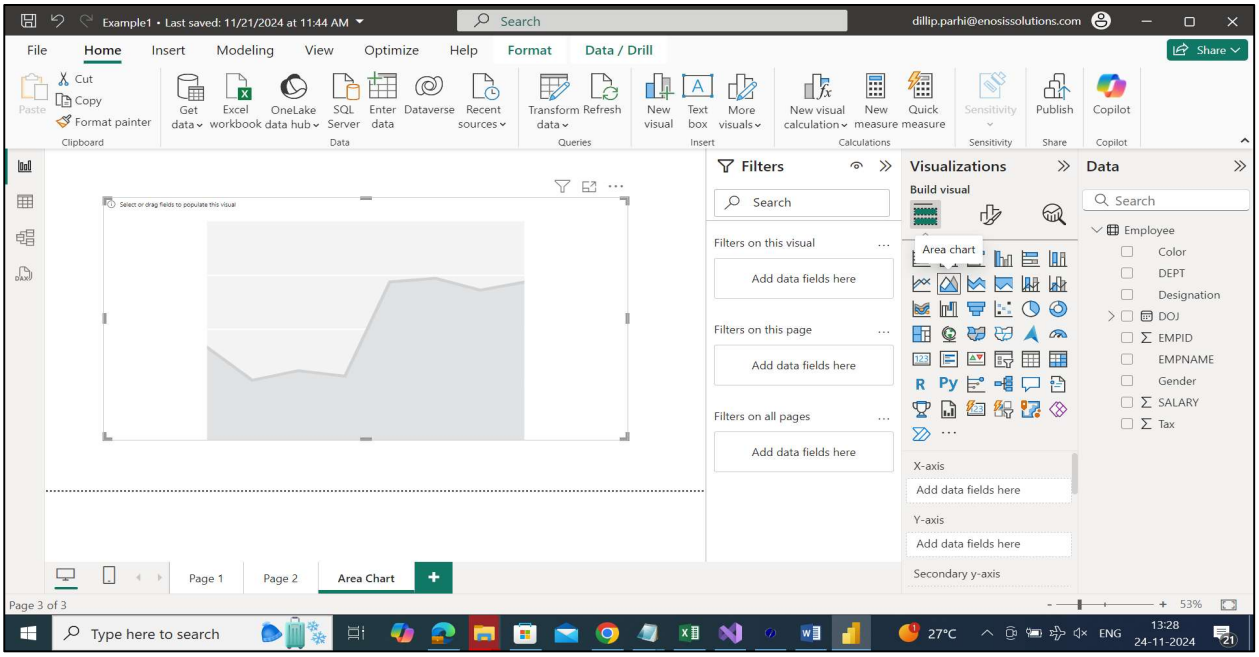


- Bind the Data, Dept. in the X – Axis and EMPID in the Y – Axis

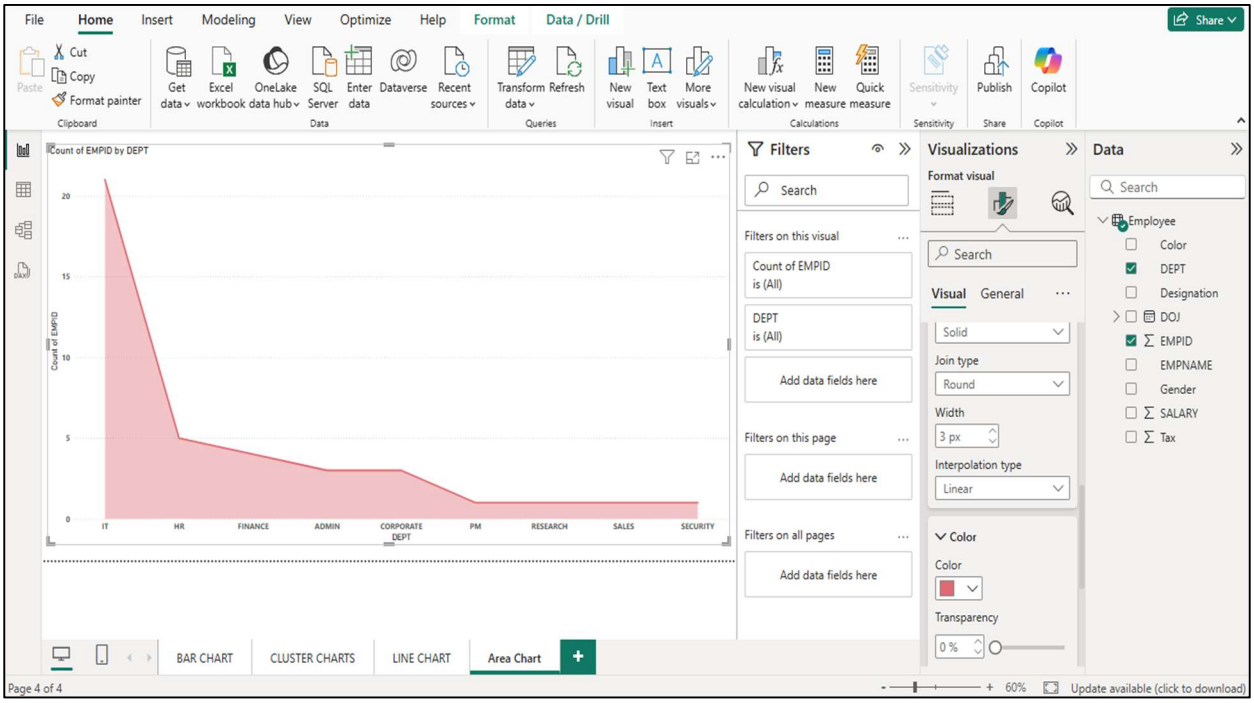


AREA CHART : An area chart displays graphically quantitative data.

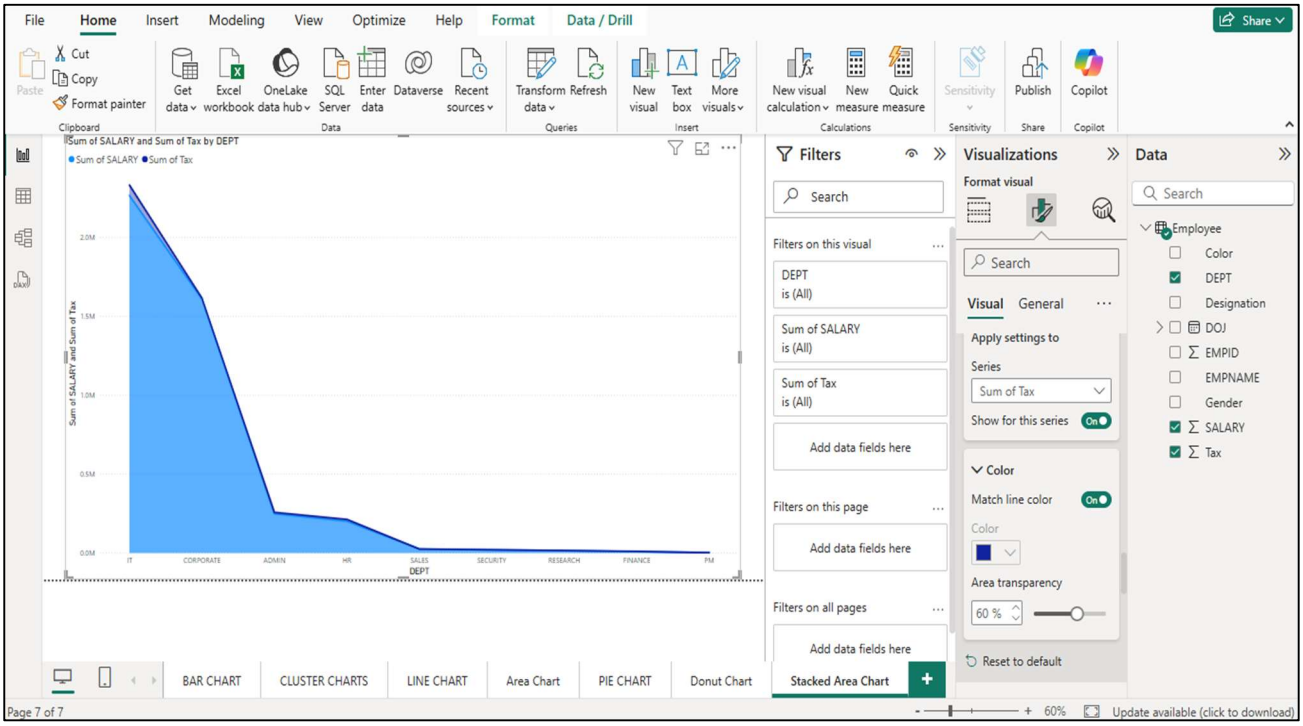
- Go to the Visualization Pane and add the Area Chart



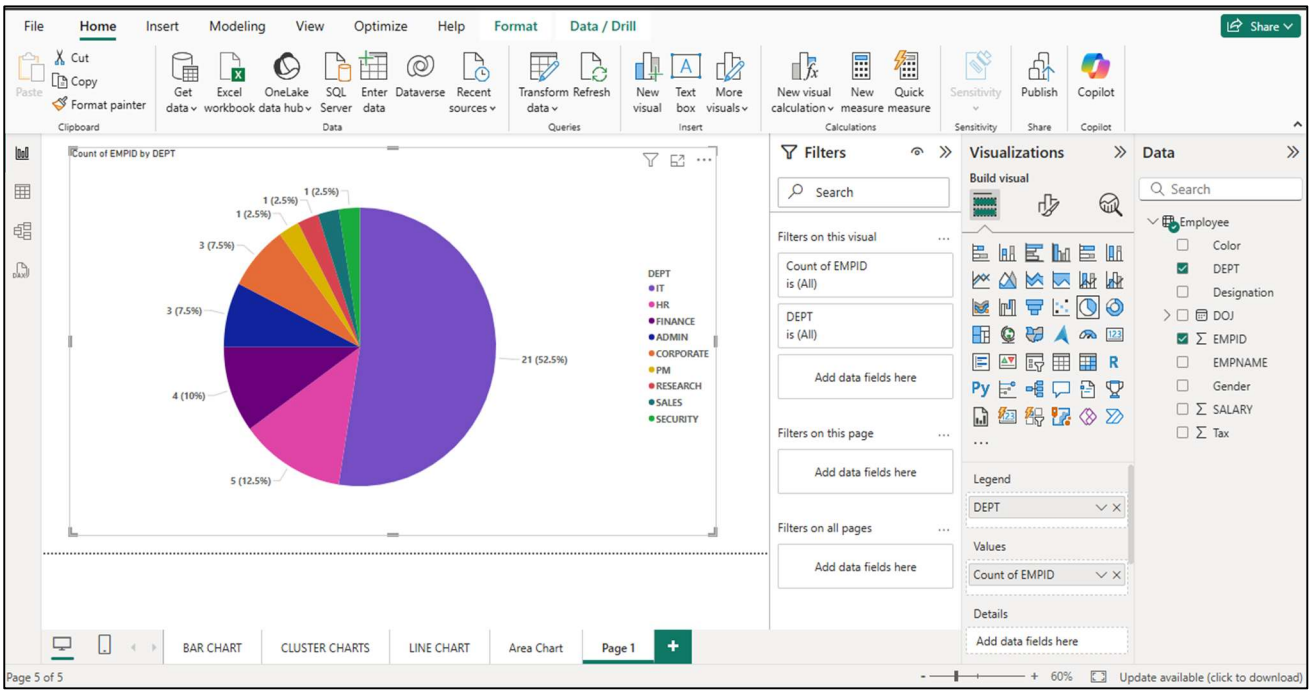
- Bind the Data, Dept. in the X – Axis and EmpId in the Y – Axis



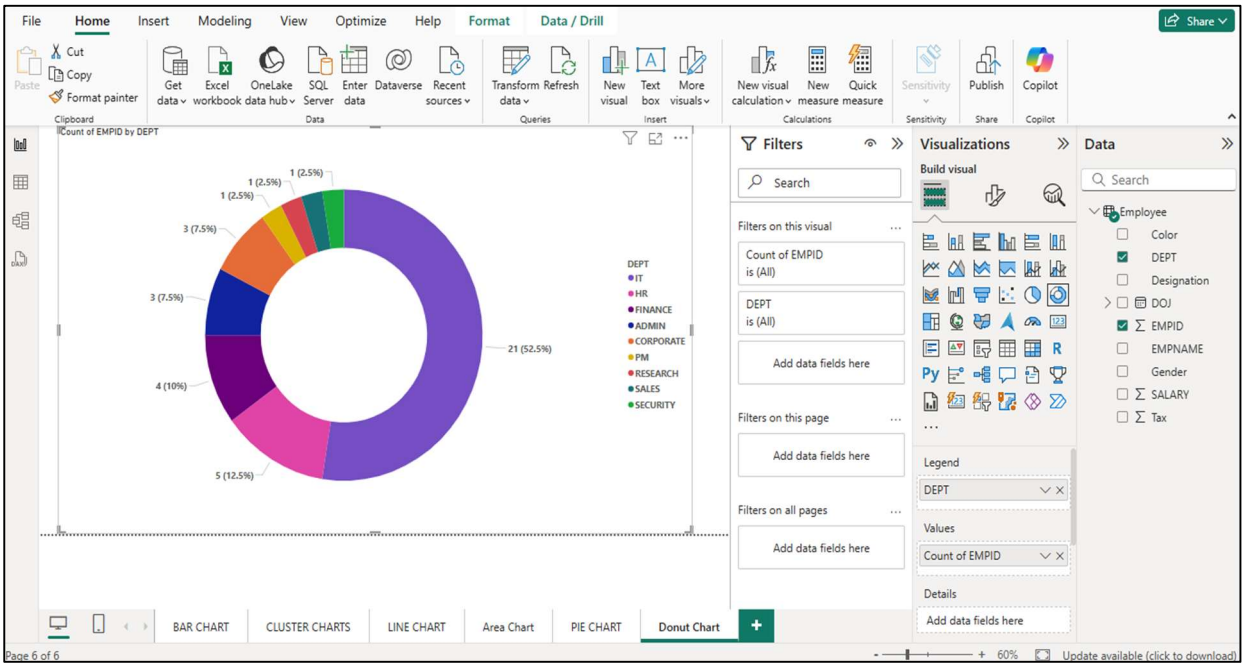
STACKED AREA CHART: A stacked area chart is the extension of a basic area chart to display the evolution of the value of several groups on the same graphic.



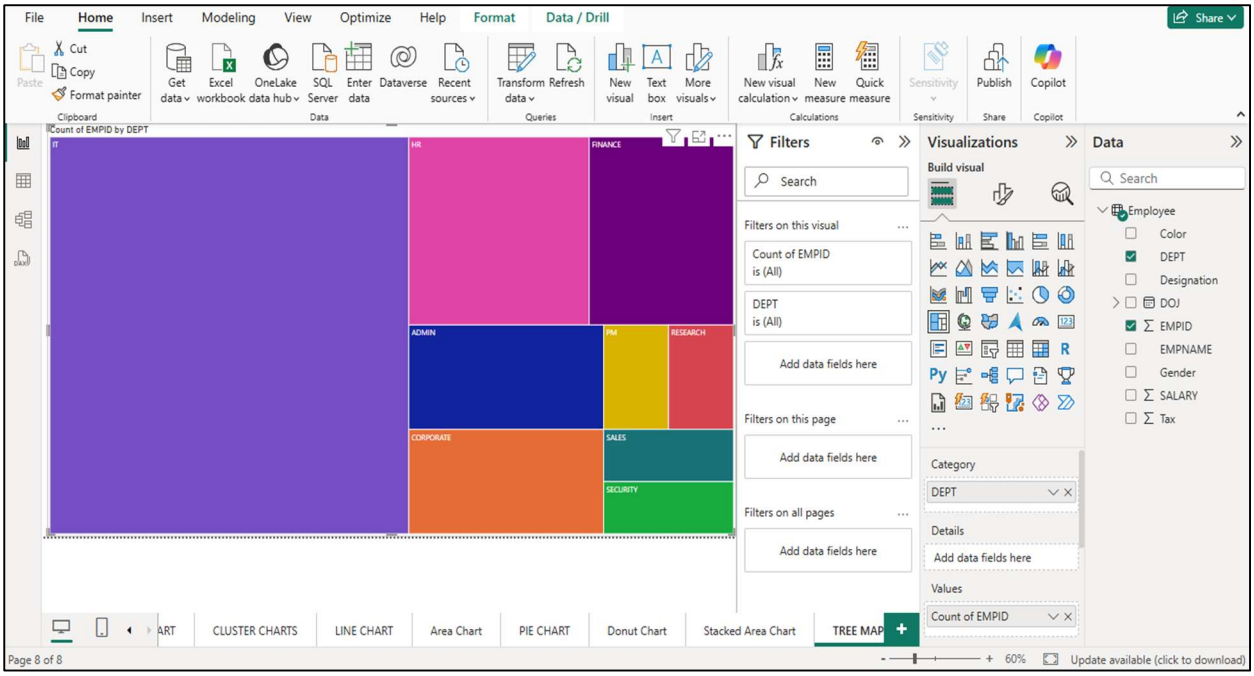
PIE CHART : The purpose of a Pie chart is to illustrate the contribution of different values to a total.



DONUT CHART: A doughnut chart is similar to a pie chart. The only difference is that the center is blank and allows space for a label or icon.

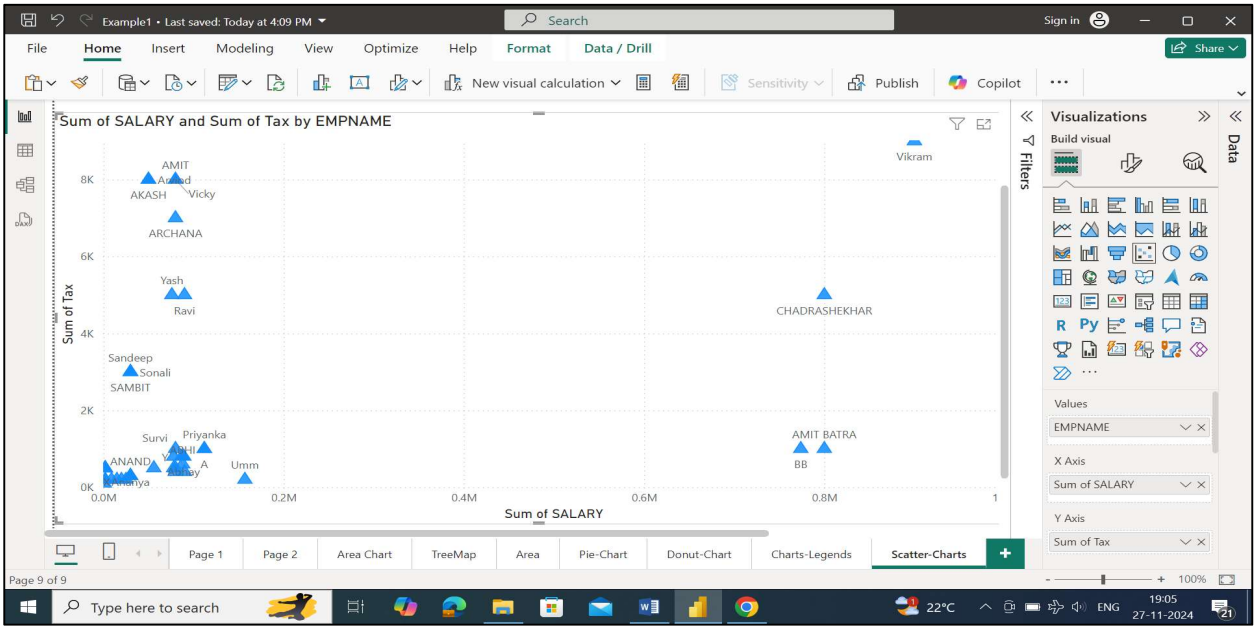


TREEMAP: **Power BI Treemap** is used to display the data in Rectangle Boxes. The **Tree maps in Power BI** are very useful to display largest data in small region.

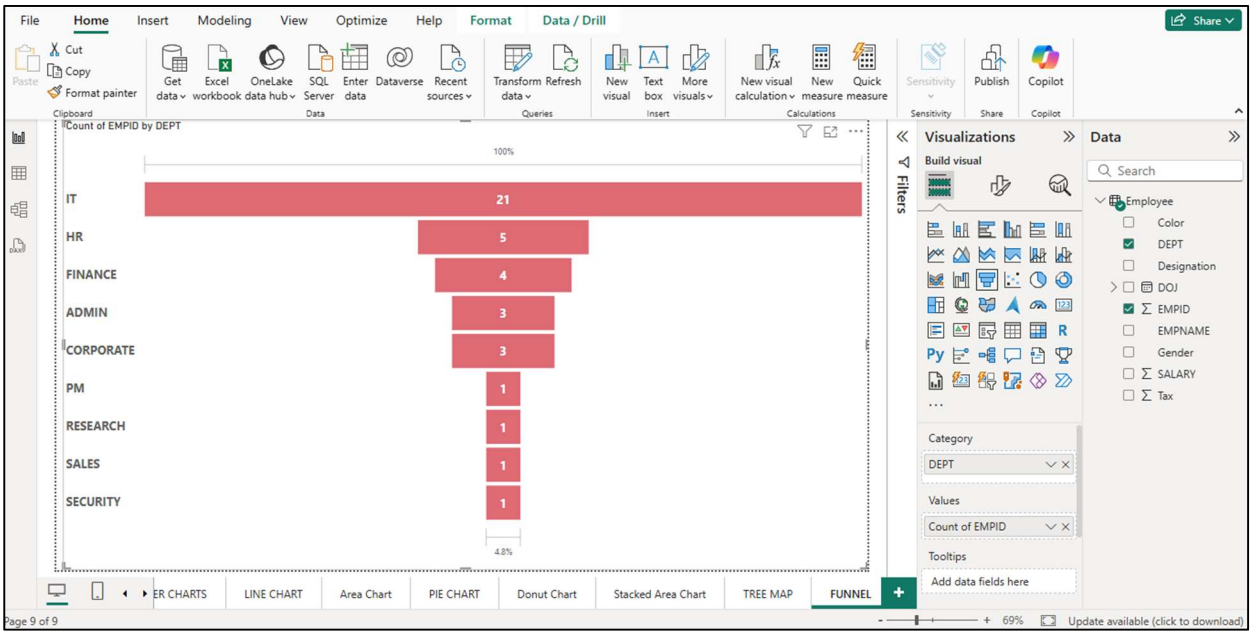


SCATTER CHART: A scatter chart always has two value axes to show: one set of numerical data along a horizontal axis and another set of numerical values along a vertical axis.

- The **chart** displays points at the intersection of an x and y numerical value, combining these values into single data points.



FUNNEL: A funnel chart helps you visualize a linear process that has sequential connected stages



QUESTIONS

1.What are Power BI Reports and why are they important in data analysis?
2.Explain the steps involved in creating a report in Power BI Desktop.
3.What is the purpose of the Formatting and Analytics panes in Power BI?
4.Differentiate between Pie Chart, Donut Chart, Scatter Chart, and Funnel Chart in Power BI.
5.How do Power BI Reports help organizations make data-driven decisions?

SUMMARY

Power BI **reports** are the main outputs of data analysis—dynamic visual representations that turn raw data into meaningful insights. Each report, saved as a **.PBIX file**, can contain multiple pages filled with interactive visuals, charts, and KPIs. Reports are created in **Power BI Desktop** by connecting to data sources, adding visuals, customizing them with formatting tools, and then publishing them to the **Power BI Service** for sharing. Users can apply formatting options to modify axis labels, colors, titles, and backgrounds, enhancing readability and visual appeal. The **Analytics pane** allows the addition of constant, average, or percentile lines for deeper analysis. Power BI supports various chart types—**Line, Area, Pie, Donut, Treemap, Scatter, and Funnel**—to highlight patterns, trends, and relationships effectively. Overall, reports in Power BI combine design flexibility, interactivity, and storytelling, enabling users to explore data dynamically and make well-informed, data-driven decisions.

Reports are the core deliverables in Power BI — visual representations of data insights. Each report is saved as a .PBIX file and can contain multiple pages of visuals.

Steps to create a report:

1. Load data from sources.
2. Build visuals by binding data fields.
3. Customize using formatting and analytics options.
4. Save and publish to Power BI Service.

Common chart types include Line, Area, Pie, Donut, Treemap, Scatter, and Funnel charts. These visuals help users analyze trends, patterns, and comparisons effectively.

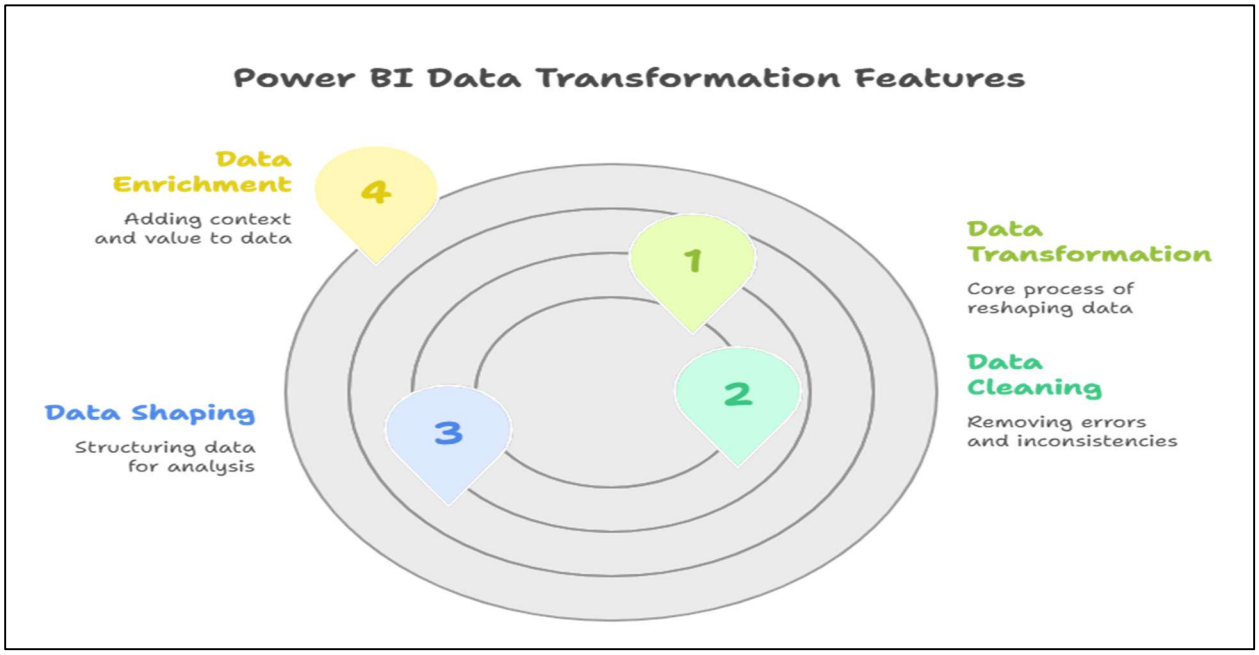
Formatting options:

- Adjust X-axis, Y-axis, colors, and labels.
- Add background, title, and tooltips.
- Use the Analytics Pane for constant lines, averages, and percentiles.

Reports in Power BI transform raw data into actionable insights by offering flexibility in design, interactivity, and storytelling — empowering users to make data-driven decisions.

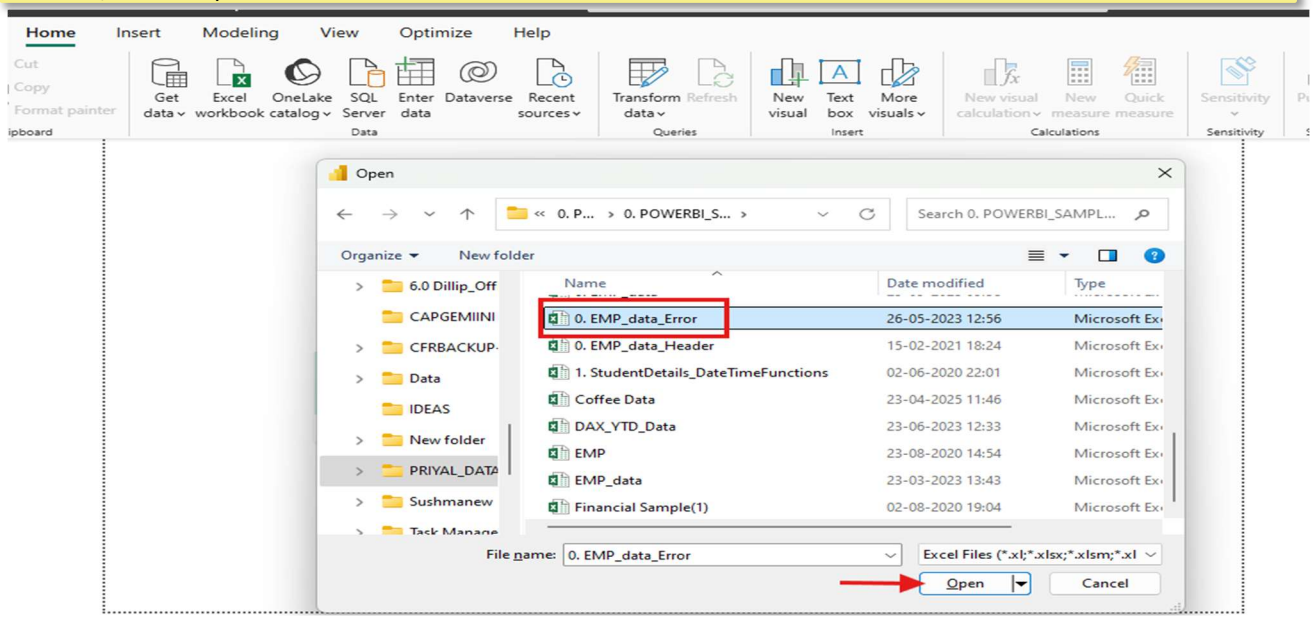
CHAPTER 5: POWER BI - DATA TRANSFORMATION

In this chapter, we will explore Data Transformation features of Power BI

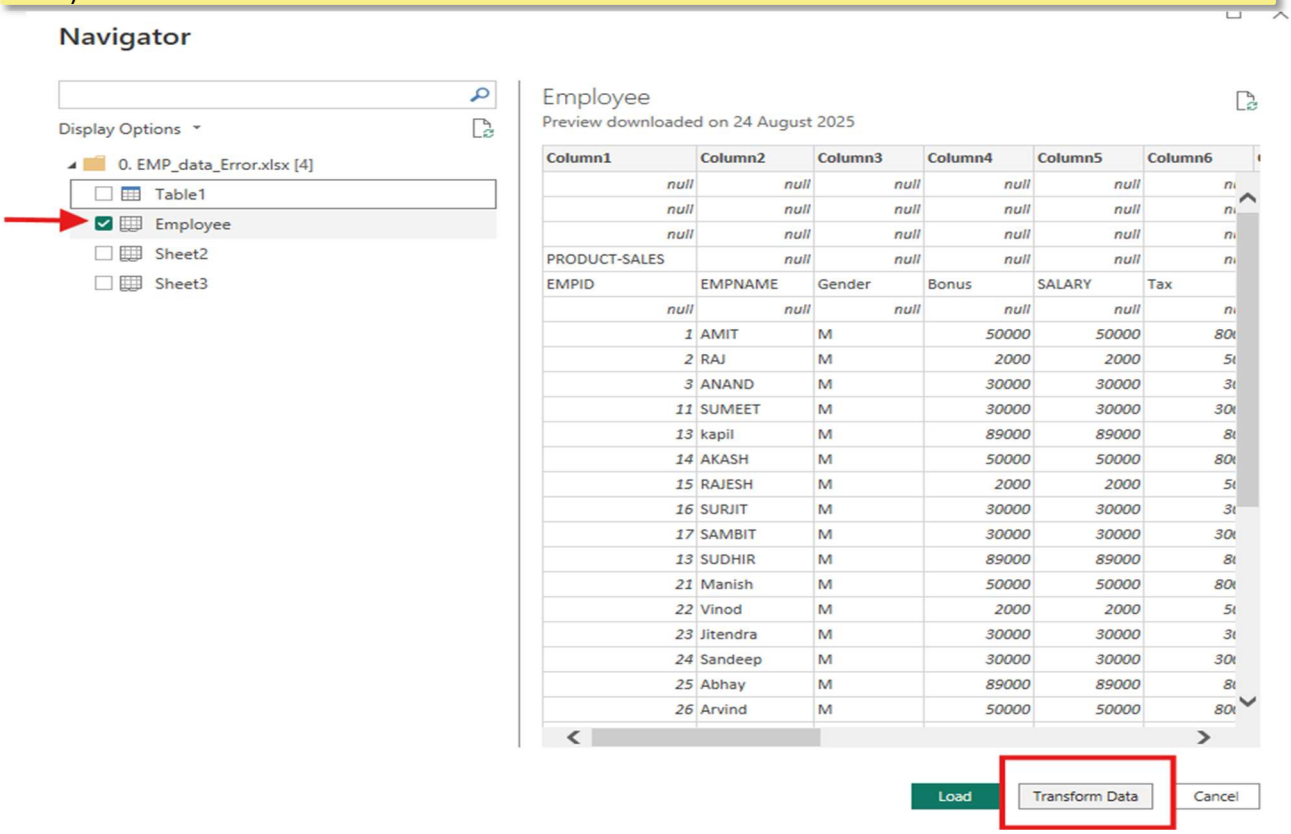


STEPS TO OPEN POWER QUERY EDITOR IN POWER BI

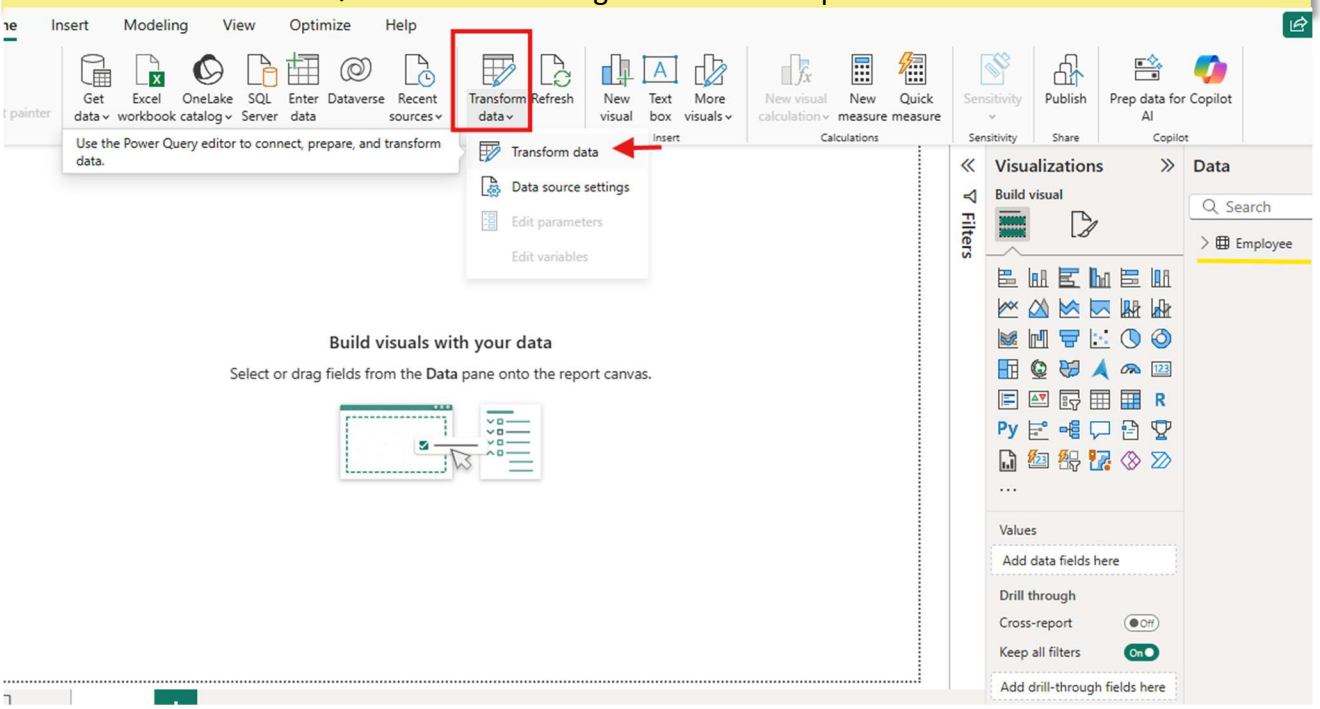
STEP 1: **OPEN POWER BI DESKTOP AND LOAD YOUR DATASET** (FOR EXAMPLE, FROM **EXCEL**, **CSV**, **SQL SERVER**, OR **WEB**).



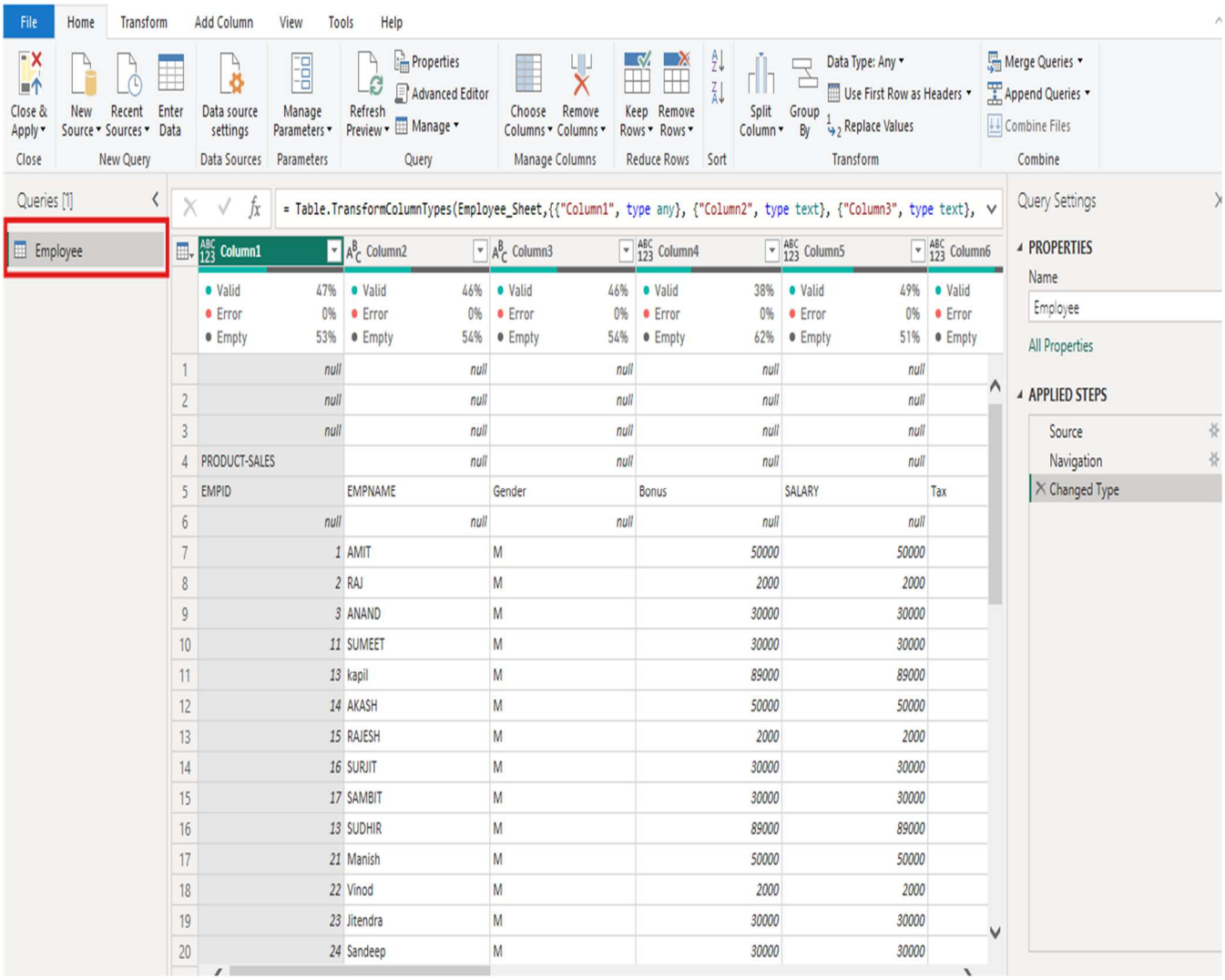
STEP 2 : **DURING THE DATA LOADING PROCESS**, YOU CAN CLICK ON **TRANSFORM DATA** (INSTEAD OF LOAD) — THIS ALLOWS YOU TO **CLEAN OR MODIFY THE DATA BEFORE LOADING IT** INTO POWER BI.



STEP 3 : Once the data is loaded, go to the **Home** tab on the ribbon.
Click on **Transform Data → Transform Data** again from the dropdown.



STEP 4 : THIS OPENS THE **POWER QUERY EDITOR** WINDOW, WHERE YOU CAN APPLY VARIOUS DATA TRANSFORMATION STEPS.



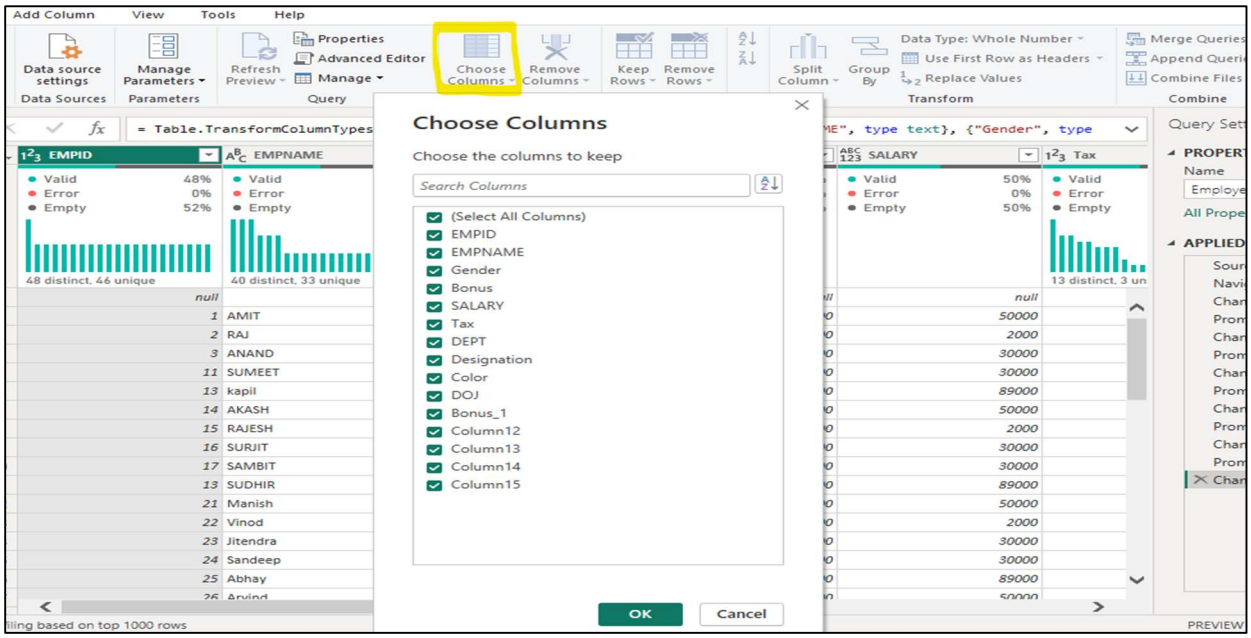
- The **Employee data** has been successfully transformed and opened **in Power Query Editor**. Now we can apply various data transformation functions **to convert raw data into clean data**, so that we can **take decisions based upon meaningful data**.

Note: Power Query Editor in Power BI is used to clean, shape, and prepare data before analysis. It allows users to modify, filter, and transform raw datasets into structured formats. Performing transformations reports.ensures data accuracy and consistency. This step is crucial for creating reliable and insightful Power BI

DATA TRANFORMATION FEATURES IN POWER BI

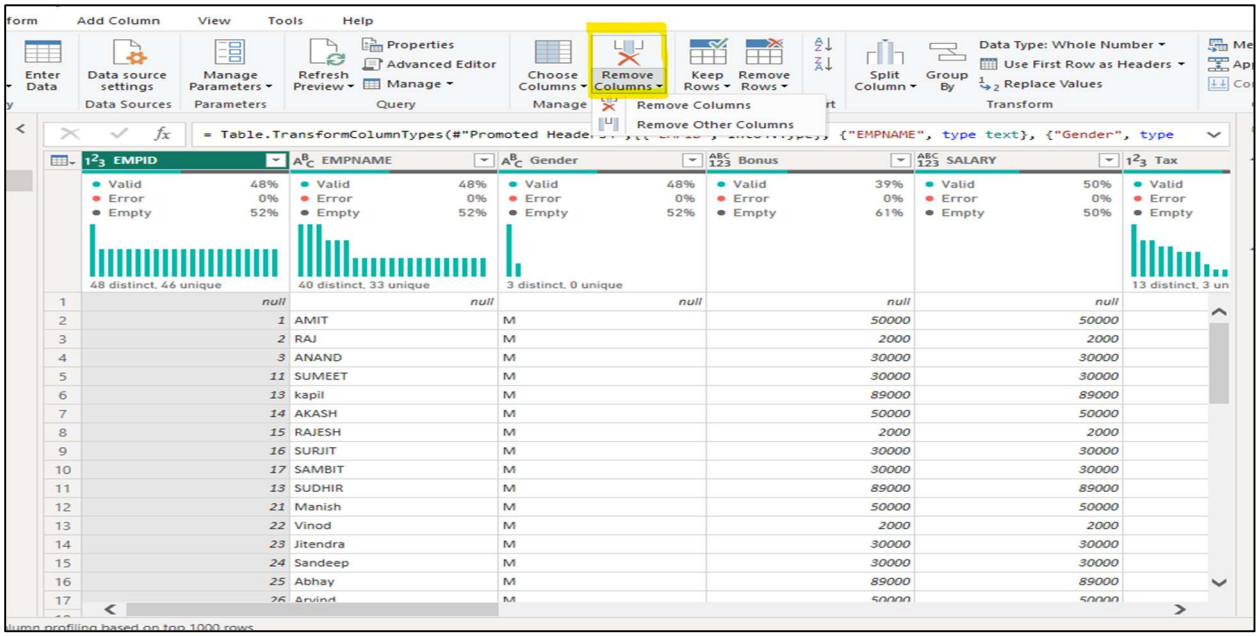
CHOOSE COLUMNS

This will allow users to choose columns from the selected datasets.



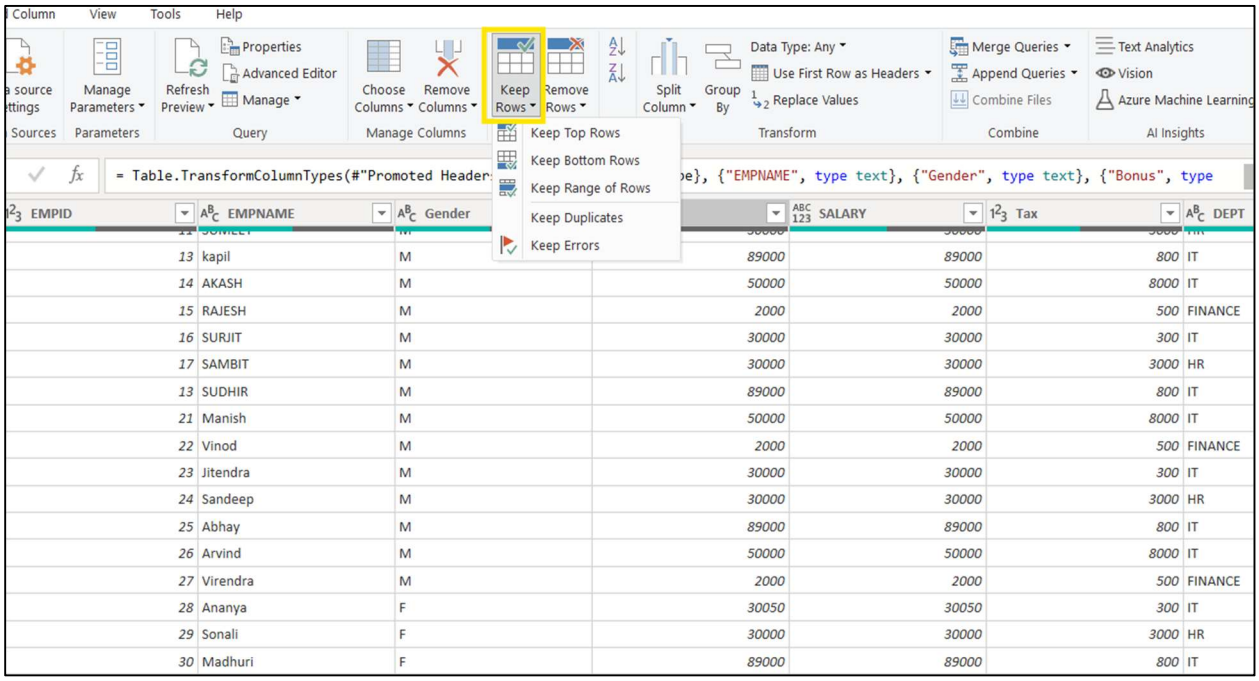
REMOVE COLUMNS

This will remove columns from the selected datasets.



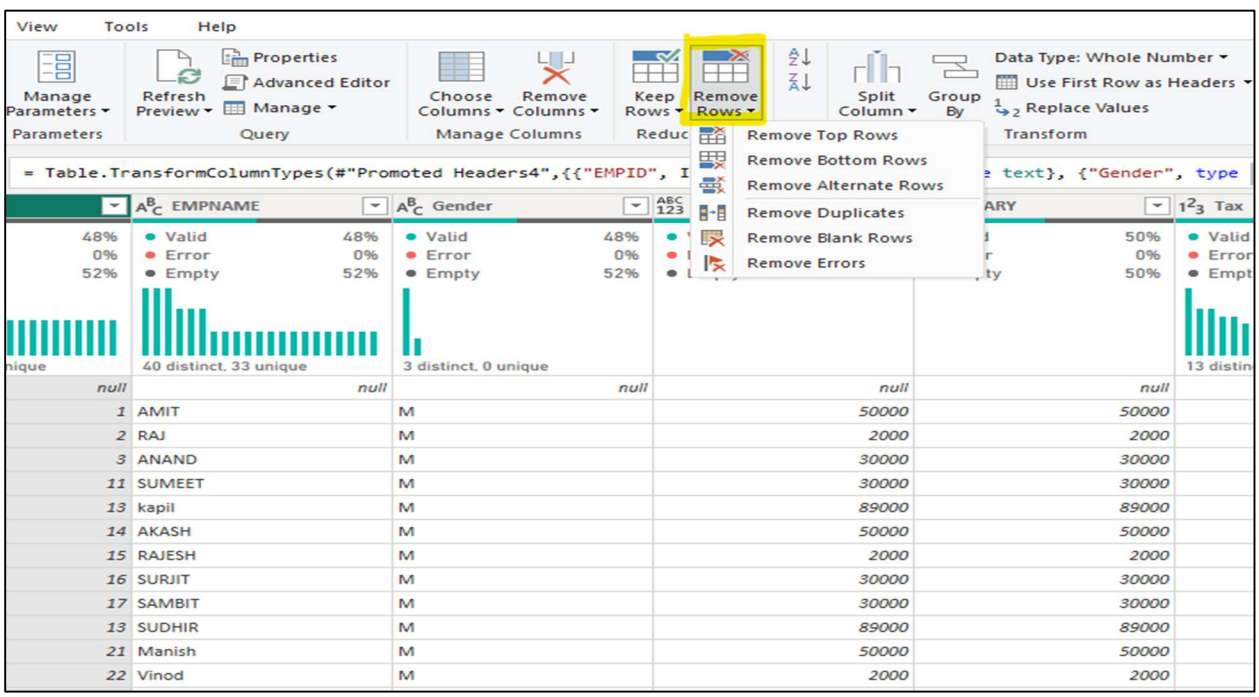
KEEP ROWS

This is keep the number of rows in the dataset as specified.



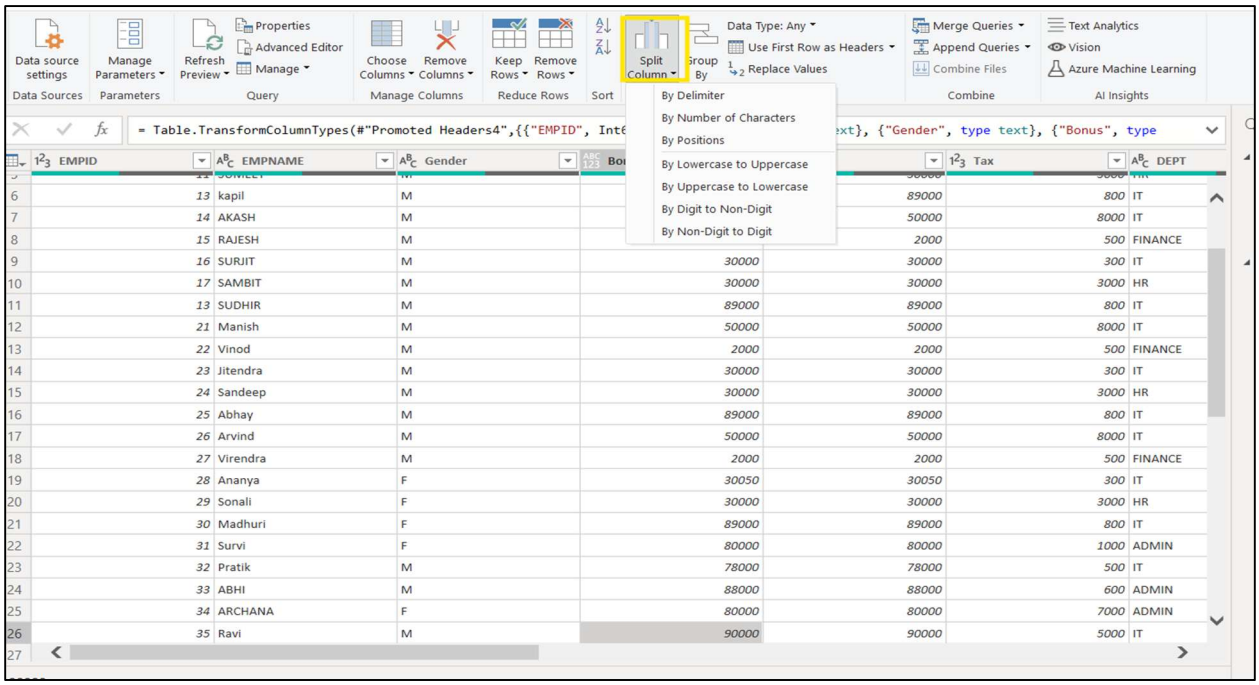
REMOVE ROWS

This will remove the rows from the dataset as specified.



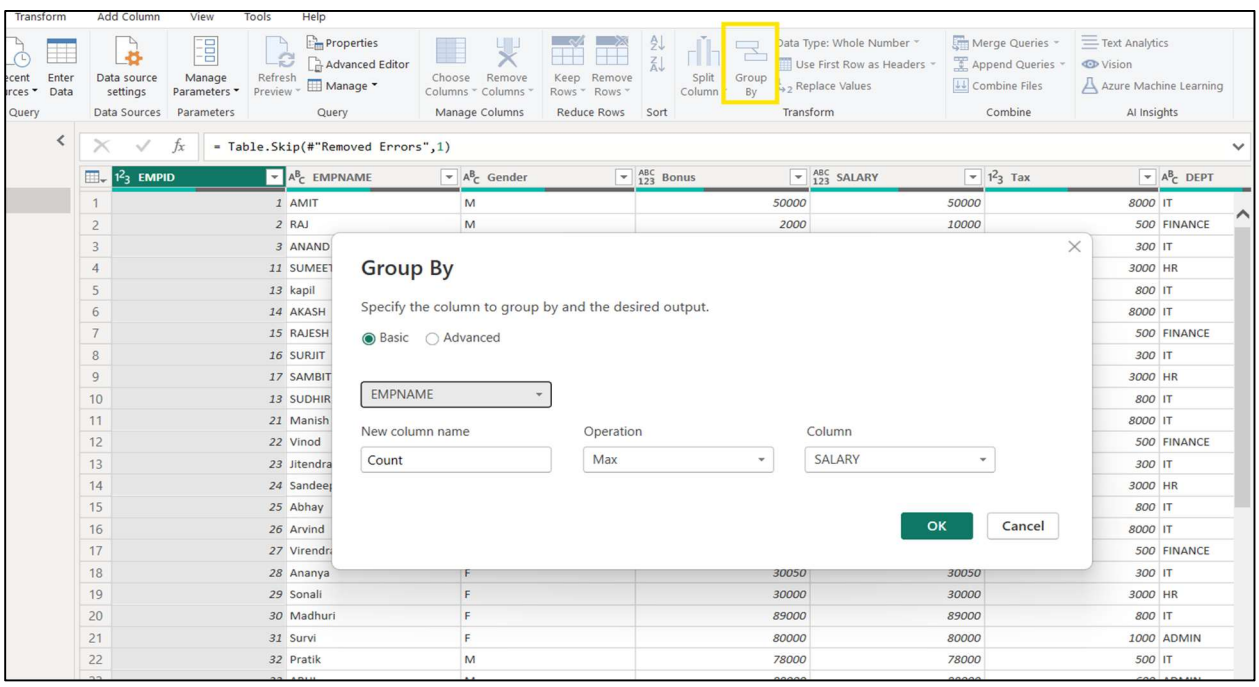
SPLIT COLUMN

We can use this to split column based upon delimiters.



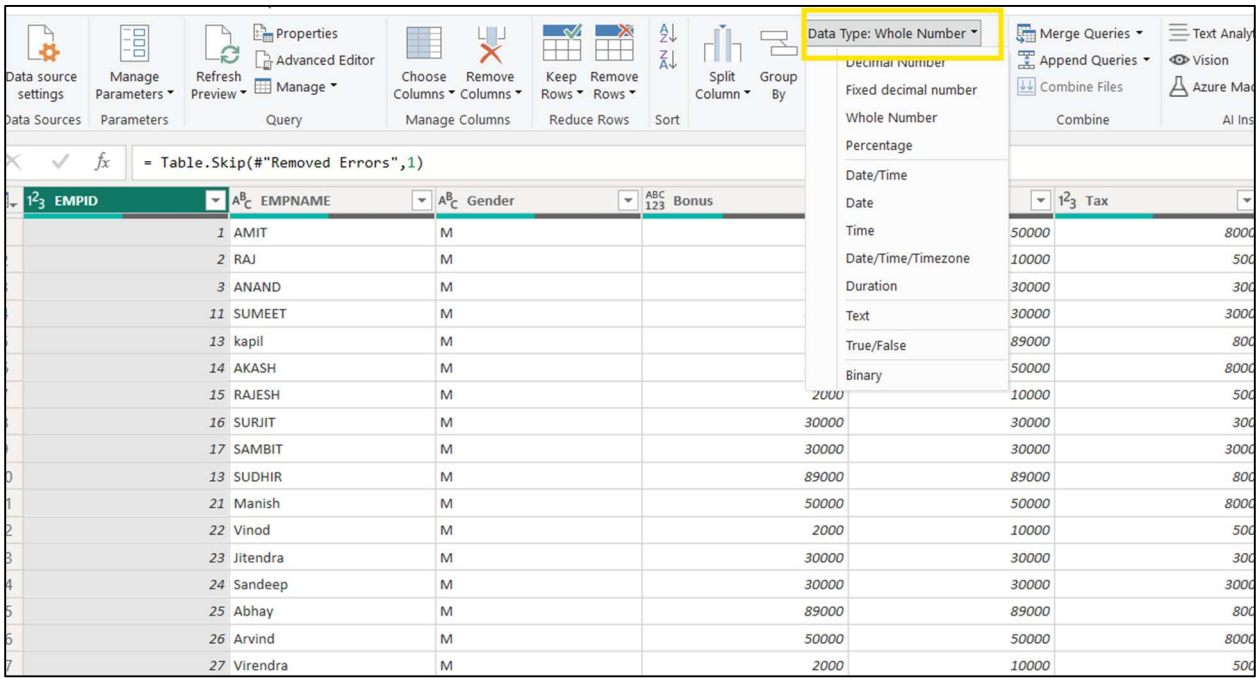
GROUP BY

This is used to add a Group By on the data.



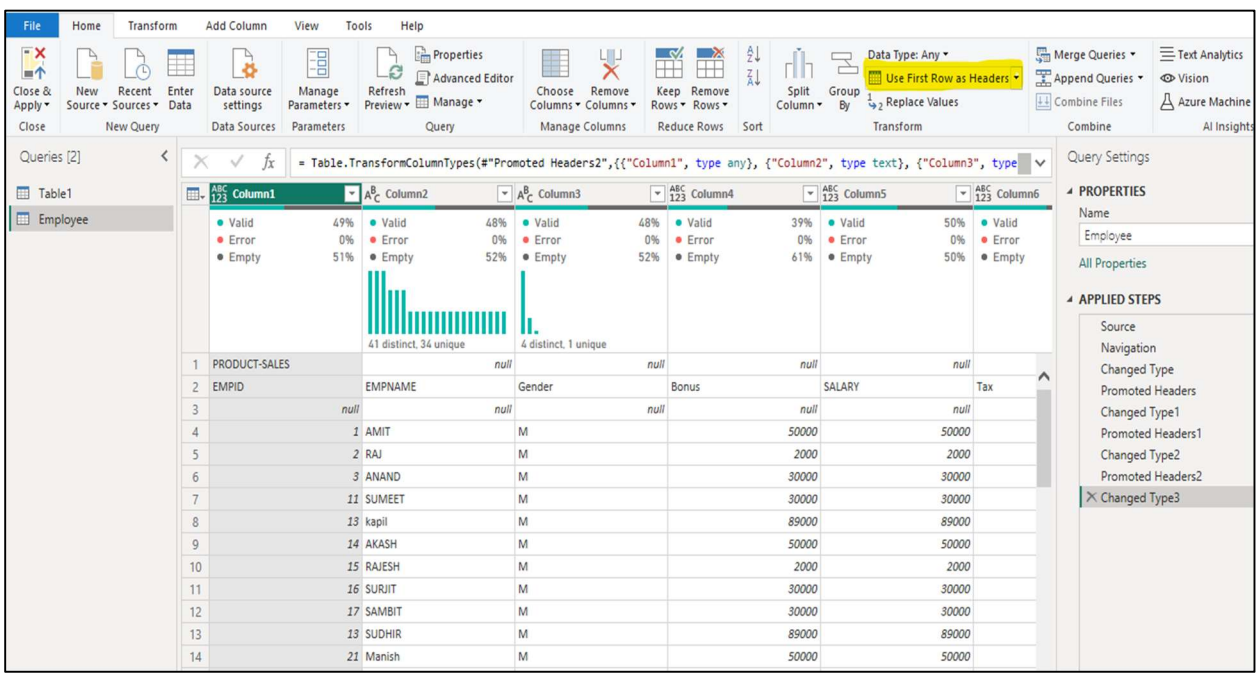
DATA TYPES

This is to store the particular data type.



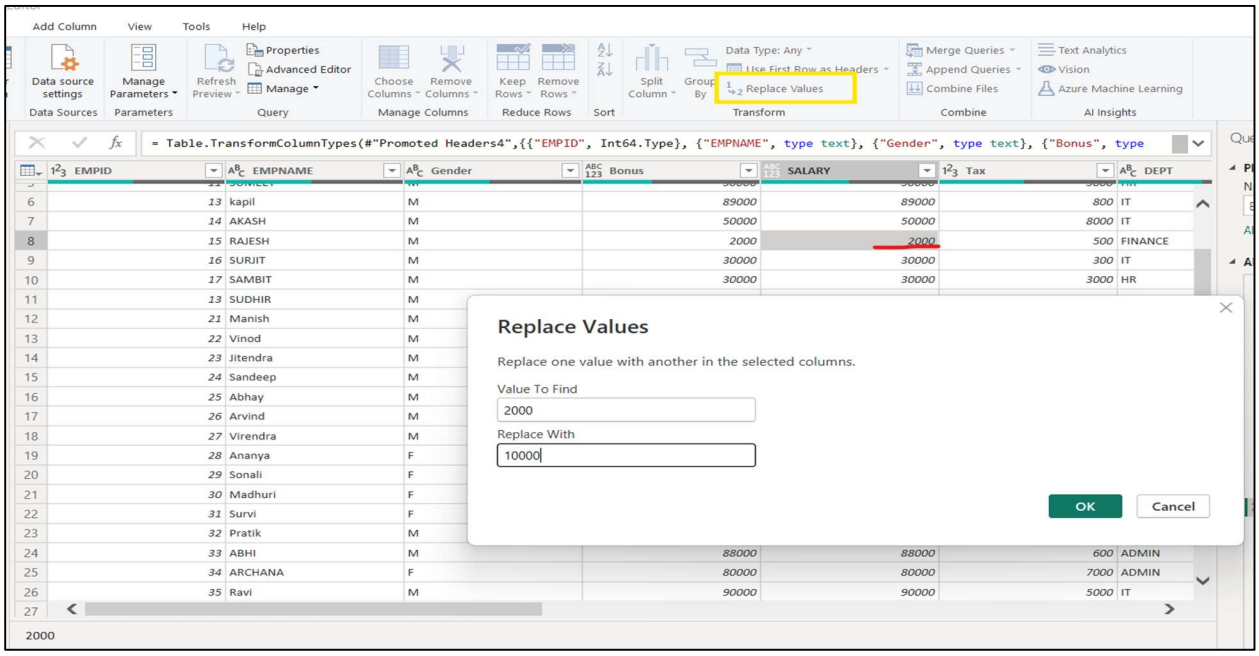
USE FIRST ROW AS HEADERS

Using this we can make the first row as header.



REPLACE VALUES

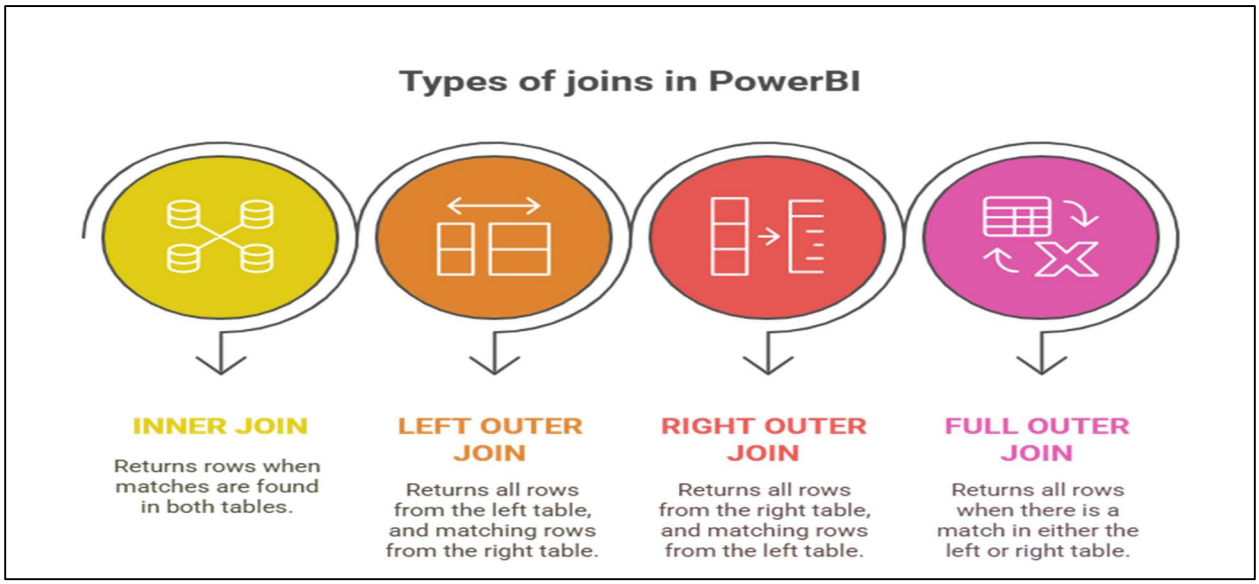
This will replace values in the datasets for the columns



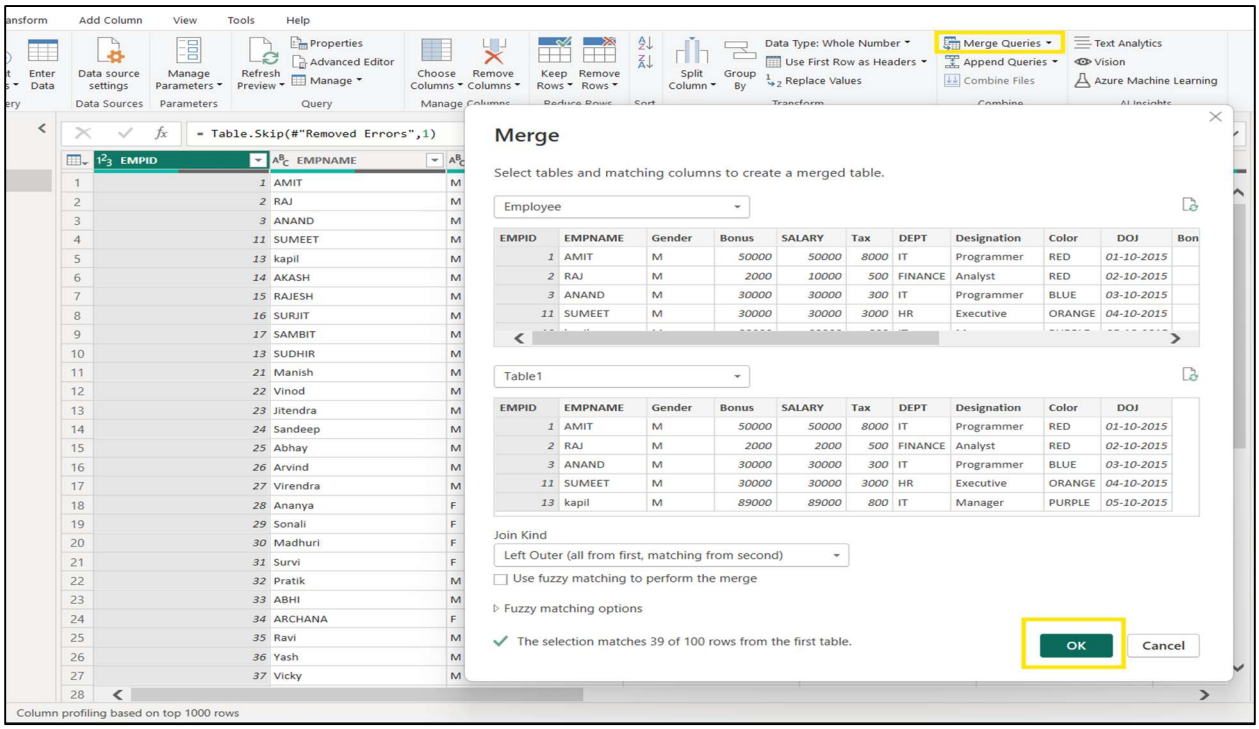
MERGE QUERIES

This will be used to join two datasets/tables in PowerBI. We can do different types of joins like

- INNER JOIN
- LEFT OUTER JOIN
- RIGHT OUTER JOIN
- FULL OUTER JOIN

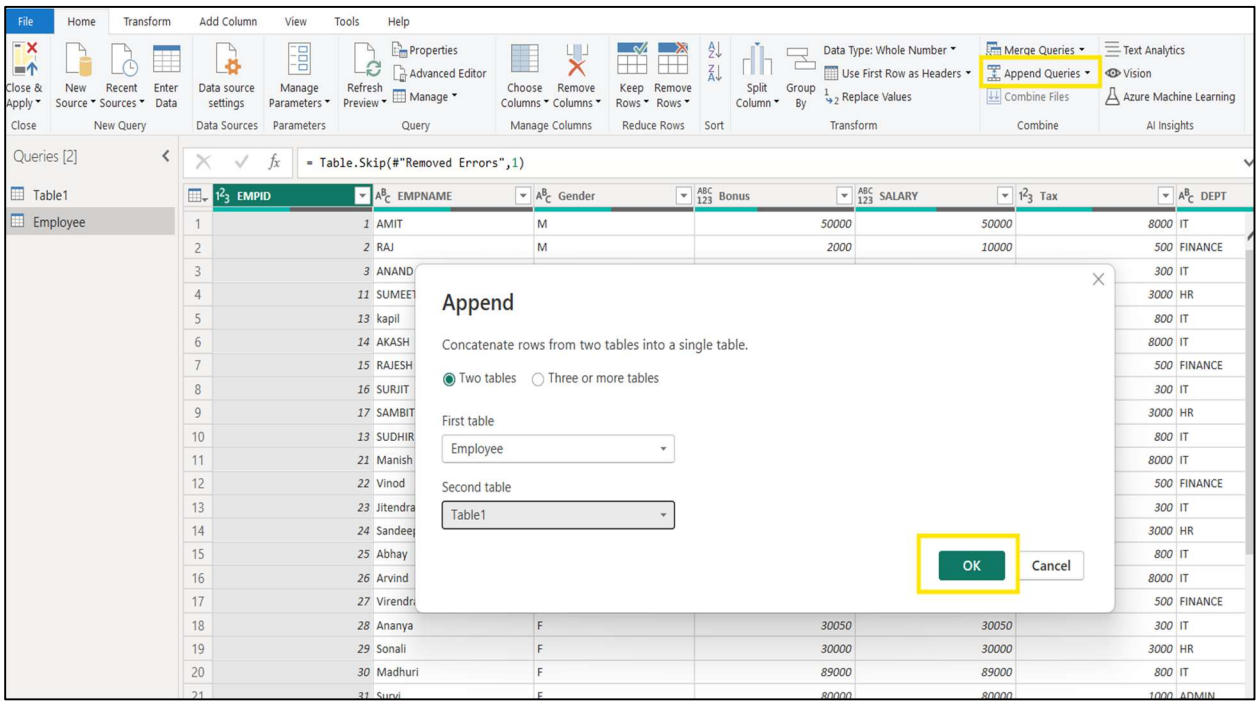


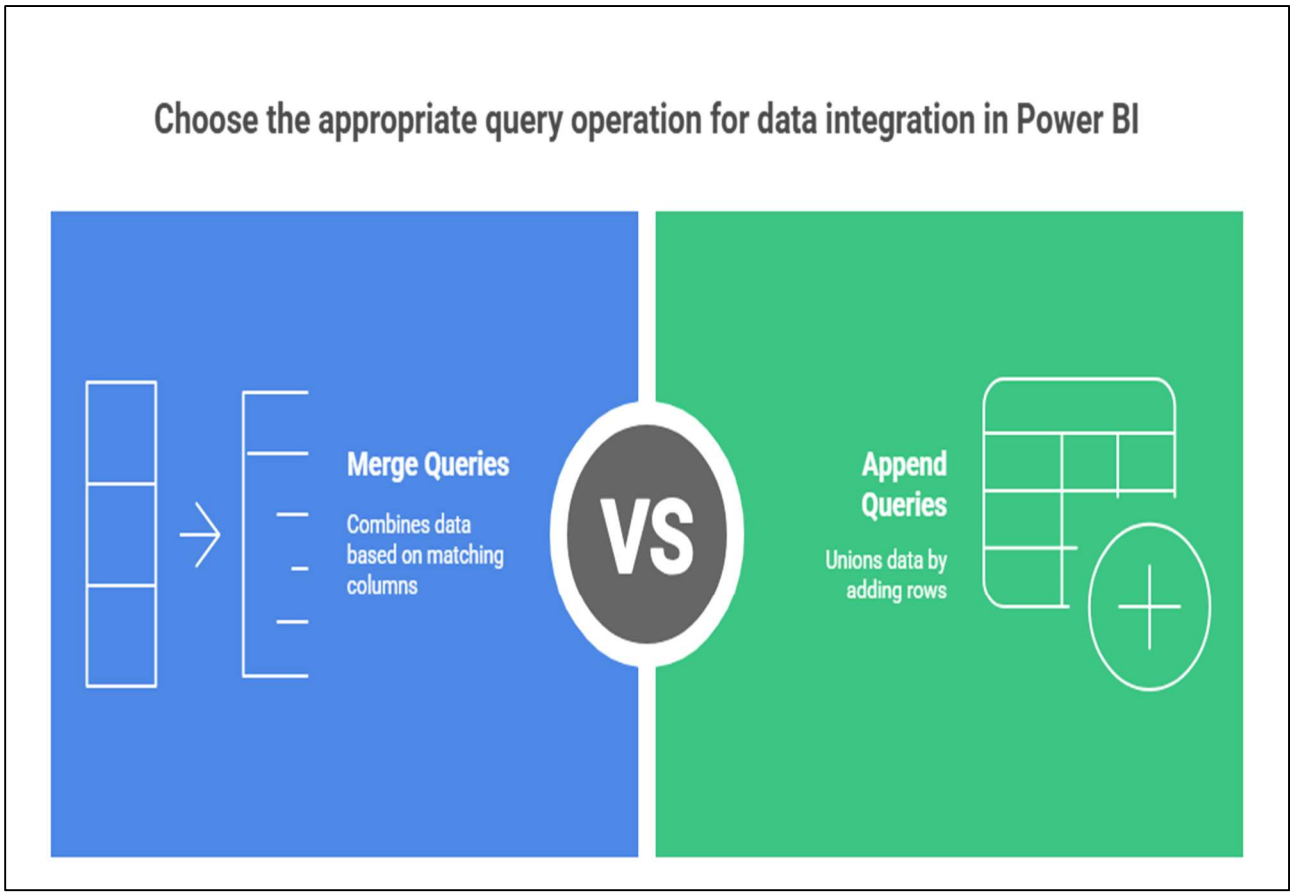
Example:



APPEND QUERIES

This is used to union two of more datasets/tables in Power BI.





Note: Data transformation features in Power BI help clean, organize, and prepare data for analysis. Tools like Split Column, Group By, and Replace Values improve data quality and structure. Merging and appending queries allow combining data from multiple sources. These steps ensure accurate and meaningful insights in reports.

QUESTIONS

1.What is Data Transformation in Power BI and why is it important?
2.Explain the steps to open Power Query Editor in Power BI.
3.What is the difference between Merge Queries and Append Queries in Power Query Editor?
4.How do operations like Split Column, Group By, and Replace Values help in data cleaning and preparation?
5.Why is changing data types important while transforming data in Power BI?

SUMMARY

Data Transformation in Power BI is the process of cleaning and shaping raw data to make it suitable for analysis, done through the **Power Query Editor**. It allows users to modify datasets before loading them into the Power BI model. Common transformation tasks include selecting or removing columns, filtering rows, splitting columns, grouping data, changing data types, replacing incorrect values, and combining data using **Merge** (joins) or **Append** (union). These operations help maintain accuracy, consistency, and structure in the dataset. By applying such transformations, Power BI ensures that the data used for visualization is **reliable, well-organized, and ready for meaningful insights**, forming a strong foundation for effective reporting and analytics.

Data transformation in Power BI is performed using Power Query Editor. It allows users to clean, shape, and prepare data before loading it into the model.

Steps to open Power Query Editor:

1. Load a dataset.
2. Click on Transform Data instead of Load.
3. Modify the data in Power Query Editor.

Common data transformation operations:

- Choose/Remove Columns – Select relevant columns.
- Keep/Remove Rows – Filter unwanted data.
- Split Column – Divide text using delimiters.
- Group By – Aggregate data based on categories.
- Change Data Types – Ensure data consistency.
- Replace Values – Substitute incorrect or missing data.
- Merge Queries – Join multiple tables (inner, left, right, full joins).
- Append Queries – Union multiple datasets.

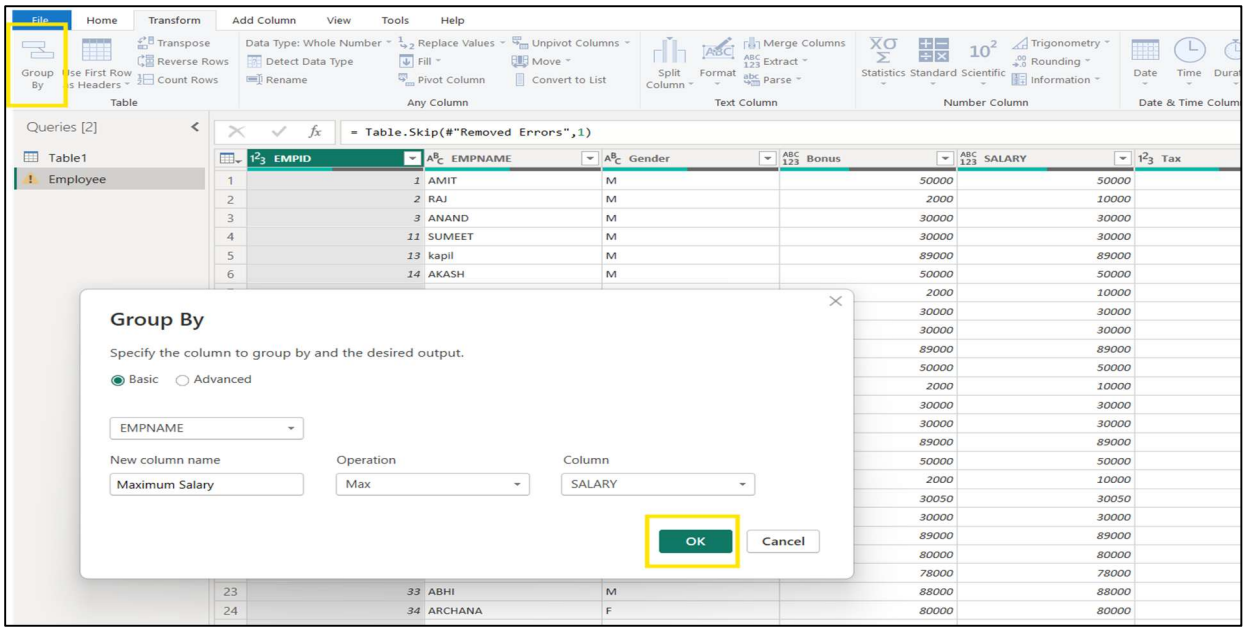
Power Query Editor ensures data consistency, reduces redundancy, and prepares clean, structured datasets for accurate visualization and analysis.

CHAPTER 6: POWER BI – ADVANCE TRANSFORMATIONS

In this chapter, we will explore advance capabilities of Power BI Data Transformation.

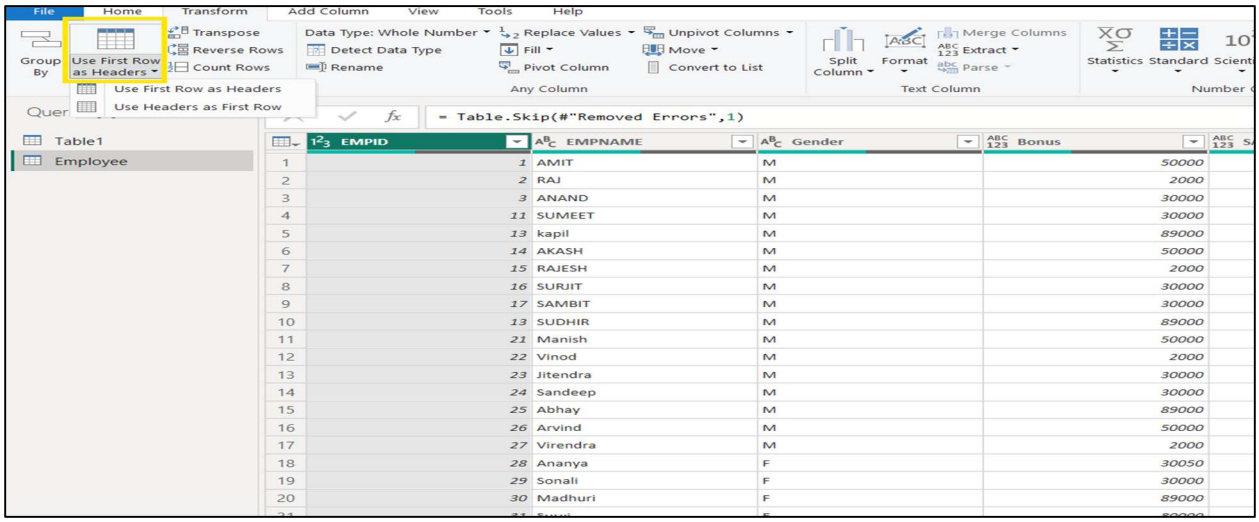
GROUP BY

This is used to apply Group By clause on the tabular data.



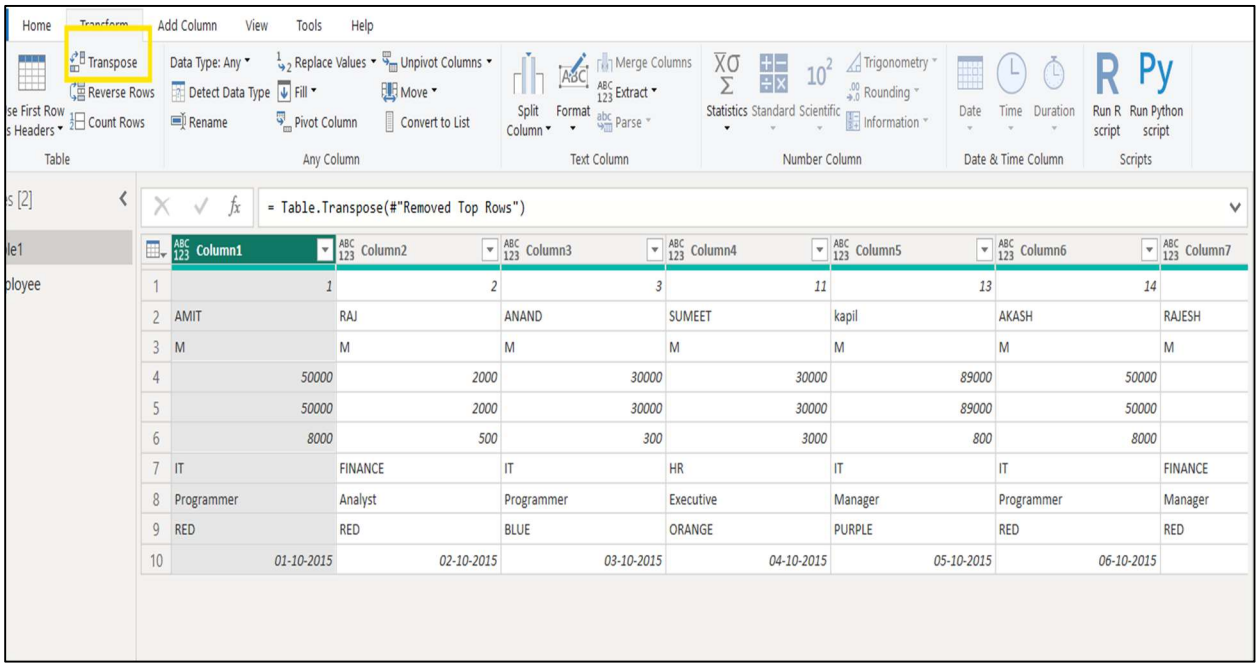
USE FIRST ROW AS HEADER

This is used to make the first row as the header row.



TRANSPOSE

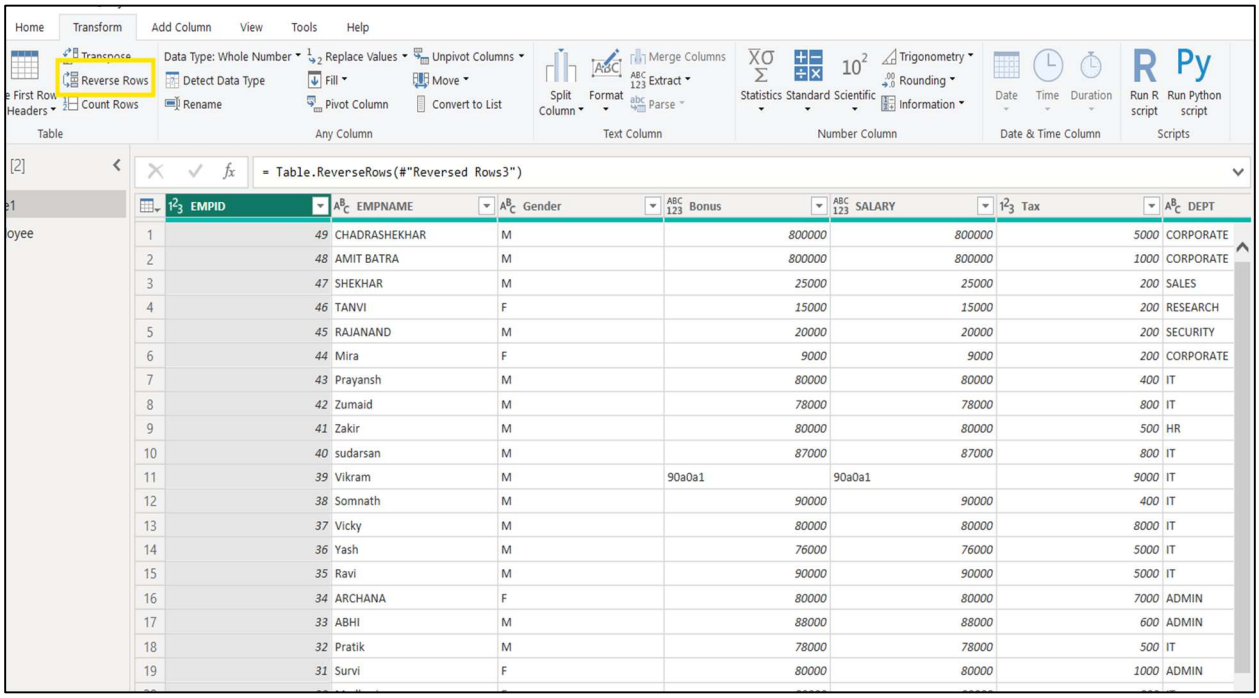
Transpose the table, treating rows as column and column as rows.



	Column1	Column2	Column3	Column4	Column5	Column6	Column7
1	1	2	3	11	13	14	
2	AMIT	RAJ	ANAND	SUMEET	kapil	AKASH	RAJESH
3	M	M	M	M	M	M	M
4	50000	2000	30000	30000	89000	50000	
5	50000	2000	30000	30000	89000	50000	
6	8000	500	300	3000	800	8000	
7	IT	FINANCE	IT	HR	IT	IT	FINANCE
8	Programmer	Analyst	Programmer	Executive	Manager	Programmer	Manager
9	RED	RED	BLUE	ORANGE	PURPLE	RED	RED
10	01-10-2015	02-10-2015	03-10-2015	04-10-2015	05-10-2015	06-10-2015	

REVERSE ROWS

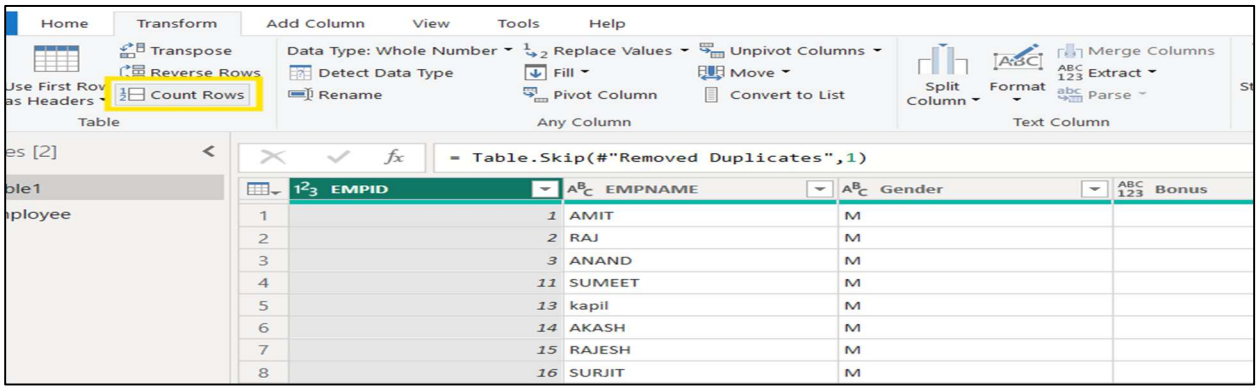
This will reverse the rows of the table. Will take the bottom rows to the top rows.

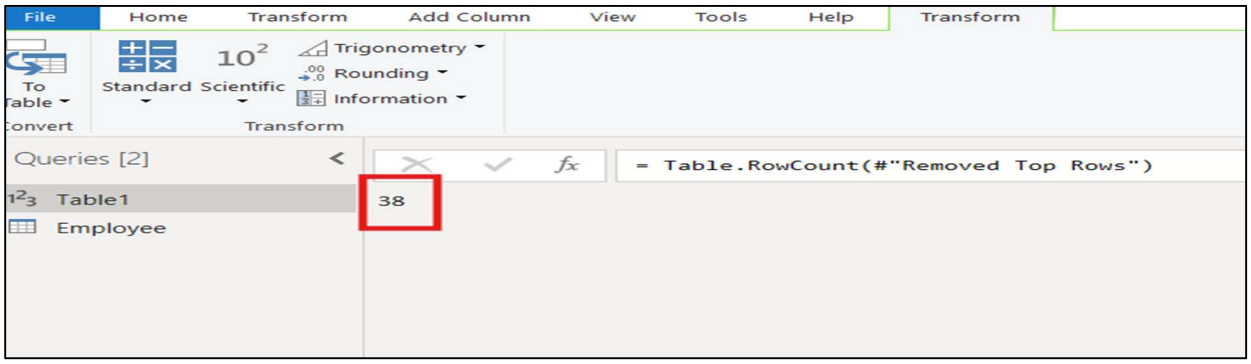


	EMPID	EMPNAME	Gender	Bonus	SALARY	Tax	DEPT
1	49	CHADRASHEKHAR	M	800000	800000	5000	CORPORATE
2	48	AMIT BATRA	M	800000	800000	1000	CORPORATE
3	47	SHEKHAR	M	25000	25000	200	SALES
4	46	TANVI	F	15000	15000	200	RESEARCH
5	45	RAJANAND	M	20000	20000	200	SECURITY
6	44	Mira	F	9000	9000	200	CORPORATE
7	43	Prayansh	M	80000	80000	400	IT
8	42	Zumaid	M	78000	78000	800	IT
9	41	Zakir	M	80000	80000	500	HR
10	40	sudarsan	M	87000	87000	800	IT
11	39	Vikram	M	90a0a1	90a0a1	9000	IT
12	38	Somnath	M	90000	90000	400	IT
13	37	Vicky	M	80000	80000	8000	IT
14	36	Yash	M	76000	76000	5000	IT
15	35	Ravi	M	90000	90000	5000	IT
16	34	ARCHANA	F	80000	80000	7000	ADMIN
17	33	ABHI	M	88000	88000	600	ADMIN
18	32	Pratik	M	78000	78000	500	IT
19	31	Survi	F	80000	80000	1000	ADMIN

COUNT ROWS

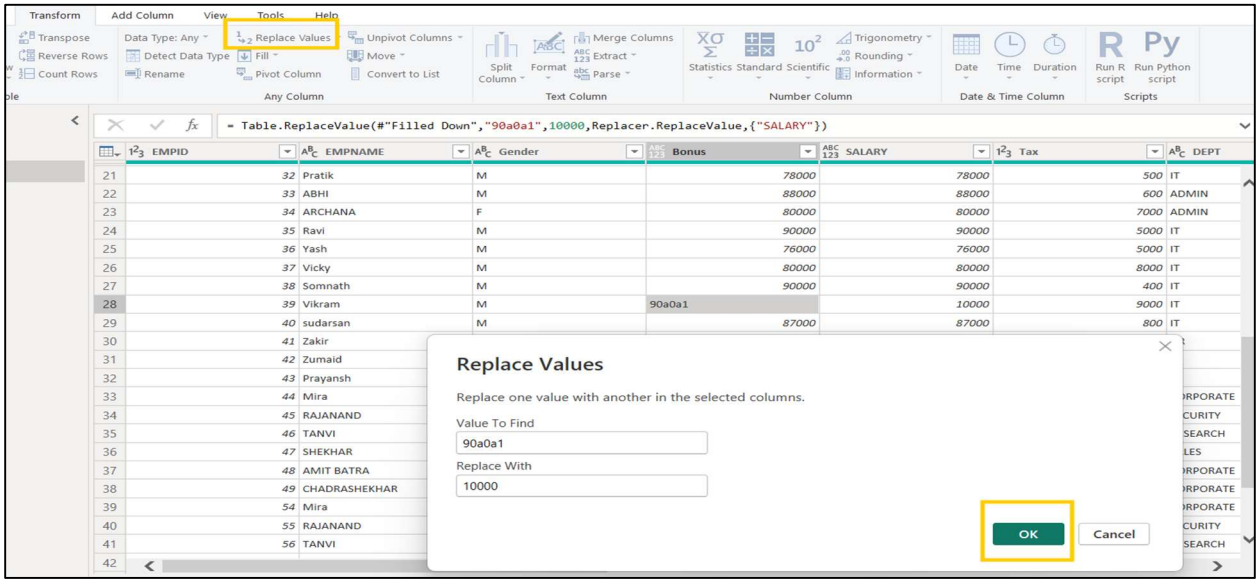
This will count the no of rows





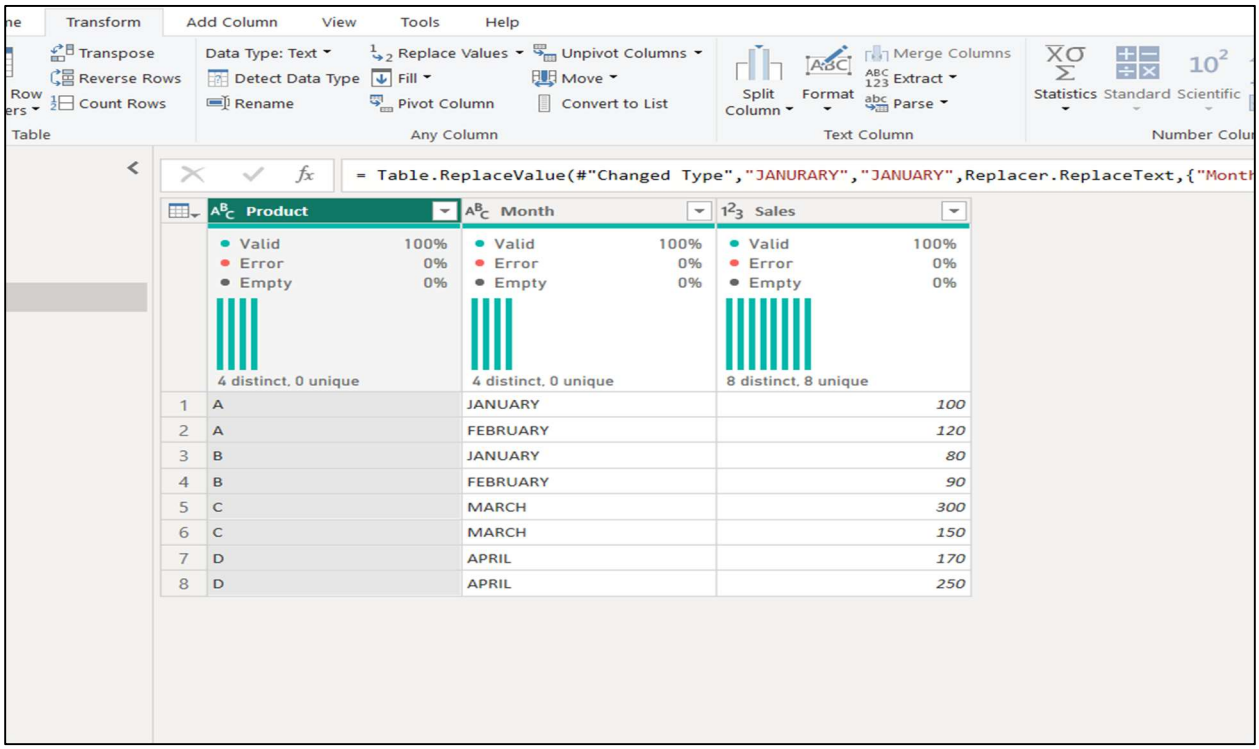
REPLACE VALUES

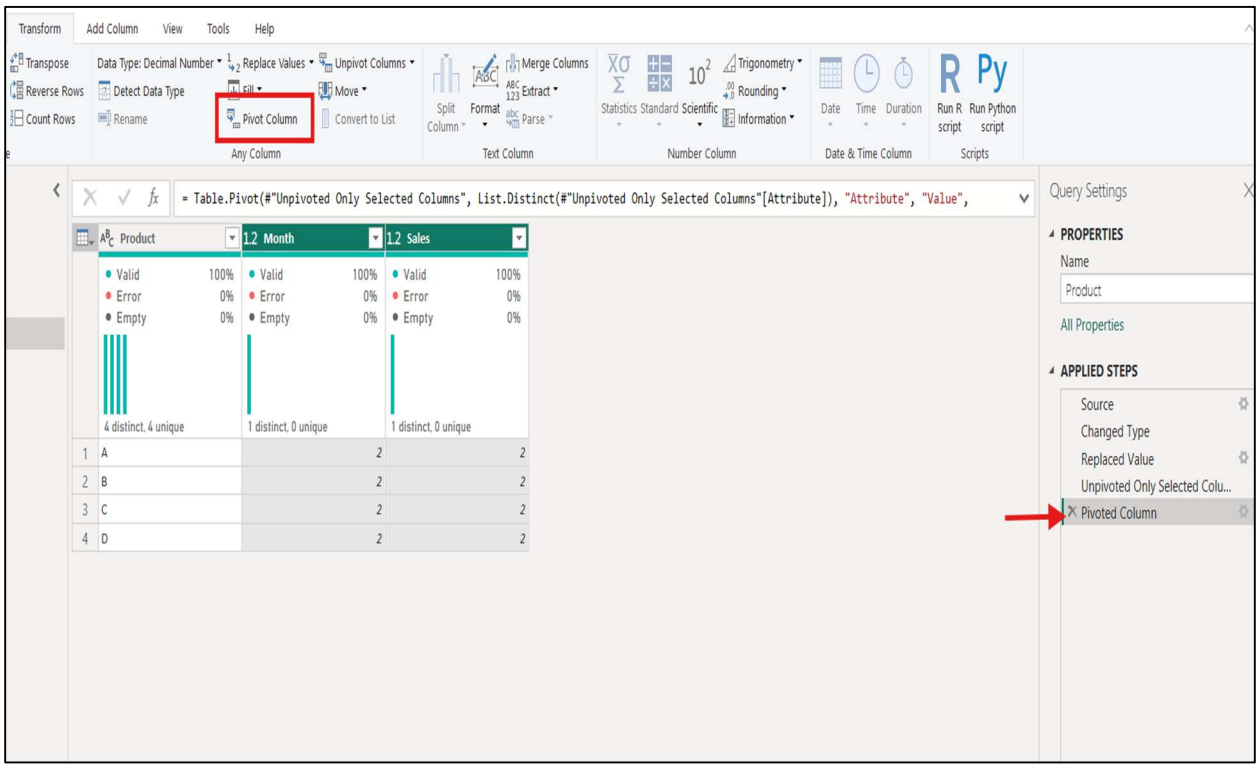
This will replace the values of the table



PIVOT COLUMN

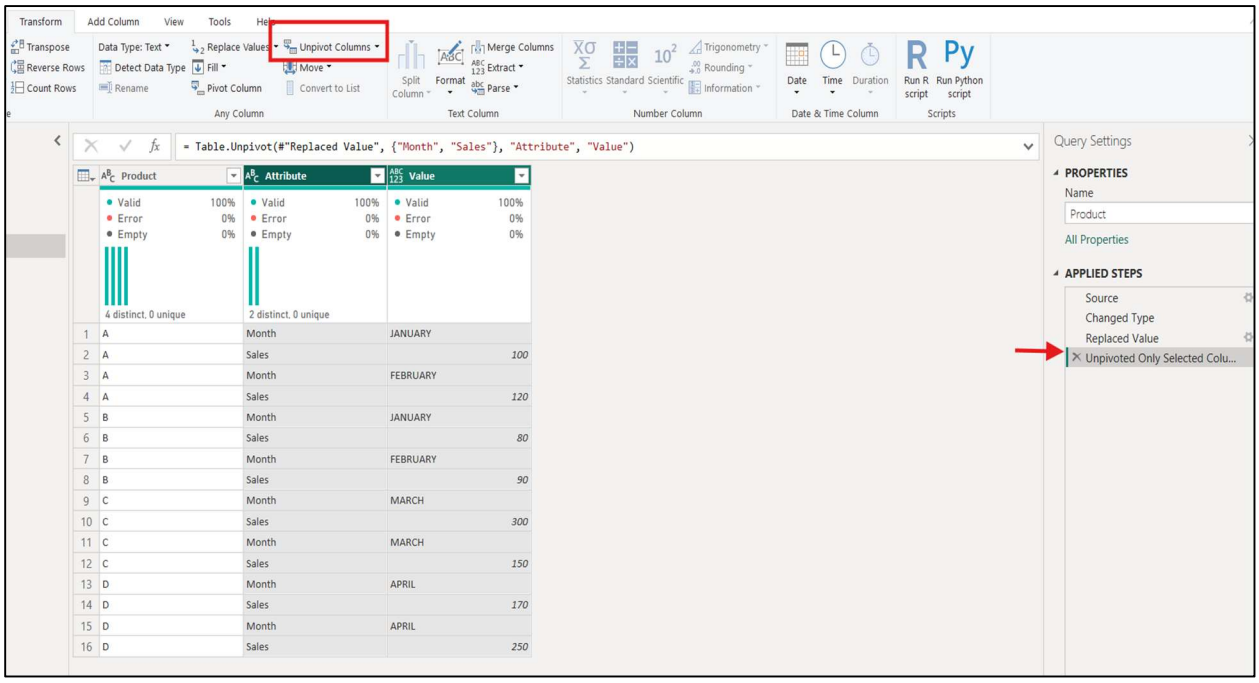
This will apply Pivot on the data of Power BI.





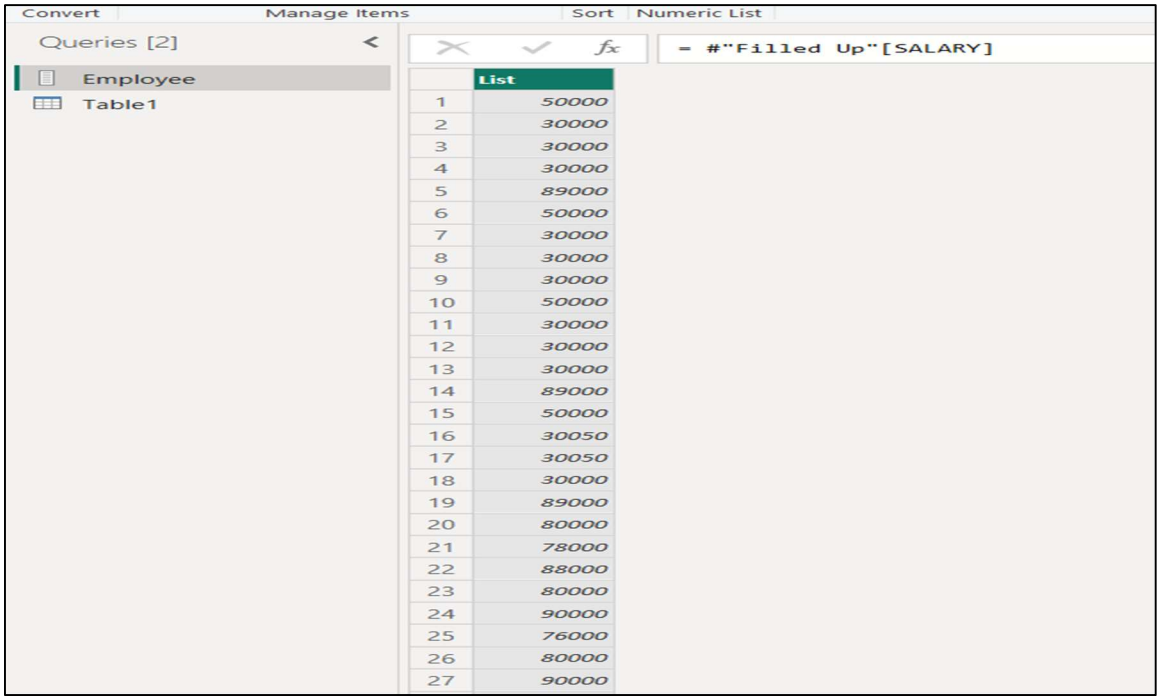
UNPIVOT COLUMNS

This will apply UnPivot on the data of Power BI.



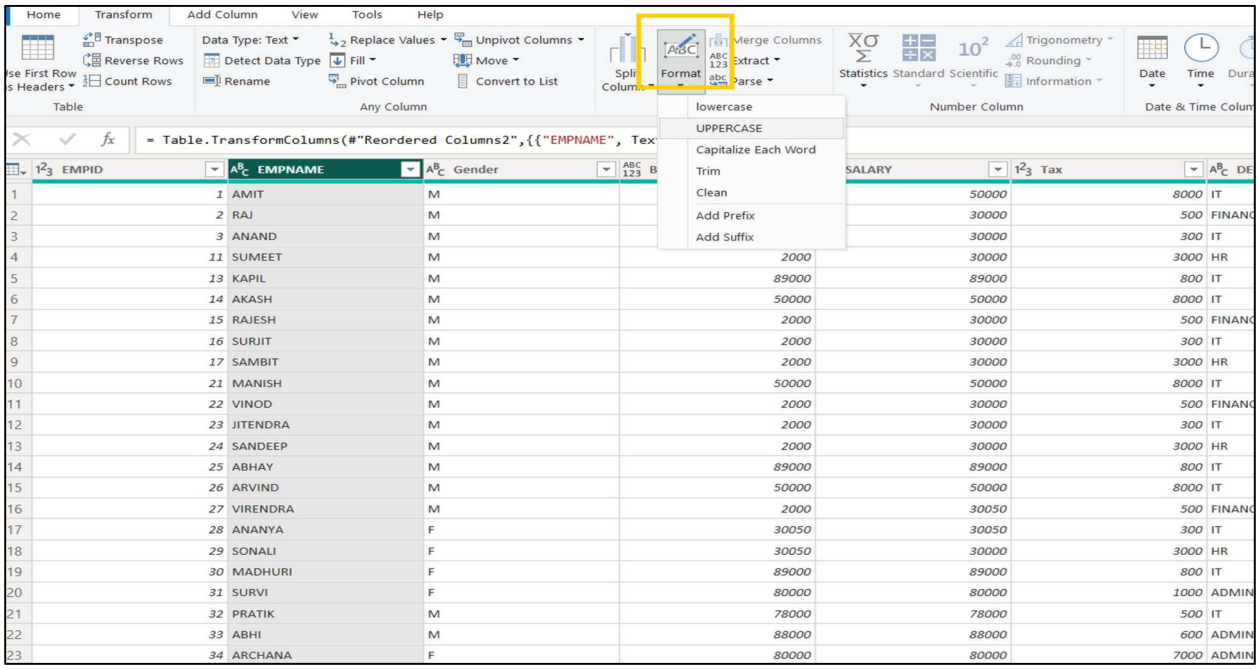
CONVERT TO LIST

This will create the DataTable column values to List from the data of PowerBI.



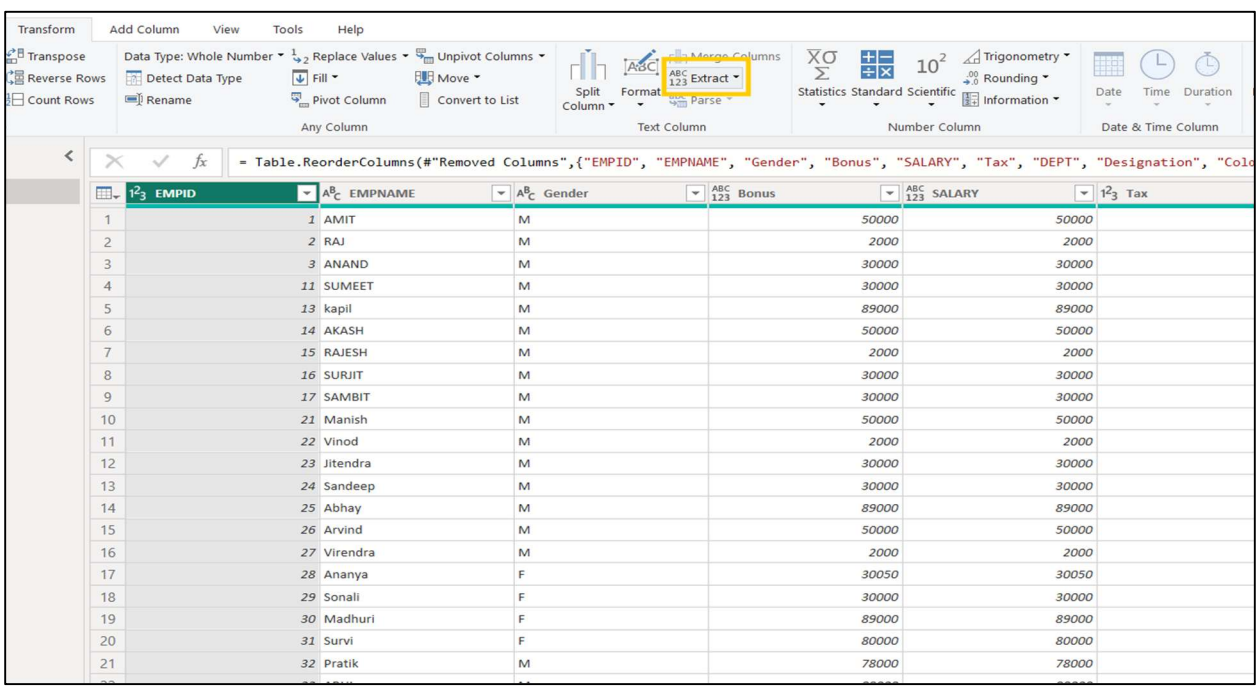
FORMAT

This will apply format function to the tabular data.



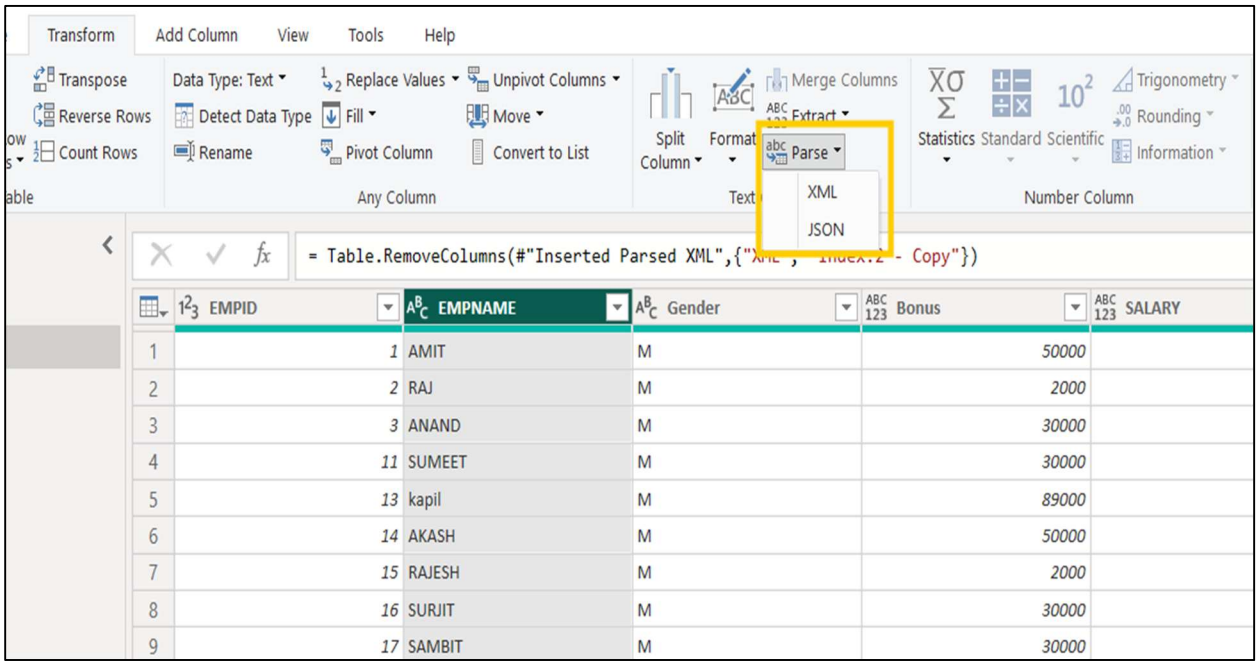
EXTRACT

Extract character from text value



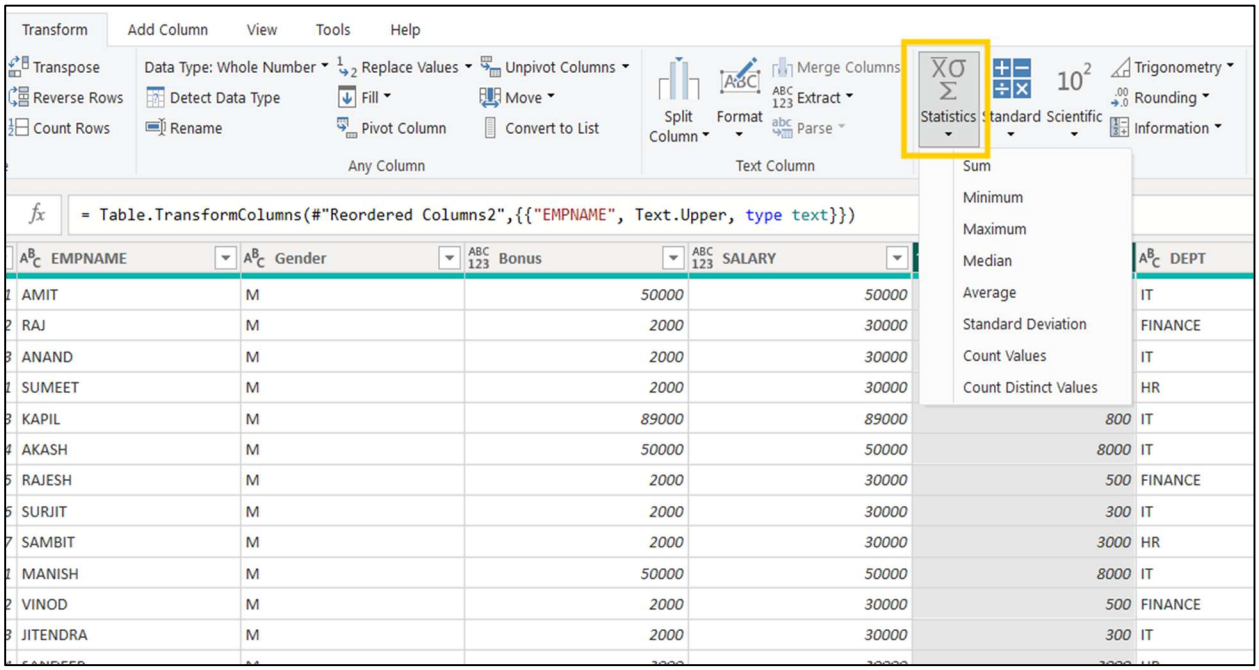
PARSE

Extract rows and column from XML or JSON format texted



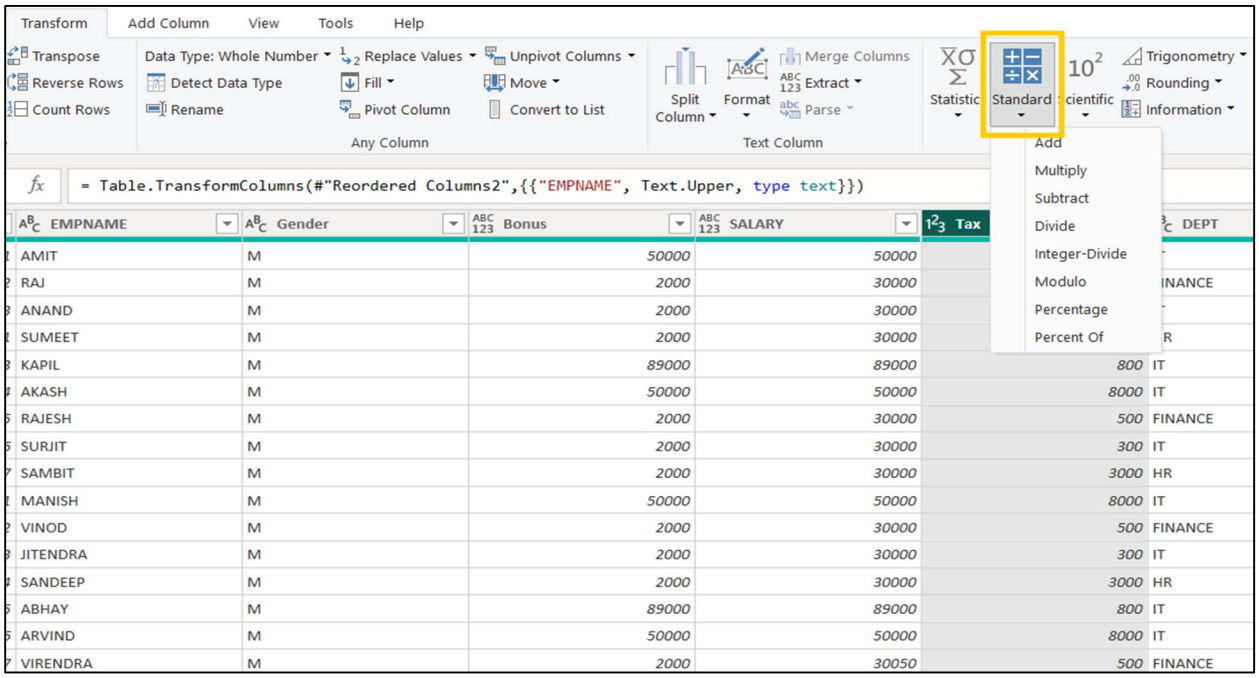
STATISTICS

Perform statistical operations.



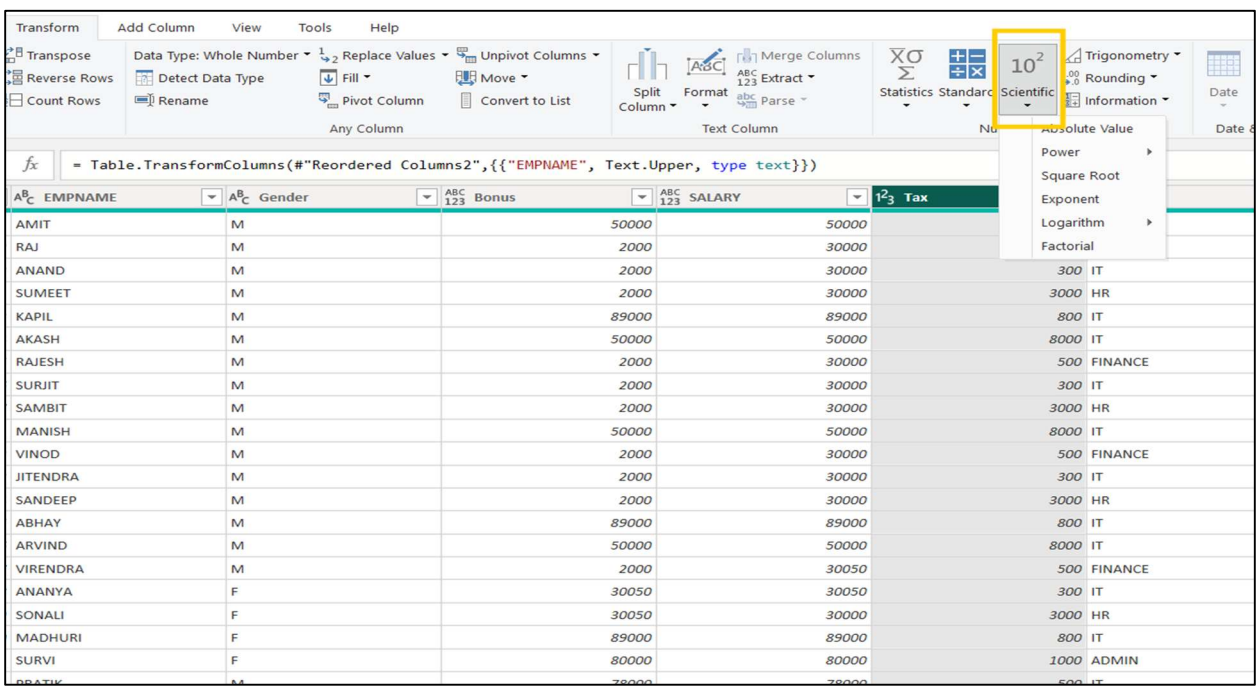
STANDARD

Performs basic math operations



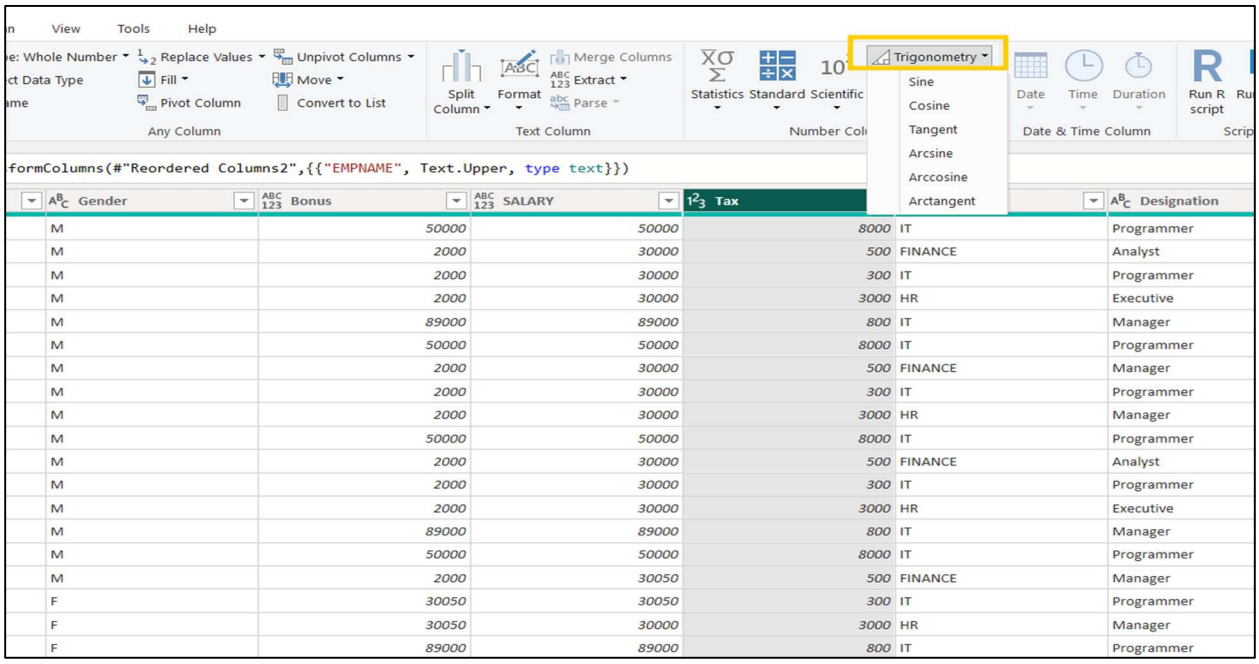
SCIENTIFIC

Performs scientific operations



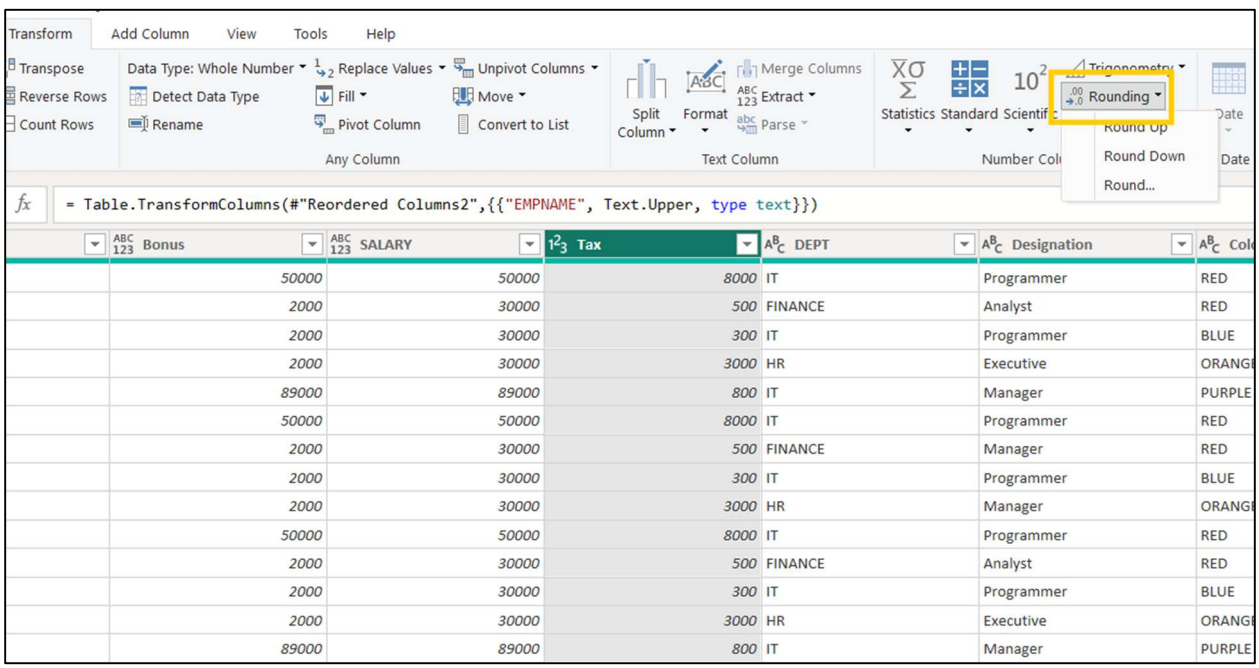
TRIGNOMETRY

Performs trigonometric operations



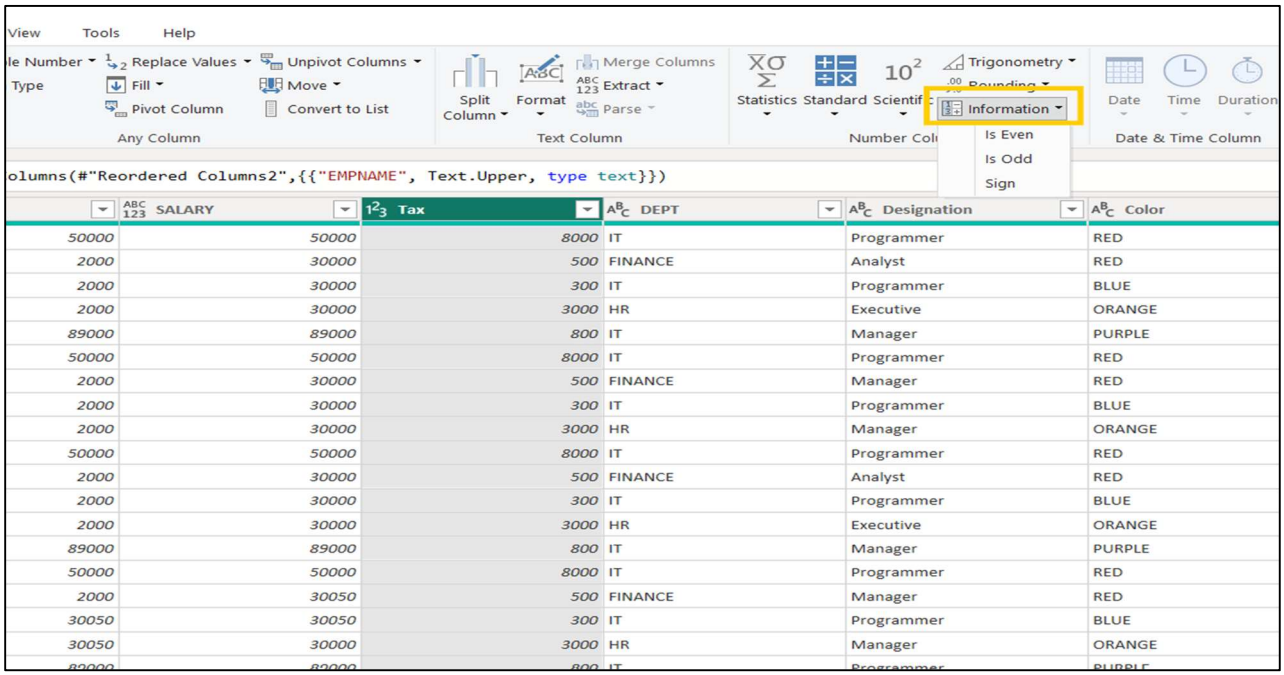
ROUNDING

Perform rounding on number values



INFORMATION

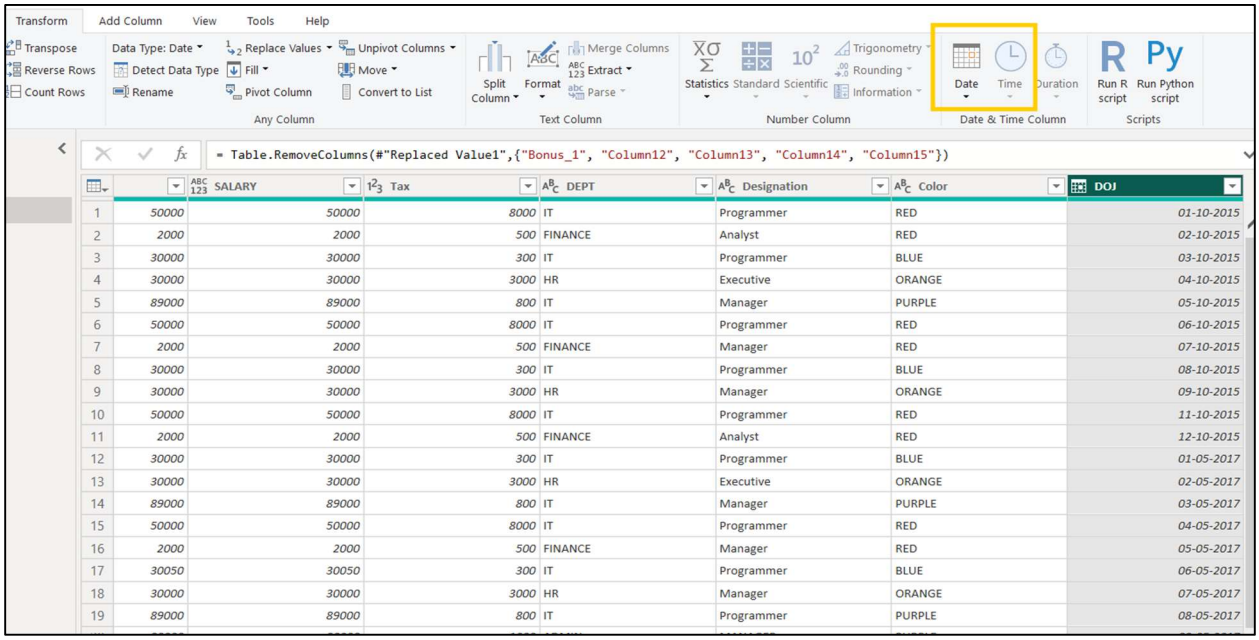
Extract information about a column



SALARY		Tax	DEPT	Designation	Color
50000	50000	8000	IT	Programmer	RED
2000	30000	500	FINANCE	Analyst	RED
2000	30000	300	IT	Programmer	BLUE
2000	30000	3000	HR	Executive	ORANGE
89000	89000	800	IT	Manager	PURPLE
50000	50000	8000	IT	Programmer	RED
2000	30000	500	FINANCE	Manager	RED
2000	30000	300	IT	Programmer	BLUE
2000	30000	3000	HR	Manager	ORANGE
50000	50000	8000	IT	Programmer	RED
2000	30000	500	FINANCE	Analyst	RED
2000	30000	300	IT	Programmer	BLUE
2000	30000	3000	HR	Executive	ORANGE
89000	89000	800	IT	Manager	PURPLE
50000	50000	8000	IT	Programmer	RED
2000	30050	500	FINANCE	Manager	RED
30050	30050	300	IT	Programmer	BLUE
30050	30000	3000	HR	Manager	ORANGE

DATE & TIME COLUMN

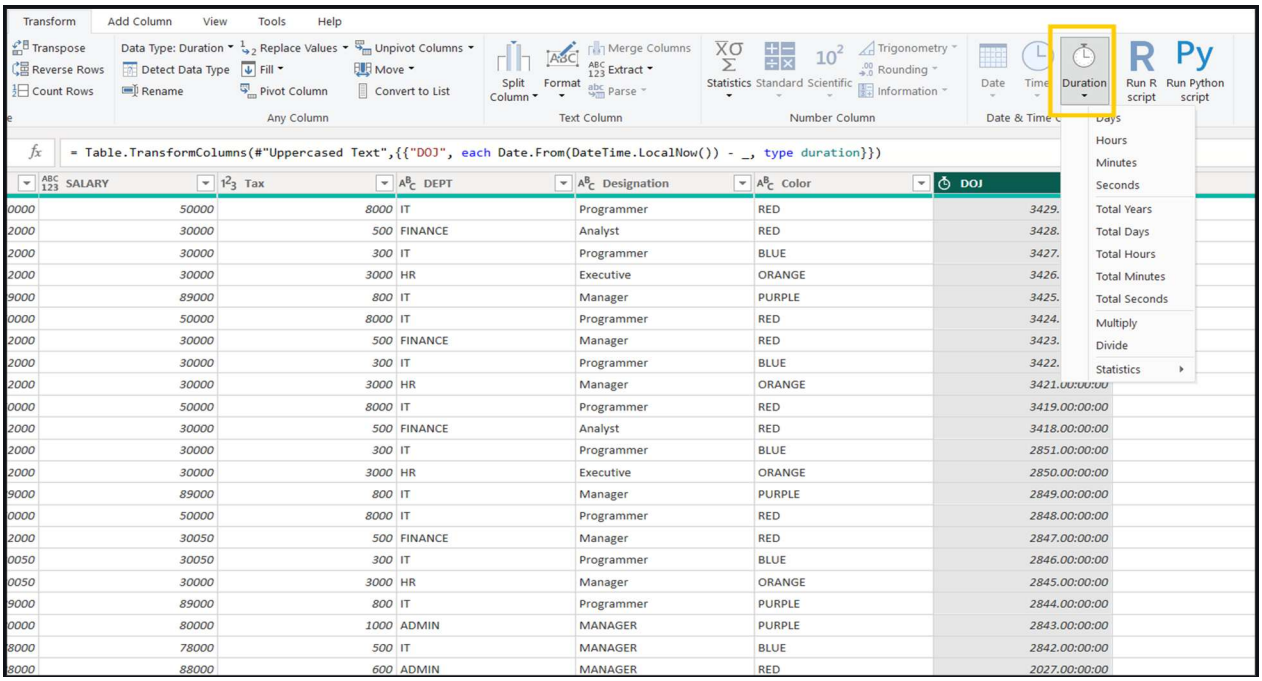
- Date Column-Format data value or extract elements of date values
- Date Column- Format time value or extract elements of time values



	SALARY		Tax	DEPT	Designation	Color	DOJ
1	50000	50000	8000	IT	Programmer	RED	01-10-2015
2	2000	2000	500	FINANCE	Analyst	RED	02-10-2015
3	30000	30000	300	IT	Programmer	BLUE	03-10-2015
4	30000	30000	3000	HR	Executive	ORANGE	04-10-2015
5	89000	89000	800	IT	Manager	PURPLE	05-10-2015
6	50000	50000	8000	IT	Programmer	RED	06-10-2015
7	2000	2000	500	FINANCE	Manager	RED	07-10-2015
8	30000	30000	300	IT	Programmer	BLUE	08-10-2015
9	30000	30000	3000	HR	Manager	ORANGE	09-10-2015
10	50000	50000	8000	IT	Programmer	RED	11-10-2015
11	2000	2000	500	FINANCE	Analyst	RED	12-10-2015
12	30000	30000	300	IT	Programmer	BLUE	01-05-2017
13	30000	30000	3000	HR	Executive	ORANGE	02-05-2017
14	89000	89000	800	IT	Manager	PURPLE	03-05-2017
15	50000	50000	8000	IT	Programmer	RED	04-05-2017
16	2000	2000	500	FINANCE	Manager	RED	05-05-2017
17	30050	30050	300	IT	Programmer	BLUE	06-05-2017
18	30000	30000	3000	HR	Manager	ORANGE	07-05-2017
19	89000	89000	800	IT	Programmer	PURPLE	08-05-2017

DURATION

Duration represents a length of time, and converts into a decimal number type when loaded into the model.



Note: Advanced transformation features in Power BI allow reshaping, summarizing, and refining data efficiently. Tools like Pivot, Unpivot, Transpose, and Group By enhance data structuring. Functions for formatting, parsing, and calculations improve accuracy and flexibility. These options help prepare complex datasets for powerful analysis and reporting.

QUESTIONS

1.What are Advanced Transformations in Power BI and why are they useful?
2.Explain the purpose of Group By, Transpose, and Reverse Rows in Power Query Editor.
3.What is the difference between Pivot Columns and Unpivot Columns in Power Query Editor?
4.How does the Convert to List feature help analysts perform advanced data operations in Power Query Editor?
5.Why are Date & Time functions and mathematical operations important in advanced data transformation in Power BI?

SUMMARY

Advanced transformations in Power BI enhance data preparation by enabling complex shaping and calculations. They allow analysts to summarize data using Group By, switch rows and columns with Transpose, reorder data with Reverse Rows, and reshape datasets using Pivot and Unpivot. Columns can also be converted into lists for further operations. Functional transformations include formatting data, extracting specific patterns, and parsing structured formats like XML or JSON. Analysts can apply mathematical, statistical, scientific, and trigonometric operations, as well as date and time functions to extract parts or calculate durations. These transformations improve data quality, reduce manual work, support advanced modeling, and provide flexible structures for deeper business insights.

Advanced transformations enhance Power BI's data-shaping capabilities, enabling complex operations and mathematical manipulations.

Key transformation techniques include:

- Group By – Summarize data using aggregation functions.
- Transpose – Convert rows into columns and vice versa.
- Reverse Rows – Reorder data for analysis.
- Pivot/Unpivot Columns – Reshape datasets for easier reporting.
- Convert to List – Convert a column into a list structure.

Mathematical and functional transformations:

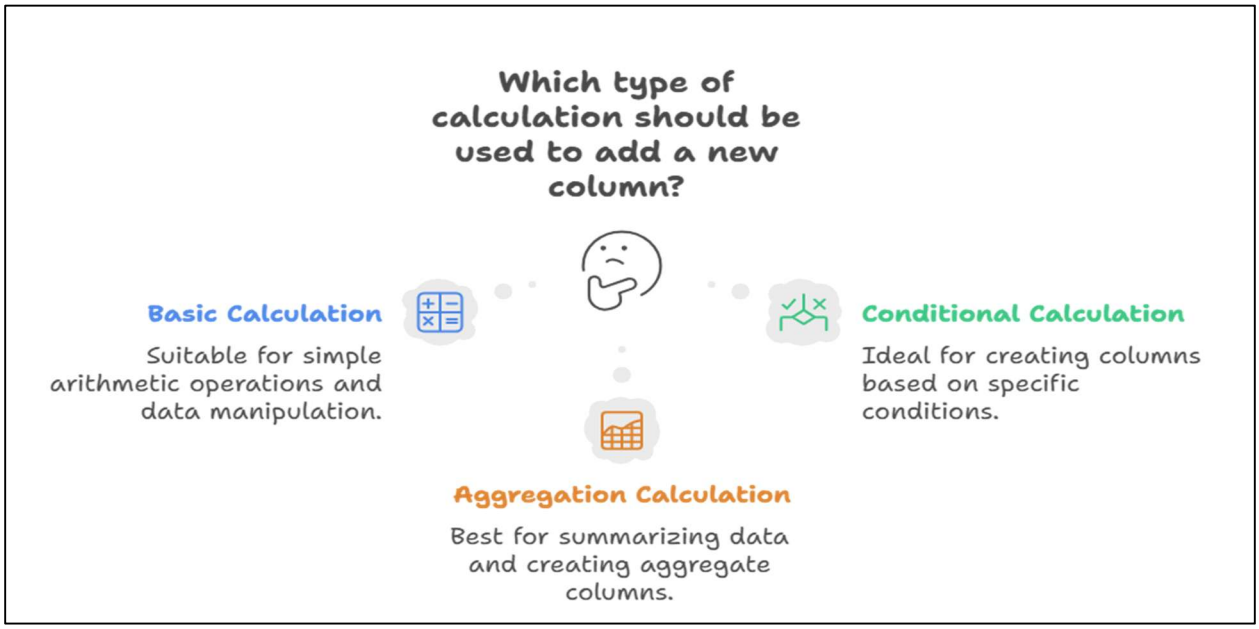
- Format – Change data representation (e.g., text to numbers).
- Extract – Pull specific characters or patterns.
- Parse – Interpret XML/JSON formatted data.
- Apply Statistical, Scientific, and Trigonometric operations.
- Date & Time Functions – Extract date parts and durations.

These transformations help analysts prepare data for advanced modeling and ensure flexible analytical structures for deeper business insights.

CHAPTER 7: CALCULATED COLUMNS IN DATA TRANSFORMATION

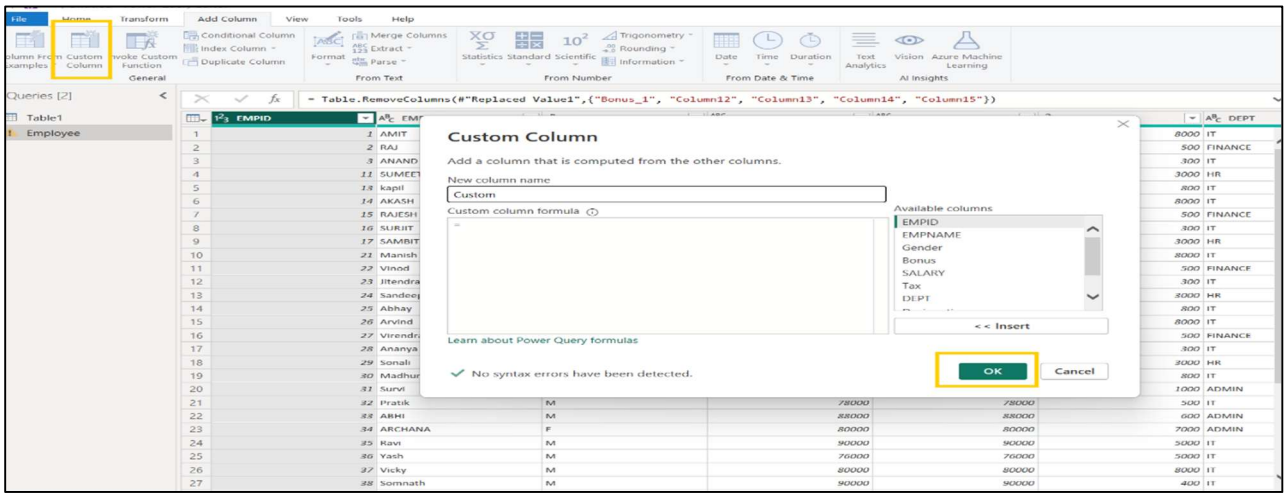
In this Chapter, will explore different types of Calculations in Power BI. We will understand how to add different types of columns in Data Transformation of Power BI.

Power BI Supports different types of calculations for create new calculated columns in the existing data.



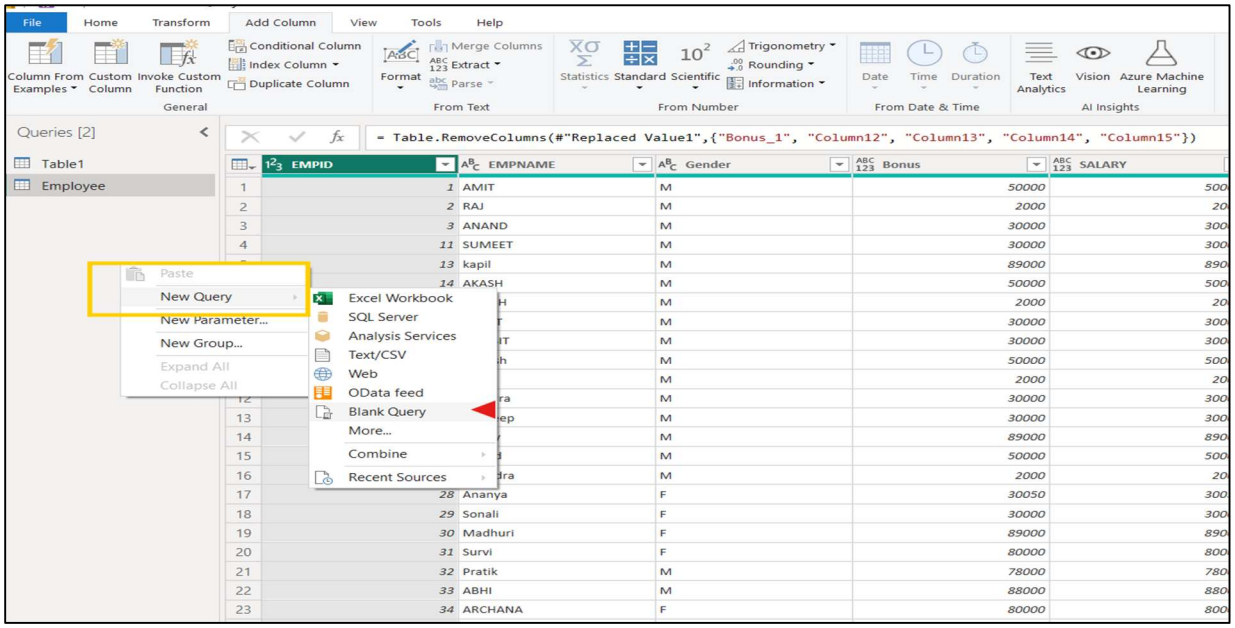
CUSTOM COLUMN

Create new column on this table based on the custom formula.

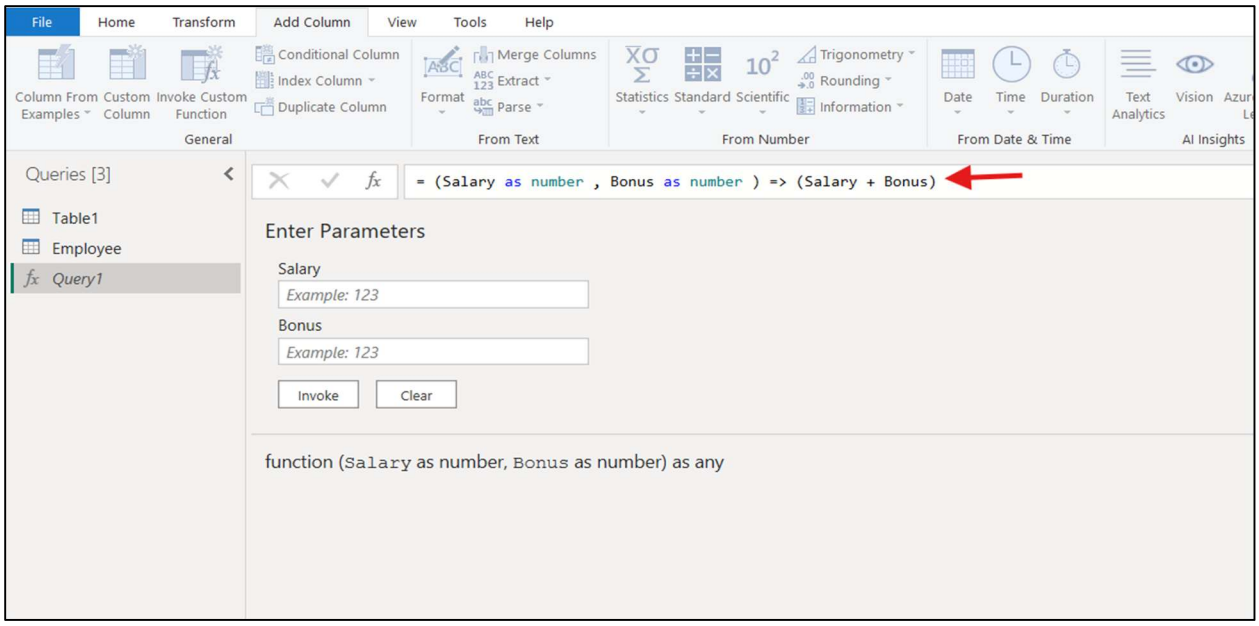


INVOKE CUSTOM FUNCTION

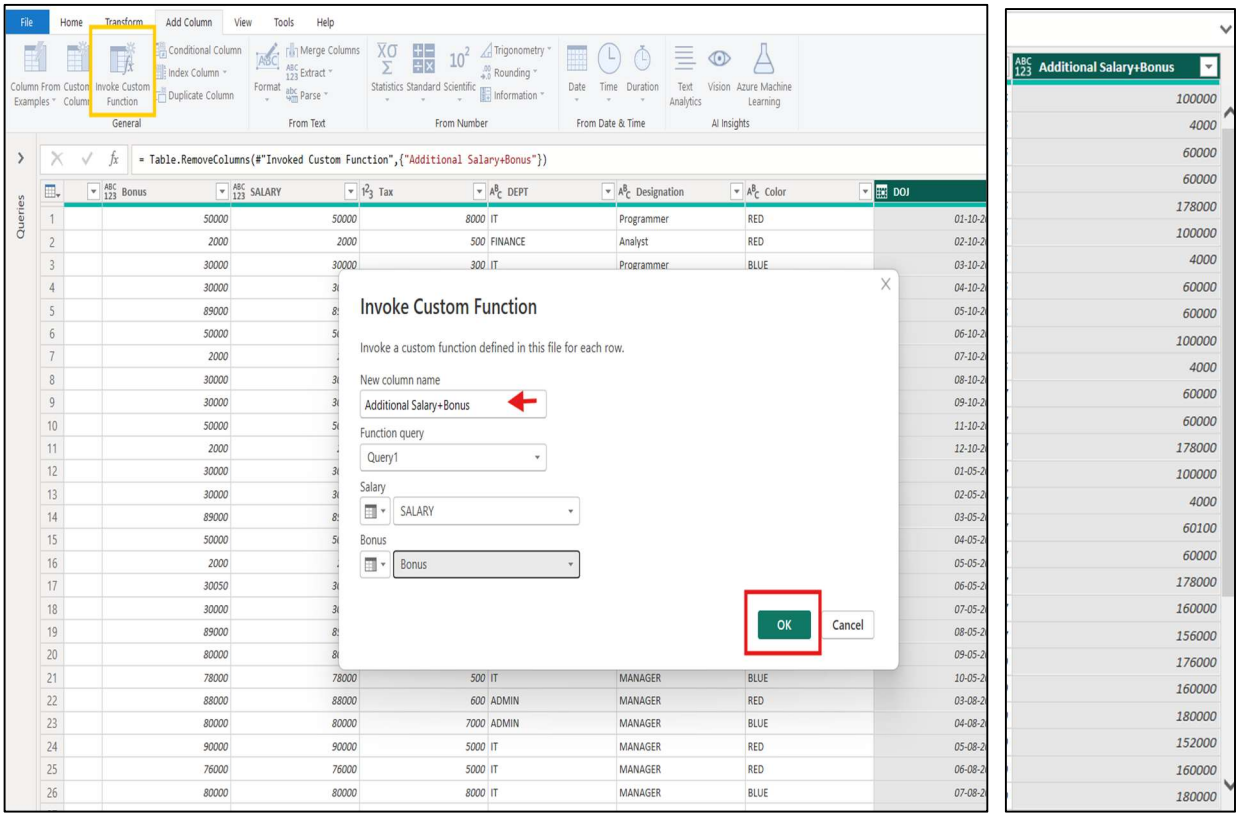
STEP 1: FIRST WE NEED TO GENERATE THE NEW QUERY



STEP 2: INSERT THE FORMULA IN THE QUERY GENERATED.



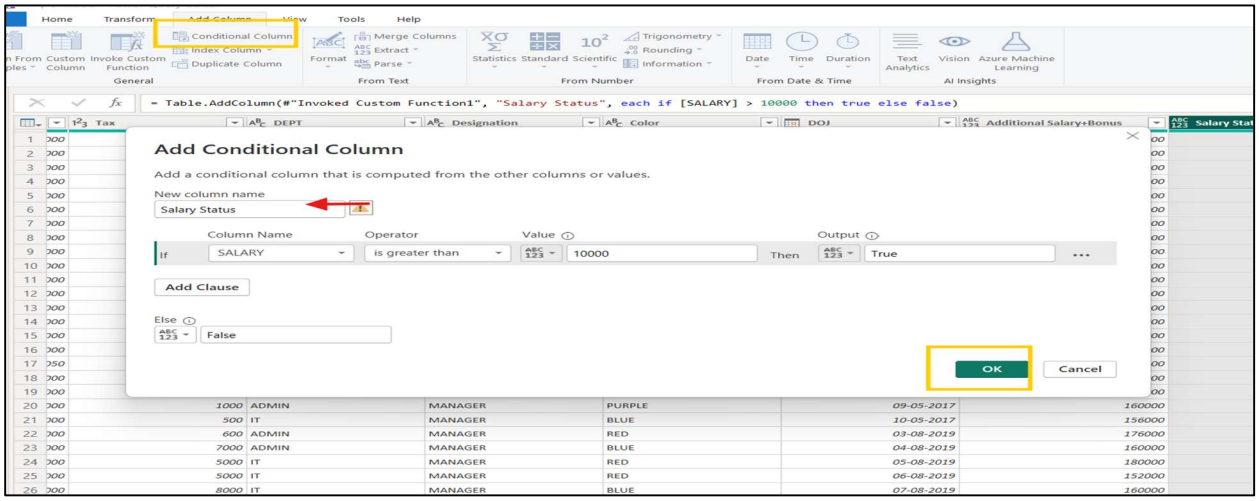
STEP 3: AFTER THAT CLICK ON THE INVOKE CUSTOM FUNCTION AND GIVE THE CONDITION IN THE IF-ELSE STATEMENT.



Note: Invoke Custom Function in Power BI allows users to apply reusable logic or calculations across queries. It helps automate data transformation using custom-defined functions.

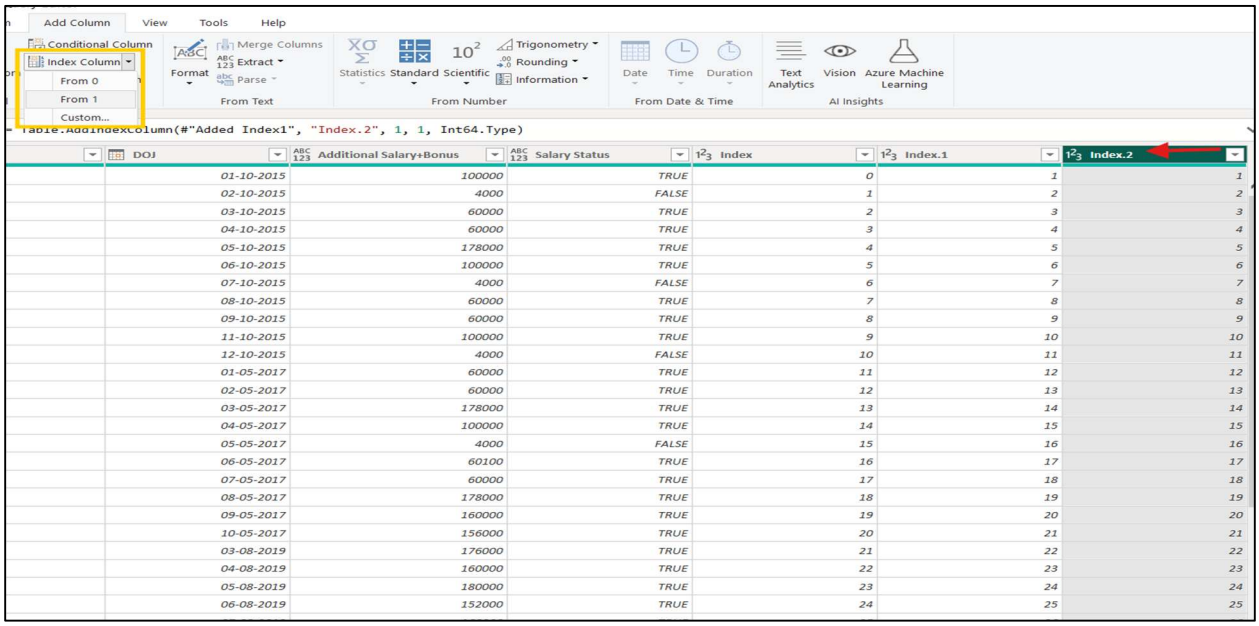
CONDITIONAL COLUMN

Create new column that conditionally add the values in the currently selected column



INDEX COLUMN

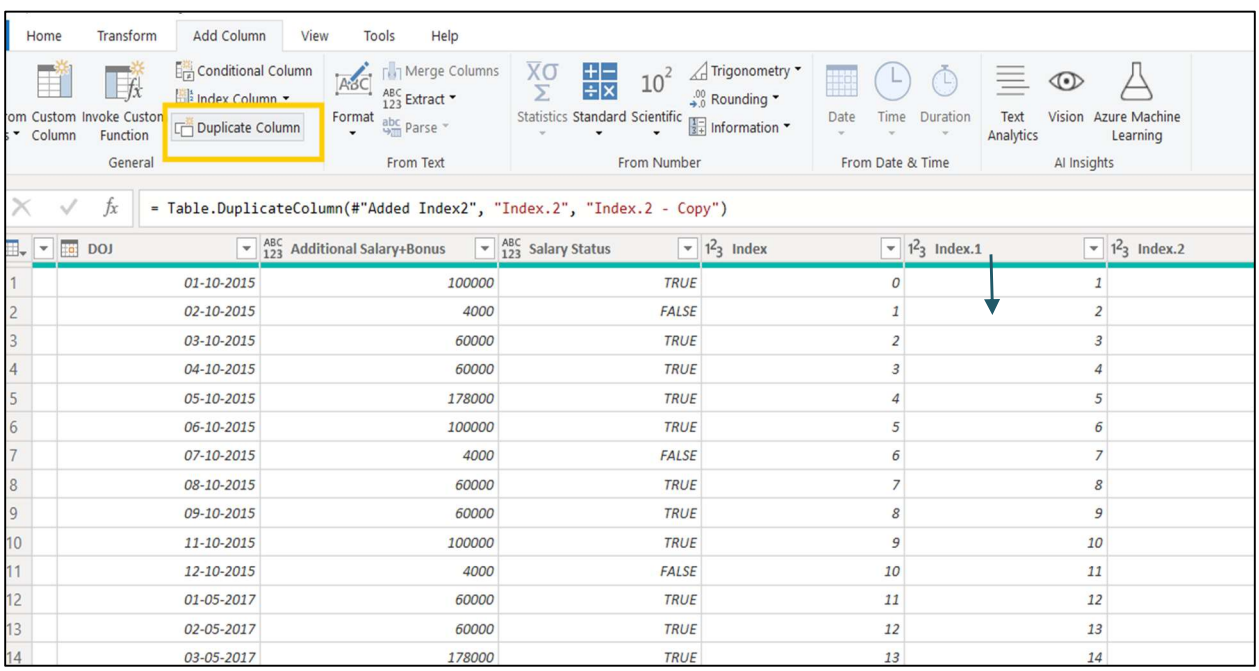
Creates a new column with an index starting at 0



DOJ	Additional Salary+Bonus	Salary Status	Index	Index.1	Index.2
01-10-2015	100000	TRUE	0	1	1
02-10-2015	4000	FALSE	1	2	2
03-10-2015	60000	TRUE	2	3	3
04-10-2015	60000	TRUE	3	4	4
05-10-2015	178000	TRUE	4	5	5
06-10-2015	100000	TRUE	5	6	6
07-10-2015	4000	FALSE	6	7	7
08-10-2015	60000	TRUE	7	8	8
09-10-2015	60000	TRUE	8	9	9
11-10-2015	100000	TRUE	9	10	10
12-10-2015	4000	FALSE	10	11	11
01-05-2017	60000	TRUE	11	12	12
02-05-2017	60000	TRUE	12	13	13
03-05-2017	178000	TRUE	13	14	14
04-05-2017	100000	TRUE	14	15	15
05-05-2017	4000	FALSE	15	16	16
06-05-2017	60100	TRUE	16	17	17
07-05-2017	60000	TRUE	17	18	18
08-05-2017	178000	TRUE	18	19	19
09-05-2017	160000	TRUE	19	20	20
10-05-2017	156000	TRUE	20	21	21
03-08-2019	176000	TRUE	21	22	22
04-08-2019	160000	TRUE	22	23	23
05-08-2019	180000	TRUE	23	24	24
06-08-2019	152000	TRUE	24	25	25

DUPLICATE COLUMN

Create new column that duplicates the value in the currently selected column



	DOJ	Additional Salary+Bonus	Salary Status	Index	Index.1	Index.2
1	01-10-2015	100000	TRUE	0	1	
2	02-10-2015	4000	FALSE	1	2	
3	03-10-2015	60000	TRUE	2	3	
4	04-10-2015	60000	TRUE	3	4	
5	05-10-2015	178000	TRUE	4	5	
6	06-10-2015	100000	TRUE	5	6	
7	07-10-2015	4000	FALSE	6	7	
8	08-10-2015	60000	TRUE	7	8	
9	09-10-2015	60000	TRUE	8	9	
10	11-10-2015	100000	TRUE	9	10	
11	12-10-2015	4000	FALSE	10	11	
12	01-05-2017	60000	TRUE	11	12	
13	02-05-2017	60000	TRUE	12	13	
14	03-05-2017	178000	TRUE	13	14	

QUESTIONS

1.What are Calculated Columns in Power BI and why are they important?
2.Explain the purpose of Custom Column and Conditional Column in Power Query Editor.
3.How does an Index Column help in organizing and analyzing data in Power BI?
4.What is the use of Invoke Custom Function in Power Query Editor?
5.How do calculated columns improve report flexibility and support advanced business analysis in Power BI

SUMMARY

This chapter focuses on enhancing data for analysis using advanced transformations and calculated columns. **Advanced transformations** allow complex data shaping, including summarizing with Group By, switching rows and columns with Transpose, reordering with Reverse Rows, and reshaping data using Pivot/Unpivot. Functional operations like formatting, extracting patterns, parsing XML/JSON, and applying mathematical, statistical, scientific, trigonometric, and date/time functions further prepare data for modeling. **Calculated columns** enable the creation of new fields based on existing data without altering the source, including Custom, Conditional, Index, Duplicate columns, and invoking custom functions. These tools help derive metrics like Profit = Sales – Cost, improve report flexibility, and support advanced analysis, empowering analysts to generate deeper business insights efficiently.

Calculated columns add derived data fields based on existing columns, enhancing analytical capabilities without altering the source data.

Common calculated column types:

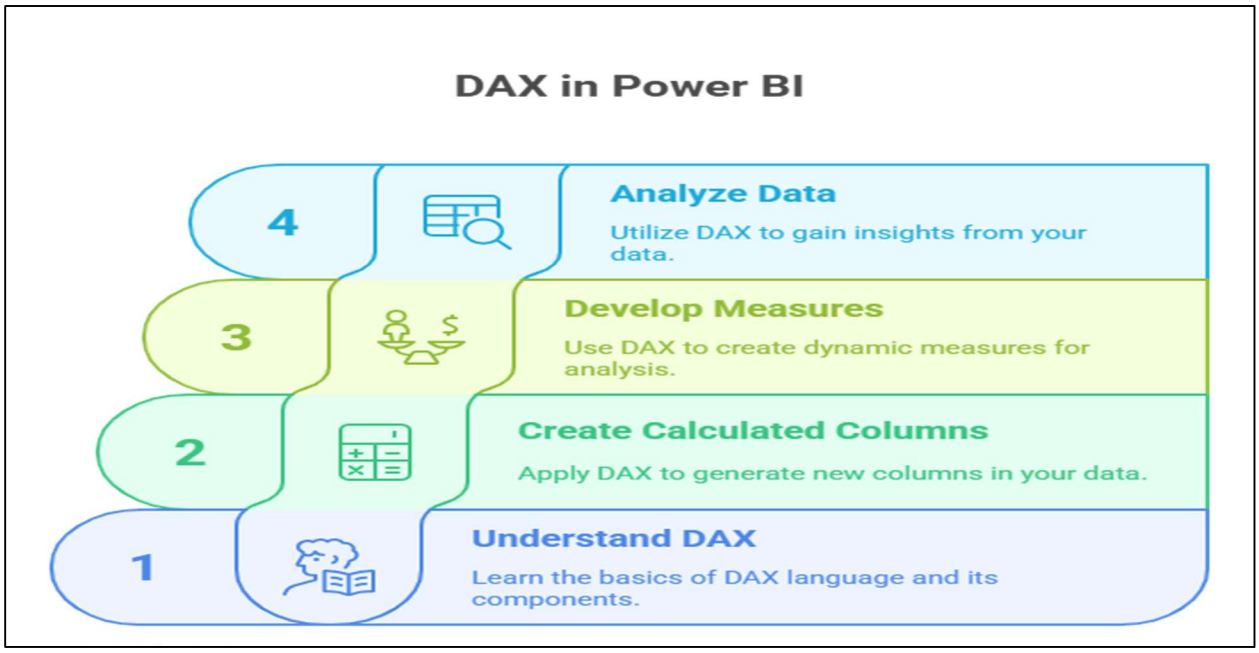
- Custom Column – Create new columns using custom formulas.
- Conditional Column – Add logic-based values (IF-ELSE conditions).
- Index Column – Add unique identifiers or sequence numbers.
- Duplicate Column – Create a copy of an existing column for transformation.
- Invoke Custom Function – Apply user-defined functions across data.

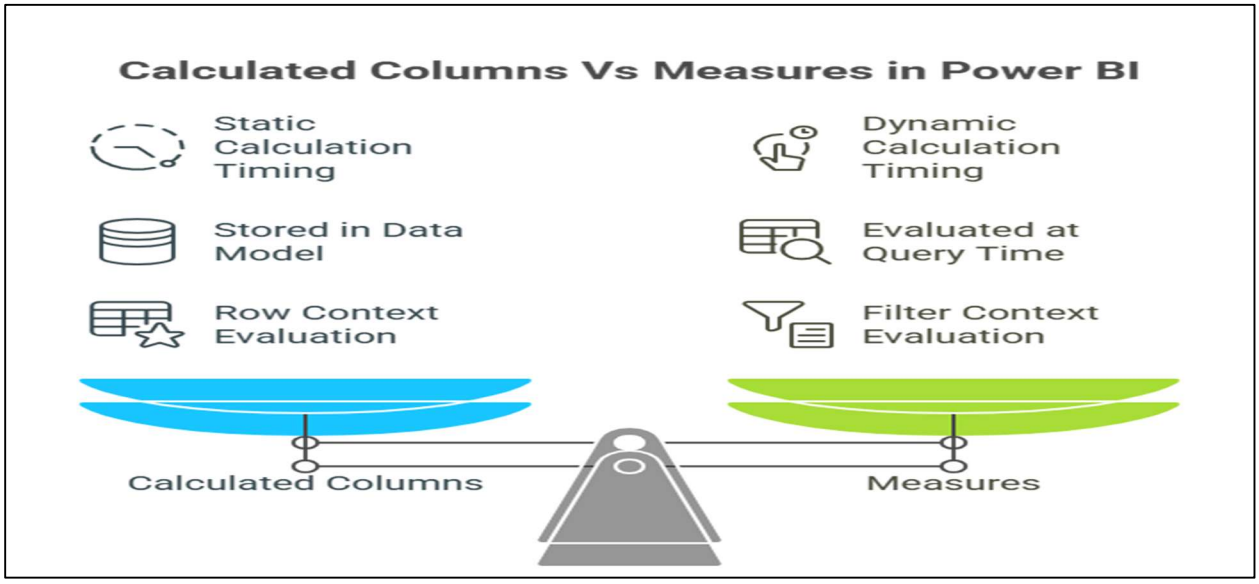
Calculated columns are essential in scenarios where derived metrics (e.g., Profit = Sales - Cost) are needed. They improve report flexibility and enable complex analysis directly within Power BI's data model.

CHAPTER 8: DATA ANALYSIS EXPRESSION (DAX)

In this Chapter, will explore how to use DAX in Power BI.

DAX is a formula language. It is the collection of functions, operators and constants that can be used in a formula or expression to calculate and return one or more value.





INTRODUCTION TO DAX

D*ata Analysis Expression language or DAX,* *was originally released with Power Pivot in Excel in 2010 by Microsoft.*

The DAX (Data Analysis Expressions) is a mathematical language used to build custom computations and measurements on data models in Power BI, Excel Power Pivot, and SQL Server Analysis Services (SSAS).

The DAX function's primary is used in analysing and summarizing data by generating sophisticated formulae and expressions. DAX allows users to execute complicated calculations and modify data by utilizing functions comparable to Excel formulae but with more advanced capabilities including time intelligence functions, table functions, and filtering functions.

DAX functions in Power BI show many similarities with Excel & SQL, but they are mostly used to perform relational tasks and calculations.

It's a functional language that is very similar to excel, as you would expect as it was part of the excel tools. Its initial basic function was to allow excel users solve more complex business problems.

DAX Consists of Operators, Functions, Expressions, and Filter Conditions etc.

DAX is used to create Calculated Columns, Measures and Tables in Power BI.

COLUMNS	MEASURES	TABLE
Columns are evaluated for each row	Measures are aggregated functions evaluated for the table.	

In general, we use calculated columns and measures to perform arithmetic operations on top of the data that we already have in our model.

Calculated columns can compute aggregate values, the evaluation context is performed by default for each row, which suits calculated columns for some computations, but not others.

Measures are used to calculate aggregates, such as the sum or average of a column.

Measures are calculated at the time of your query, which means that they aren't stored in your database, but use processing power to execute a query at the time of your request.

A simple example could be having a sales table with a Unit Price column and an Order Quantity column, and the desire to know the Order Total for each line, or in aggregate.

Now, not only is DAX used in Power Pivot but it is also the language used in Analysis Services tabular and Power BI.

OLTP / DB	SSAS - / CUBE	POWER BI / POWER PIVOT
SQL EXPRESSION	MDX	DAX

TYPES OF POWER BI DAX FUNCTIONS

Below are a few types of Power BI DAX functions:

- **Aggregation Functions: This function in Power BI is used to work with the numbers, it includes operations like SUM, AVERAGE, MIN, MAX, COUNT, COUNTA, DISTINCT COUNT, SUMX, AVERAGEX, MINX, and MAXX.**
- **Time Intelligence Functions: These functions provide insight into the date and dimensions of time, it includes operations like TOTALYTD, TOTALQTD, TOTALMTD, SAMEPERIODLASTYEAR, DATESYTD, and DATEADD.**
- Text Functions: This function works with tables and columns, it constitutes operations like CONCATENATE, UPPER, LOWER, SUBSTITUTE, TRIM, SEARCH, FIND, LEFT, and RIGHT.
- Logical Functions: It is one of the most unique DAX functions in Power BI, it works with returning the logical values and includes TRUE, FALSE, IF, AND, OR, NOT, SWITCH, and IFERROR.
- Information Functions: This function works by letting the user know if the value matches the expected type, it includes the following operations, ISBLANK, ISNUMBER, ISTEXT, and ISLOGICAL.
- Statistical Functions: This function performs the statistical operations and includes STDEV.P, VAR.P, MEDIAN, PERCENTILE, RANKX, and AVERAGEX.

Uses of DAX Functions in Power BI
Now that we know about the DAX function in power bi. Let us look through some of the uses that it offers and are based on the functions that it possesses:

- Creating columns: DAX functions can be used to create new columns in a table based on calculations that involve one or more columns in the table.
- Filtering Data: DAX functions can be used to filter data based on specific conditions, such as date ranges, text matches, or numerical values.
- Creating measures: DAX functions can be used to create new measures that aggregate data in a table or across tables, such as sum, average, and count.
- Creating logic: DAX functions can be used to create complex conditional logic that involves multiple conditions and operations.
- Creating hierarchies: DAX functions can be used to create hierarchies in a data model based on parent-child relationships between tables.

AGGREGRATE FUNCTIONS OF DAX

Function	Description
AVERAGE(<column>)	Returns the average (arithmetic mean) of all the numbers in a column.
COUNT(<column>)	The COUNT function counts the number of cells in a column that contain values. COUNT(CANDIDATES[RESUME])
COUNTBLANK(<column>)	Counts the number of blank cells in a column. (blank + null) values COUNTBLANK(CANDIDATES[RESUME])
COUNTROWS(<table>)	The COUNTROWS function counts the number of rows in the specified table, or in a table defined by an expression.
COUNT and COUNTA are identical in DAX for all the data types except Boolean. **COUNTA can operate on a Boolean data type, whereas COUNT cannot do that.**	
MAX(<column>)	Returns the largest numeric value in a column.
MIN(<column>)	Returns the smallest numeric value in a column. Ignores logical values and text.
SUM(<column>)	Adds all the numbers in a column.

FILTER AND VALUE FUNCTIONS (TABLE FUNCTIONS)

Function	Description
ALL(<table_or_column>)	Returns all the rows in a table, or all the values in a column, ignoring any filters that might have been applied.

Table2 = ALL(DimSalesTerritory) ALL(<table> or <column>) | ALL(<columnname1>,<columnname2>) Count of all sales orders = COUNTROWS(ALL('Sales')) NewCourses = ALL(COURSES_DETAIL[COURSENAME],COURSES_DETAIL[FEES])	
ALLEXCEPT(<table>,column1>,<column2>,…)	Overrides all context filters in the table except filters that have been applied to the specified columns.All tables excep the following columns.
BLANK()	Returns a blank.
CALCULATE(<expression>,<filter1>,<filter2>…)	Evaluates an expression in a context that is modified by the specified filters.

CALCULATETABLE(<expression>, <filter1>, <filter2>,…)	CALCULATETABLE(<table>, <filter1>, <filter2>,…) Evaluates a table expression in a context modified by filters. NewTable1 = CALCULATETABLE(LocationSales,Location[Country]="INDIA") SQLCandidatesCourses = CALCULATETABLE(CANDIDATES,COURSES_DETAIL[COURSENAME]="SQL SERVER") **DIFF : CALCULATE RETURNS A SINGLE VALUE AND CALCULATETABLE RETURNS A TABLE**
DISTINCT(<column>)	Returns a one-column table that contains the distinct values from the specified column.
FILTER(<table>,<filter>)	Returns a table that represents a subset of another table or expression.It is like a where conditions in the table. Extracts data as per the filter condition specified.
Table 2 = FILTER(DimSalesTerritory,DimSalesTerritory[SalesTerritoryKey]>5) Count of sales orders over 100 = COUNTROWS(FILTER('Sales', 'Sales'[Sales] > 100))	

Count of sales greater than 3000= COUNTROWS(FILTER(FactInternetSales, FactInternetSales[SalesAmount] > 3000))

VALUES(<column>)	Returns a one-column table that contains the distinct values from the specified column. This function is similar to DISTINCT function, but VALUES function can also return Unknown member
Table = VALUES(DimCustomer[AddressLine2]) CourseId = VALUES(CANDIDATES[COURSEID])	
ALLNONBLANKROW(?)	Returns all the rows, except for blank rows, in a table or column, and disregards any context filters that might exist.
Table = ALLNOBLANKROW(DimCustomer[AddressLine1])	
FIRSTNONBLANK(<column>,<expression>)	Returns the first non-blank values in column, filtered by expression. DateofFirstRegistration = CALCULATE(FIRSTNONBLANK(CANDIDATES[DateofRegistration],TRUE())) FirstOrderDate = CALCULATE(FIRSTNONBLANK(FactSales[Order_Date],TRUE()), FILTER(FactSales,YEAR(FactSales[Order_Date])=2014)) FirstOrderValue = CALCULATE(SUM(FactSales[Net_Sales]), FILTER(FactSales,YEAR(FactSales[Order_Date])=2014), FIRSTNONBLANK(FactSales[Order_Date],TRUE()))
LASTNONBLANK(<column>,<expression>)	The LASTNONBLANK() Function returns the last value in the column, filtered by the current context, where the expression is not blank. Syntax : LASTNONBLANK(<Column>,<Expression>) Returns the LAST non-blank values in column, filtered by expression. LastDateofRegistration = CALCULATE(LASTNONBLANK(CANDIDATES[DateofRegistration],TRUE()))

RELATED as the name itself says it will return the related value one table to another table. **This is similar to the lookup value function we have in MS Excel i.e. VLOOKUP. However in Power BI before we apply this DAX function we need to create a data model relationship between tables we are referring to.**

Below is the syntax of the RELATED DAX function in Power BI.

RELATED(COLUMN_NAME)

We just need to select the result column from the other table and it will fetch the details for us.

EXAMPLE : UnitPrice=RELATED(ProductPrice[Price])

Add a New Column CourseFees to Candidate Table in Power BI
CourseFees = RELATED(COURSES_DETAIL[FEES])

LOGICAL FUNCTIONS

Function	Description
AND(<logical1>,<logical2>,…) &&	Checks whether all arguments are TRUE, and returns TRUE if all arguments are TRUE. And(1,1) = true And(1,0) = false
FALSE()	Returns the logical value FALSE.
IF(logical_test>, <value_if_true>, value_if_false)	Checks if a condition provided as the first argument is met. Returns one value if the condition is TRUE, and returns another value if the condition is FALSE. If(tax> 100,”Higher”,”Lower”) CandidateStatus = if(CANDIDATES[CourseFees] > 10000,"High","Low")
IFERROR(value, value_if_error)	Returns value_if_error if the first expression is an error and the value of the expression itself if otherwise.

NOT(<logical>)	Changes FALSE to TRUE, or TRUE to FALSE.
TRUE()	Returns the logical value TRUE.
OR(<logical1>,<logical2>,…)	Checks whether one of the arguments is TRUE to return TRUE. The function returns FALSE if all arguments are FALSE OR(1,0) = true

- EXAMPLES

```
RedOrBlack Sales IN := CALCULATE (  [Sales Amount], Products[Color] IN { "Red", "Black" } )

WeekDaySales = CALCULATE(sum(factTable[Sales]),WEEKDAY(factTable[Date]) in{,2,3,4,5,6})

WeekEndSales = CALCULATE(SUM(factTable[Sales]),WEEKDAY(factTable[Date]) in{1,7})
```

- SWITCH –CASE IN POWER BI

SWITCH (COLNAME, WHENVALUE1,CALVALUE1, WHENVALUE2,CALVALUE2)	ConvertedIndianPrice = SWITCH('Product'[CURRENCY], "DOLLAR','Product'[UNIT PRICE] * 72, "POUND','Product'[UNIT PRICE] * 90, "RUPPEE','Product'[UNIT PRICE] * 1, "RAND','Product'[UNIT PRICE] * 7)

INFORMATION FUNCTIONS

Function	Description
ISBLANK(<value>)	Checks whether a value is blank, and returns TRUE or FALSE.
ISERROR(<value>)	Checks whether a value is an error, and returns TRUE or FALSE.

ISLOGICAL(<value>)	Checks whether a value is a logical value, (TRUE or FALSE), and returns TRUE or FALSE.
ISNONTEXT(<value>)	Checks whether a value is not text (blank cells are not text), and returns TRUE or FALSE.
ISNUMBER(<value>)	Checks whether a value is a number, and returns TRUE or FALSE. Column 2 = ISNUMBER("test") = false
ISTEXT(<value>)	Checks whether a value is text, and returns TRUE or FALSE.

MATH AND TRIG FUNCTIONS

Function	Description
ABS(<number>)	Returns the absolute value of a number. COL1 = ABS(101.11) = 101.11
CEILING(<number>, <significance>)	Rounds a number up, to the nearest integer or to the nearest multiple of significance. COL1 = CEILING(101.1111,1) = 102 = CEILING(4.6,1) returns 5.
EXP(<number>)	Returns e raised to the power of a given number. The constant e equals 2.71828182845904, the base of the natural logarithm.
FACT(<number>)	Returns the factorial of a number, equal to the series 1*2*3*...* , ending in the given number. COL1 = FACT(4) | OUTPUT : 24 4 * 3*2

FLOOR(<number>, <significance>)	Rounds a number down, toward zero, to the nearest multiple of significance. FLOOR(101.1,1) = 101
INT(<number>)	Rounds a number down to the nearest integer.
LN(<number>)	Returns the natural logarithm of a number. Natural logarithms are based on the constant e (2.71828182845904).
LOG(<number>,<base>)	Returns the logarithm of a number to the base you specify.
LOG10(<number>)	Returns the base-10 logarithm of a number.
MOD(<number>, <divisor>)	Returns the remainder after a number is divided by a divisor. The result always has the same sign as the divisor. MOD(14,5) = 4 MOD(14,2) = 0
PI()	Returns the value of Pi, 3.14159265358979, accurate to 15 digits. pi() * 2 = 6.28
POWER(<number>, <power>)	Returns the result of a number raised to a power. POWER(10,3) = 1000 POWER(10,4)=10000
QUOTIENT(<numerator>, <denominator>)	Performs division and returns only the integer portion of the division result. Use this function when you want to discard the remainder of division. QUOTIENT(10,2)= 5 11,2 = 5 reminder: 1

ROUND(<number>, <num_digits>)	Rounds a number to the specified number of digits. ROUND(10.111,2)=10.11 ROUND(10.567,2) = 10.57
ROUNDDOWN(<number>, <num_digits>)	Rounds a number down, toward zero. = ROUNDDOWN (5.55555,2) returns 5.55. = ROUNDDOWN (5.55555,0) returns 5.
ROUNDUP(<number>, <num_digits>)	Rounds a number up, away from 0 (zero). = ROUNDUP (5.55555,2) returns 5.56. = ROUNDUP (5.55555,0) returns 6.
SIGN(<number>)	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.
SQRT(<number>)	Returns the square root of a number.
TRUNC(<number>)	Truncates a number to an integer by removing the decimal, or fractional, part of the number. TRUNC (5.8) returns 5.
[vb] RAND()	Returns a random number greater than or equal to 0 and less than 1, evenly distributed. The number that is returned changes each time the cell containing this function is recalculated.
[vb] RANDBETWEEN(<bottom>,<top>)	**Returns a random number between the numbers you specify**

TEXT FUNCTIONS

Function	Description
UNICODE(<text>)	Returns a numeric code for the first character in a text string, in the character set used by your computer.
EXAMPLE : NEWNAME = UNICODE(DimCustomer[FirstName])	
CONCATENATE(<text1>, <text2>,…)	The CONCATENATE function joins multiple text strings into one text string. The joined items can be text, numbers or Boolean values represented as text, or a combination of those items. You can also use a column reference if the column contains appropriate values.
EXAMPLE : NEWNAME = CONCATENATE(DimCustomer[FirstName], DimCustomer[LastName])	
EXACT(<text1>,<text2>)	Compares two text strings and returns TRUE if they are exactly the same, FALSE otherwise. EXACT is case-sensitive but ignores formatting differences. You can use EXACT to test text being entered into a document.
EXAMPLE : NEWNAME = EXACT(DimCustomer[FirstName],DimCustomer[LastName])	
FIND(<find_text, within_text, start_num)	Returns the starting position of one text string within another text string. FIND is case-sensitive.
EXAMPLE : NEWNAME = FIND("AND",DimCustomer[LastName],1,100) NEWNAME = FIND("AND",DimCustomer[LastName],11,20)	
FIXED(<number>, <decimals>, <no_commas>)	Rounds a number to the specified number of decimals and returns the result as text. You can specify that the result be returned with or without commas.
Column 2 = FIXED(100,2,TRUE()) = 100.00	

LEFT(<text>, <num_chars>)	Returns the specified number of characters from the start of a text string.
Column 2 = LEFT("amitsharma",4) = amit	
LEN(<text>)	Returns the number of characters in a text string.
LOWER(<text>)	Converts all letters in a text string to lowercase.
MID(<text>, <start_num>, <num_chars>)	Returns a string of characters from the middle of a text string, given a starting position and length.
MID("amitsharma",3,3) = its	MID("amitsharma",5,5) = sharm
REPLACE(<old_text>, <start_num>, <num_chars>, <new_text>)	REPLACE replaces part of a text string, based on the number of characters you specify, with a different text string.
REPLACE("dillipparhi", 7, 2, "das") = dillipdasrhi	
REPT(<text>, <num_times>)	Repeats text a given number of times. Use REPT to fill a cell with a number of instances of a text string
RIGHT(<text>, <num_chars>)	RIGHT returns the last character or characters in a text string, based on the number of characters you specify.
SEARCH(<search_text>, <within_text>, [start_num])	Returns the number of the character at which a specific character or text string is first found, reading left to right. Search is case-sensitive.
SUBSTITUTE(<text>, <old_text>, <new_text>, <instance_num>)	Replaces existing text with new text in a text string.
TRIM(<text>)	Removes all spaces from text except for single spaces between words.
UPPER (<text>)	Converts a text string to all uppercase letters

VALUE(<text>)	Converts a text string that represents a number to a number.
Column 2 = VALUE("11")=11	
FORMAT(<value>, <format_string>)	**Converts a value to text according to the specified format.**
FORMAT (0.742, "Percent") -- Returns 74.20% FORMAT (1234.567, "$#,0.00") -- Returns $1,234.57 FORMAT (1234.567, """US$"" #,0.00") -- Returns US$ 1,234.57 FORMAT (DATE (2019, 3, 28), "yyyy-mm-dd") -- Returns 2019-03-28 FORMAT (DATE (2019, 3, 28), "m/d/yy") -- Returns 3/28/19 FORMAT(TableName[Duration]/60, "HH:mm:ss") --	

DATE AND TIME FUNCTIONS

Function	Description
DATE(<year>, <month>, <day>)	Returns the specified date in datetime format.
DD = DATE("1990","01","10")	
DATEVALUE(date_text)	Converts a date in the form of text to a date in datetime format
DAY(<date>)	Returns the day of the month, a number from 1 to 31.
EDATE(<start_date>, <months>) Add months	Returns the date that is the indicated number of months before or after the start date. Use EDATE to calculate maturity dates or due dates that fall on the same day of the month as the date of issue
EDATE("01/10/2017",3) = 04/10/2017	
EOMONTH(<start_date>, <months>)	Returns the date in datetime format of the last day of the month, before or after a specified number of months. Use EOMONTH to

Add months and give the last day of month	calculate maturity dates or due dates that fall on the last day of the month.
EOMONTH("01/10/2017",3)	4/30/2017
HOUR(<datetime>)	Returns the hour as a number from 0 (12:00 A.M.) to 23 (11:00 P.M.).
MINUTE(<datetime>)	Returns the minute as a number from 0 to 59, given a date and time value.
MONTH(<datetime>)	Returns the month as a number from 1 (January) to 12 (December).
NOW()	Returns the current date and time in datetime format.
SECOND(<time>)	Returns the seconds of a time value, as a number from 0 to 59
TIME(hour, minute, second)	Converts hours, minutes, and seconds given as numbers to a time in datetime format.
TIMEVALUE(time_text)	Converts a time in text format to a time in datetime format.
TODAY()	Returns the current date.
WEEKDAY(<date>, <return_type>)	Returns a number from 1 to 7 identifying the day of the week of a date. By default the day ranges from 1 (Sunday), 2 Monday to 7 (Saturday).
Column 2 = WEEKDAY("10/10/2018",2) = 3	
WEEKNUM(<date>, <return_type>)	Returns the week number for the given date and year according to the specified convention. The week number indicates where the week falls numerically within a year.
YEAR(<date>)	Returns the year of a date as a four digit integer in the range 1900-9999.
YEARFRAC(<start_date>, <end_date>, <basis>)	Calculates the fraction of the year represented by the number of whole days between two dates. Use the YEARFRAC worksheet function

	to identify the proportion of a whole year's benefits or obligations to assign to a specific term.
YEARFRAC("10/10/2018","10/10/2020",4)=2	

USEFUL DATE FUNCTIONS

CALENDAR : RETURNS A TABLE WITH THE LIST OF ALL THE DATES SPECIFIED IN THE STARTDATE AND ENDDATE Calendar(startdate,enddate) DimDate= CALENDAR (DATE (2005, 1, 1), DATE (2015, 12, 31)) The following formula returns a table with dates between January 1st, 2005 and December 31st, 2015.
DATESBETWEEN(<column>,<start_date>,<end_date>) Returns a table. returns all dates between the starting date to the ending date. Returns a table of dates that can be found in the specified date column beginning with the start date and ending with the end date. Calendar = DATESBETWEEN(LocationSales[Date],FIRSTDATE(LocationSales[Date]),LASTDATE(LocationSales[Date])) datesdata = DATESBETWEEN(CANDIDATES[DateofRegistration].[Date],FIRSTDATE(CANDIDATES[DateofRegistration].[Date]),LASTDATE(CANDIDATES[DateofRegistration].[Date]))

TIME INTELLIGENCE FUNCTIONS

Function	Description
TotalMTD(<expression>,<dates>,<filter>) QTDTotalsalesthisyear = TOTALMTD(SUM(LocationSales[Profit]),LocationSales[Date])	Evaluates the specified expression for the interval that starts at the first day of the month and ends at the latest date in the specified dates column, after applying all filters.
TotalQTD(<expression>,<dates>,<filter>) QTDTotalsalesthisyear = TOTALQTD(SUM(LocationSales[Profit]),LocationSales[Date])	Evaluates the specified expression for the interval that starts at the first day of the quarter and ends at the latest date in the specified dates column, after applying all filters.
TotalYTD(<expression>,<dates>,<filter>)	Evaluates the specified expression for the interval that starts at the first day of the year and ends at the latest date in the specified dates column, after applying all filters.
Totalsalesthisyear = TOTALYTD(SUM(FactInternetSales[SalesAmount]), DimDate[FullDateAlternateKey],FILTER(DimDate,DimDate[CalendarYear]=2008)) MTotalsalesthisyear = TOTALYTD(SUM(LocationSales[Profit]),LocationSales[Date])	
SAMEPERIOD-LAST YEAR : Return the last year total PreviousYear = CALCULATE(sum(factTable[Sales]),SAMEPERIODLASTYEAR(factTable[Date]))	

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

PARALLELPERIOD(<dates>,<number_of_intervals>,<interval>)

- Dates – A column that contains dates.
- Number_of_intervals – an integer that specifies the number of intervals to add to or subtract from the dates.
- Interval – the interval by which to shift the dates. The value for interval can be one of the following, year, quarter or month.

```
StudentMonthParallelCount =
CALCULATE(COUNT(CANDIDATES[CANDIDATE_CODE]),PARALLELPERIOD(CANDIDATES[Dateof
Registration].[Date],-1,MONTH))
StudentYearParallelCount=
CALCULATE(COUNT(CANDIDATES[CANDIDATE_CODE]),PARALLELPERIOD(CANDIDATES[Dateof
Registration].[Date],-1,YEAR))
LastYearStudentCount =
CALCULATE(COUNT(CANDIDATES[CANDIDATE_CODE]),SAMEPERIODLASTYEAR(CANDIDATES[
DateofRegistration].[Date]))
```

FUNCTIONS

Function	Description
CLOSINGBALANCEYEAR(<expression>,<dates>,<filter>) Last day of the year	Evaluates the specified expression at the calendar end of the given year. The given year is calculated as the year of the latest date in the dates argument, after applying all filters. Calculate the expression on the last day of the year.
Example : Revenue on the last day of the year closingBalanceYear = CLOSINGBALANCEYEAR(SUM(TransactionData[Revenue]),CalenderDate[Date]) LastYear = CLOSINGBALANCEYEAR(COUNT(CANDIDATES[CANDIDATE_CODE]),CANDIDATES[DateofRegistration]) LastYearClosingProfit = CLOSINGBALANCEYEAR(SUM(LocationSales[Profit]),LocationSales[Date]) InventoryTotal = CLOSINGBALANCEYEAR(SUMX(ProductInventory,ProductInventory[UnitCost]*ProductInventory[UnitsBalance]),DateTime[DateKey])	
CLOSINGBALANCEQUARTER : Last day of the Quarter CLOSINGBALANCEQUARTER(<expression>,<dates>,<filter>)	
CLOSINGBALANCEMONTH : Last day of the Month CLOSINGBALANCEMONTH(<expression>,<dates>,<filter>)	
DATEADD(<date_column>,<number_of_intervals>,<interval>)	Returns a table that contains a column of dates, shifted either forward in time or back in time from the dates in the specified date column.
DATESYTD (<Dates>[, <StartDate>])	Returns a set of dates after the startdate specified.all dates after the start date will be displayed

```
DATESYTD ( <Dates>[, <StartDate>] )

WILL RETURN ALL THE DATES, PRESENT AFTER THE STARTDATE SPECIFIED
```

```
Measure 2 = CALCULATE ( [Total], DATESYTD ( 'Calendar'[Date] ) )
YearCount = CALCULATE ( COUNT(CANDIDATES[CANDIDATE_CODE]), DATESYTD
(CANDIDATES[DateofRegistration].[Date] ) )
TableYear = DATESYTD(CANDIDATES[DateofRegistration].[Date])
TableYear = DATESYTD(CANDIDATES[DateofRegistration].[Date],"05/15/2020")
```

```
DATESMTD ( <Dates> )
Returns a set of dates in the month up to the last date visible in the filter context.
MonthYear = DATESMTD(CANDIDATES[DateofRegistration].[Date])
```

```
DATEADD = CALCULATE([Total Sales], DATEADD('Calendar' [Date], -1, YEAR))

SAMEPERIODLASTYEAR = CALCULATE([Total Sales], SAMEPERIODLASTYEAR('Calendar'
[Date]))
```

TABLE FUNCTIONS IN DAX

DataTable Function Syntax

```
=DataTable("ColumnName", DataType,
           "ColumnName", DataType
           ,{
{"col1value","col2value" },
                 {" col1value "," col2value "}
            }
        )
```

```
=DataTable("Name", STRING,
           "Region", STRING
           ,{
                                {" User1","East"},
                 {" User1","East"},
                 {" User2","East"},
                 {" User3","West"},
                 {" User4","West"},
                 {" User4","East"}
            }
        )
```

UNION, INTERSECT, EXCEPT

UNION : UNION COMIBINES THE ROWS OF TWO OR MORE TABLES
UNION(TABLE1, TABLE2) UNION(UsaInventory, IndInventory) FinalProductsData = UNION(Product1,Product2) tblUnionResults = UNION(Country1,Country2)
INTERSECT : FINDS COMMON RECORDS IN TWO TABLES
INTERSECT(table1,table2) tblIntersectResults = INTERSECT(Country1,Country2)
EXCEPT : TABLE1 MINUS TABLE2
tblExceptResults = EXCEPT(Country1,Country2) Except(States1, States2)
NATUTAL INNER JOIN
tblJoins = NATURALINNERJOIN(DEPTS,Employee) tblJoins = NATURALLEFTOUTERJOIN(DEPTS,Employee)

ITERATOR FUNCTIONS IN DAX

Iterators in DAX are functions that iterate through all rows of the given table, apply the expression, and then aggregate the result. One of the known functions in this category is SUMX. because an iterator function goes through every row and apply the transformation, then the result is materialized (temporary) before aggregating it, and that causes the totals to be actual sum (or other aggregations) of values in a column.
BudgetProfit = CALCULATE(SUMX(factTable,factTable[Budget Profit]))

SUM() vs SUMX()

- SUM() is an aggregator function. It adds up all the values in a column you specify in the current filter context. SUM() has no knowledge of the existence of rows – all it can do is add everything in the single column it is presented with.
- SUMX() is an iterator function. It works through a table, row by row to complete the evaluation in the current filter context. SUMX() has awareness of rows in a table, and hence can reference the intersection of each row with any columns in the table.

```
SUMX (<table>, <expression>)
TotalSales = CALCULATE(Sumx(Sales,[Price]*[Quantity]))
Add a Profit Column to Product Table in CoffeeShop Data
Profit = CALCULATE(SUMX(factTable,factTable[Profit]))
BudgetProfit = CALCULATE(SUMX(factTable,factTable[Budget Profit]))
```

```
Status = IF((('Product'[Profit] > 'Product'[BudgetProfit]) && ('Product'[Profit] >
20000)),"HighProfitProduct","LowProfitProduct")
```

```
Status = IF((('Product'[Profit] > 'Product'[BudgetProfit]) && ('Product'[Profit] >
20000)),"HighProfitProduct","LowProfitProduct")
```

RETURN

The RETURN keyword consumes variables defined in previous VAR statements.

```
VAR <name> = <expression>
[VAR <name2> = <expression2> [...]]
RETURN <result_expression>
```

```
SalesIncrease =
var sales2019 = TOTALYTD(SUM(factTable[Sales]),factTable[Date].[Date])
var sales2018 =
CALCULATE(SUM(factTable[Sales]),PARALLELPERIOD(factTable[Date].[Date],-1,YEAR))
return (sales2019-sales2018)
```

```
GrowthPer =
VAR  TotalStudents = COUNT( CANDIDATES[CANDIDATE_CODE] )
VAR  TotalStudentsLastYear2019 = CALCULATE (
      COUNT ( CANDIDATES[CANDIDATE_CODE] ),
      SAMEPERIODLASTYEAR ( CANDIDATES[DateofRegistration].[Date])
 )
VAR  TotalStudents2020 =
     TOTALYTD(COUNT ( CANDIDATES[CANDIDATE_CODE] ),
      CANDIDATES[DateofRegistration].[Date])
RETURN       DIVIDE ( TotalStudents2020,    TotalStudentsLastYear2019        )
```

RANK ETC

SUMMARIZE Syntax

```
SUMMARIZE(<table>, <groupBy_columnName>[, <groupBy_columnName>]…[, <name>,
<expression>]…)
SummResults = SUMMARIZE(factTable,'Product'[Product],"SalesAmount"
,SUM(factTable[Sales]),"Profit",sum(factTable[Profit]))
DAX
SUMMARIZE(ResellerSales_USD,DateTime[CalendarYear]
,ProductCategory[ProductCategoryName],"Sales Amount (USD)",
SUM(ResellerSales_USD[SalesAmount_USD])
     , "Discount Amount (USD)", SUM(ResellerSales_USD[DiscountAmount]) )
```

SUMMARIZE with ROLLUP

The addition of the ROLLUP() syntax modifies the behavior of the SUMMARIZE function by adding roll-up rows to the result on the groupBy_columnName columns.

DAX

SUMMARIZE(<table>, <groupBy_columnName>[, <groupBy_columnName>]…[, ROLLUP(<groupBy_columnName>[,< groupBy_columnName>…])][, <name>, <expression>]…)
Summ_WithRollUp =
SUMMARIZE(FactInternetSales,ROLLUP(DimSalesTerritory[SalesTerritoryCountry],DimDate[CalendarYear]),"TotalSales",SUM(FactInternetSales[SalesAmount]),"TotalFreight",sum(FactInternetSales[Freight]))

The Power BI DAX RANKX function is a powerful sortation function. We will give an overview of what the RANKX function is capable of and make sure that you understand the basics. This is a super useful function so we will also explore some more complex methods of using RANX with practical examples.

RANKX(<table>, <expression>, <value>, <order>, <ties>)

RankId = RANKX('Product','Product'[ConvertedIndianPrice],,ASC)
RANX Total Sales ASC =RANKX(‘Sales Table’, ‘Sales Table'[Total Sales], , ASC)
Note here that there will be a ‘gap’ in the rank values in RANKX’s default state or if we pass SKIP into the ties parameter. For this example, the next rank value will be 5, as our rank values are 1, 2, 3, 3, 5.
If we pass DENSE in the ties parameter, it will not allow for gaps in the value of RANKX, meaning that in our used example the next rank value after our tied RANKX value of 3, will be 4.

Returns the rank of a number in a column of numbers. If more than one value has the same rank, the top rank of that set of values is returned.
RANK.EQ (<value>, table[column]) | RANK.EQ (<value>, table[column], <order>)

An order quantity for 174,728 units is received. Find the rank of this order in terms of quantity compare to all other order quantities.

Answer : Deinfe the following measures in Power BI:

rank_it = RANK.EQ(174728,TransactionData[Quantity])

TOPN(<n_value>, <table>, <orderBy_expression>, [<order>[, <orderBy_expression>, [<order>]]…])

Top3Products =
TOPN(3,VALUES('Product'[Product]),CALCULATE(SUM('Product'[ConvertedIndianPrice])),DESC)

```
TopNSubCategory =
TOPN(3,VALUES(DimProductSubcategory[EnglishProductSubcategoryName]),CALCULATE(SU
M(FactInternetSales[SalesAmount])))
```

IMPORTANT DAX EXAMPLES

```
CumulativeTotal =  CALCULATE (
SUM(factTable[Sales]),FILTER (ALL( factTable[Date] ),'factTable'[Date] <= MAX (
factTable[Date] )))
```

```
// commulative sales for each year
DATESYTD =  CALCULATE(SUM(factTable[Sales]), DATESYTD(factTable[Date]))
```

```
PTotalNewSales = CALCULATE(SUMX(factTable,factTable[Sales]))
Total sales of top 3 products
BestProductSales =
VAR BestProducts = TOPN ( 3, 'Product', [PTotalNewSales] )
VAR SalesOfBestProducts = SUMX ( BestProducts,SUMX ( RELATEDTABLE ( 'Product' ),
Product'[PTotalNewSales] ) )
RETURN   SalesOfBestProducts
```

IMPORTANT

```
CURRENTYEAR
```

```
CurrentYearDays = TOTALYTD(COUNT(DimDates[Date]),DimDates[Date])
CurrentYearDayDYTD = CALCULATE(COUNT(DimDates[Date]),DATESYTD(DimDates[Date]))
CurrentYearFYDays = TOTALYTD(COUNT(DimDates[Date]),DimDates[Date],"31/03")
```

```
LASYYEAR
```

```
PreviousYearDays = CALCULATE([CurrentYearDays],SAMEPERIODLASTYEAR(DimDates[Date]))
PreviousYearTotalDays = CALCULATE([CurrentYearDays],PARALLELPERIOD(DimDates[Date],-
1,YEAR) )
PreviousYearDaysNew =
 var lastdate1 = CALCULATE(LASTDATE(DimDates[Date]),DATEADD(DimDates[Date],-365,DAY))
 var firstdate1 = DATE(YEAR(lastdate1),01,01)
 RETURN
CALCULATE(COUNT(DimDates[Date]),DATESBETWEEN(DimDates[Date],firstdate1,lastdate1))
```

```
CURRENTMONTH
```

```
CurrentMonthTotal = TOTALMTD(COUNT(DimDates[Date]),DimDates[Date])
CurrentMonthTotalN = CALCULATE( COUNT(DimDates[Date]),DATESMTD(DimDates[Date]))
```

```
LASTMONTH
```

```
PreviouMonthTotal = CALCULATE([CurrentMonthTotal],PARALLELPERIOD(DimDates[Date],-
1,MONTH))
```

SameMonthLastYear = CALCULATE([CurrentMonthTotal],SAMEPERIODLASTYEAR(DimDates[Date])) **USING PREVIOUS MONTH** PreviouMonthTotal1 = CALCULATE([CurrentMonthTotal], PREVIOUSMONTH(LASTDATE(DimDates[Date]))) TODAY TodayCount = CALCULATE(COUNT(DimDates[Date]), FORMAT(DimDates[Date], "dd/mm/yyyy") = FORMAT(TODAY(),"dd/mm/yyyy")) TodayCount = CALCULATE(COUNT(WEB_VISITORS_DETAILS[VISITORID]), FORMAT(WEB_VISITORS_DETAILS[VISITEDDATE], "dd/mm/yyyy") = FORMAT(TODAY(),"dd/mm/yyyy")) YESTERDAY YesterdayCount = CALCULATE([TodayCount],PREVIOUSDAY(LASTDATE(DimDates[Date])))

QUESTIONS

1.What is DAX and where is it used in Power BI?
2.What is the difference between Calculated Columns and Measures in DAX?
3.Explain the purpose of aggregation functions such as SUM, AVERAGE, COUNT, and MAX in DAX.
4.How does the CALCULATE function modify filter context in DAX?
5.What are Time Intelligence Functions in DAX and why are they important in business reporting?
6.What is DAX and where is it used in Power BI?
7.How to find Today's and Previous Sales
8.Use PREVDAY,PREVMONTH,PREVMONTH. NEXTDAY, NEXTMONTH etc

SUMMARY

DAX (Data Analysis Expressions) is Power BI's formula language used for performing calculations and advanced data analysis across tables. It includes **Time Intelligence functions** like TOTALYTD(), TOTALQTD(), and TOTALMTD() to calculate Year-to-Date, Quarter-to-Date, and Month-to-Date metrics. **Text Functions** allow combining or manipulating text values, while **Filter Functions** such as CALCULATE() and FILTER() enable conditional aggregations. **Information Functions** like ISBLANK() and CONTAINS() help validate data and check for nulls. **Date & Time Functions** assist in creating calendar tables and performing date-based calculations. Together, DAX functions empower analysts to build dynamic calculations, perform complex analyses, and create flexible, insightful, and interactive reports in Power BI.

DAX (Data Analysis Expressions) is a formula language used in Power BI to perform calculations and data analysis across tables.

Common DAX categories:

- Time Intelligence – Calculate YTD, QTD, and MTD metrics using TOTALYTD(), TOTALQTD(), TOTALMTD().
- Text Functions – Combine text values using '&' or spaces.
- Filter Functions – Apply CALCULATE() and FILTER() for conditional aggregations.
- Information Functions – Validate data types or null values (e.g., ISBLANK(), CONTAINS()).
- Date & Time Functions – Create calendar tables and manipulate date fields.

DAX empowers users to build calculated measures, KPIs, and advanced business metrics that enhance Power BI's analytical and reporting power.

CHAPTER 9: DEPLOYING, SECURING, AND UPDATING POWER BI REPORTS

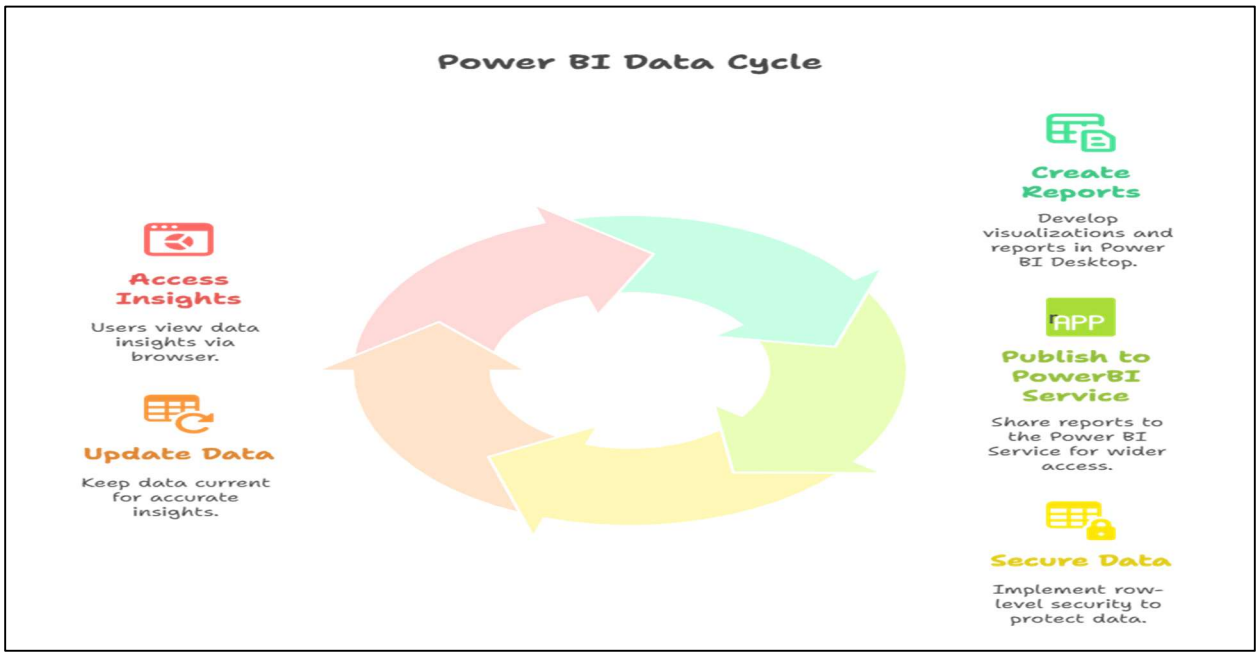
- Power BI was designed to take the liberation that Excel pioneered to the max, by allowing 'power' users to not only to create Power BI desktop reports, graphics and visualizations in Power BI App, but then link and publish these to the Power BI app.

- These in turn can be given row level security and have their underlying data kept up-to-date so that users can log in and see the relevant and appropriate data in a browser.

- **Power BI app (Browser) is used to provide updateable visualizations of an organization's business data; its graphs, plots, tables, reports and other types of visualization.**

- It is designed to make it easier and quicker to deploy summarizations of business data for that business to gain insights from.

Once your dashboard is deployed to the Power BI app, it can be viewed from anywhere, at any time.



The Power BI app also supports row-level security to secure the data access. In addition, you can, with periodic data refresh, update the report dynamically with the latest data.

To achieve all this functionality, your data or dashboard should be in the Power BI app, and you will need to use the Power BI Gateway too.

➢ **Topics**

- Create and publish a Power BI desktop report in Power BI App,
- Components and accounts are needed to link and publish Power BI desktop dashboard to the Power BI app.
- Refresh Power BI dashboard periodically and explain how to connect Power BI app and Power BI desktop.
- Finally, we will create user roles in Power BI desktop and assign roles to users in Power BI app to implement row level security.

COMPONENTS REQUIRED FOR POWER BI CLOUD SERVER

- **Power BI Gateway**: to act as a bridge between Power BI app and the data source.
- **Power BI App**: which is a Web-based visualization tool to view your data anytime, anywhere.
- **SQL Server Database:** to create an example database table. You can create this table using some other data source as well.
- **An organizational email account**: to sign in Power BI app and both Power BI desktop and gateway.

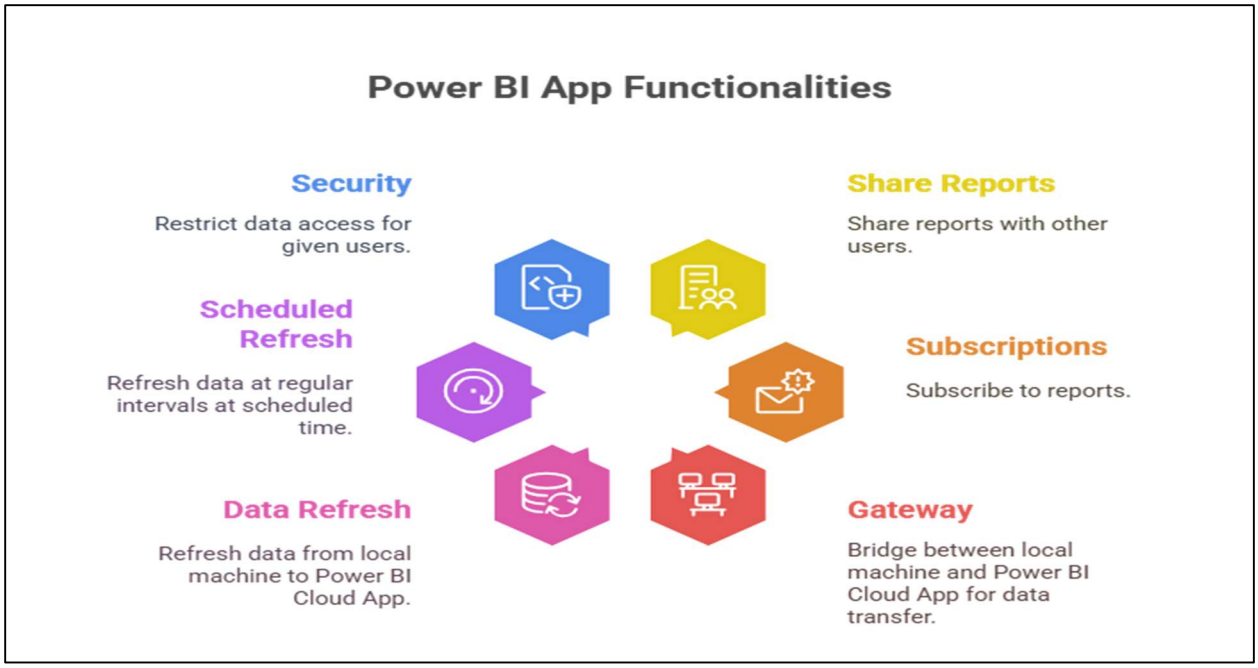
COMPONENTS OF POWER BI APP

Major Components	Description
WorkSpace	A workspace is similar to a projects or folder created in Power BI Apps where we can deploy the power bi reports.
DashBoard	A Dashboard is the main page of the Power BI Report.
Report	A report is a .pbix file which is created on Power BI Desktop and which can be deployed in Power BI Web App.
Datasets	A Dataset is the set of Source data which have been used to create the reports.

Export Options	Description
Analyze in Excel	Select Get Data option in home page and then click on excel workbook.
Export to Power Point	The Power BI Report will be exported into PPT with all the pages.
Export to PDF	The Power BI Report will be exported into PDF with all the pages.
Download PBIX	The Power BI Report will be downloaded as PBIX files which can used in desktop version of Power BI.
Embed in HTML	The Power BI Report will be converted into html version.

MAJOR FUNCTIONALITY OF POWER BI APPS

Components	Description
Share Reports	This functionality is used to share reports with other users.
Subscriptions	This functionality is used to subscribe reports.
Gateway	The Gateway is a bridge between local machine and Power BI Cloud App. Using the bridge we can transfer data between local machine and Power BI Cloud App.
Data Refresh	The Data Refresh is used to refresh the data from local machine to the Power BI Cloud App.
Scheduled Refresh	Scheduled refresh will refresh the data at regular intervals at scheduled time.
Security	Power BI can be used to restrict data access for given users



Note : Power BI App is a cloud-based platform that allows users to publish, view, and share Power BI Desktop reports and dashboards online. It enables organizations to access real-time business insights from anywhere through browsers, with data automatically refreshed via Power BI Gateway. The app supports Row-Level Security (RLS) to ensure that each user can view only the data relevant to their role. Additionally, it provides export options such as Excel, PowerPoint, PDF, and HTML, along with features like report sharing, subscriptions, and scheduled data refresh for seamless collaboration and up-to-date visualizations.

Power BI Desktop:

Using Power BI desktop, we can connect to different type of data sources and in one report we can connect to more than one data source , create relationship between data objects, massage the data using DAX query, perform real-time data analysis, integrate R script (For details about integrating R packages into SQL server, Please refer this link), and create rich visualizations. Power BI app provides the means of giving your colleagues access to these.

Power BI App:

Power BI App is a web-based application; a place to view your published application. Using Power BI app, you can share your application to many people. Power BI App uses HTML5 technology, so your application is compatible with mobile devices. In Power BI app, we can also edit the existing report or dashboard.

Example : Publishing A Power Bi Desktop Report In Power Bi App

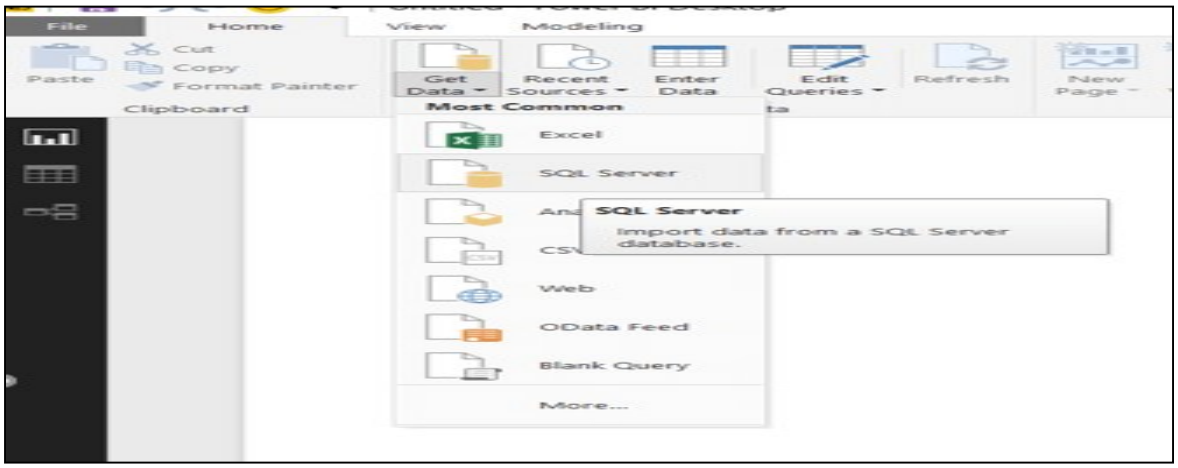
Here we will create a report in Power BI desktop and deploy it in Power BI Cloud App.

To create any chart, we need some data. We will run the following SQL script in a SQL database. It creates an employee table and adds few records in table.

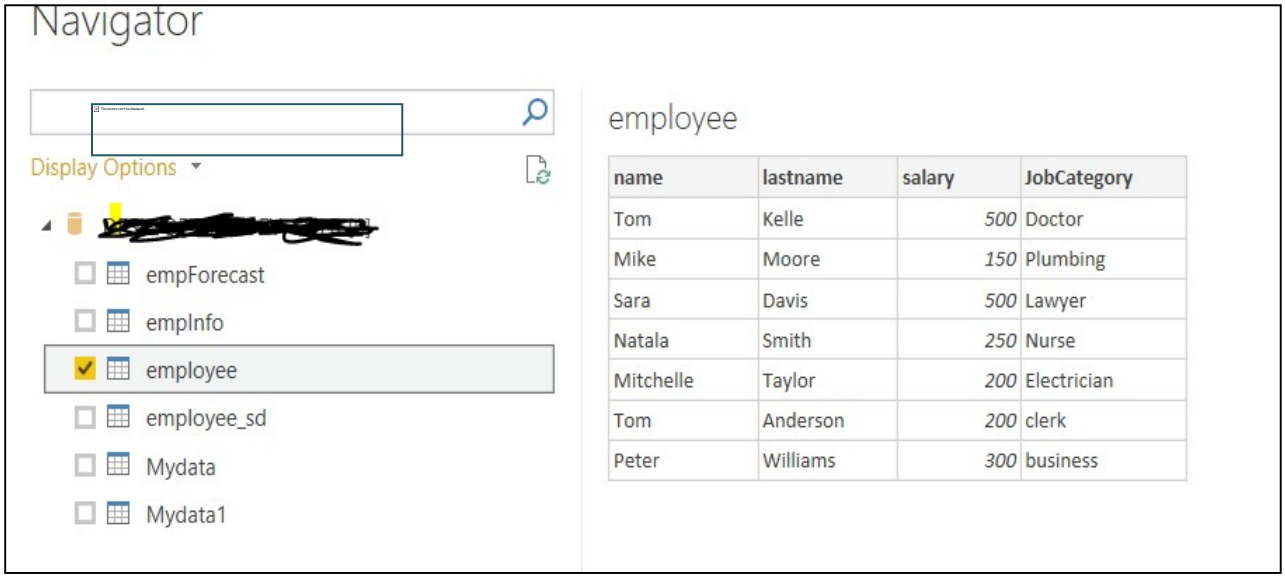
```
CREATE TABLE [dbo].[employee](
   [name] [varchar](50) NOT NULL,
   [lastname] [varchar](50) NOT NULL,
   [salary] [int] NULL,
   JobCategory [varchar](50) NOT NULL,
  )

  INSERT INTO [employee]([name],[lastname],[salary],JobCategory)VALUES('Tom','Kelle',500,'Doctor')
  INSERT INTO
[employee]([name],[lastname],[salary],JobCategory)VALUES('Mike','Moore',150,'Plumbing')
  INSERT INTO [employee]([name],[lastname],[salary],JobCategory)VALUES('Sara','Davis',500,'Lawyer')
  INSERT INTO [employee]([name],[lastname],[salary],JobCategory)VALUES('Natala','Smith',250,'Nurse')
  INSERT INTO
[employee]([name],[lastname],[salary],JobCategory)VALUES('Mitchelle','Taylor',200,'Roof')
  INSERT INTO
[employee]([name],[lastname],[salary],JobCategory)VALUES('Tom','Anderson',200,'clerk')
  INSERT INTO
[employee]([name],[lastname],[salary],JobCategory)VALUES('Peter','Williams',300,'business')
```

STEP 1: TO CREATE A REPORT, WE NEED TO BRING OUR DATA INTO POWER BI DESKTOP

- Open Power BI desktop and click on '*Get Data*' and select '*SQL server*' as the data source.

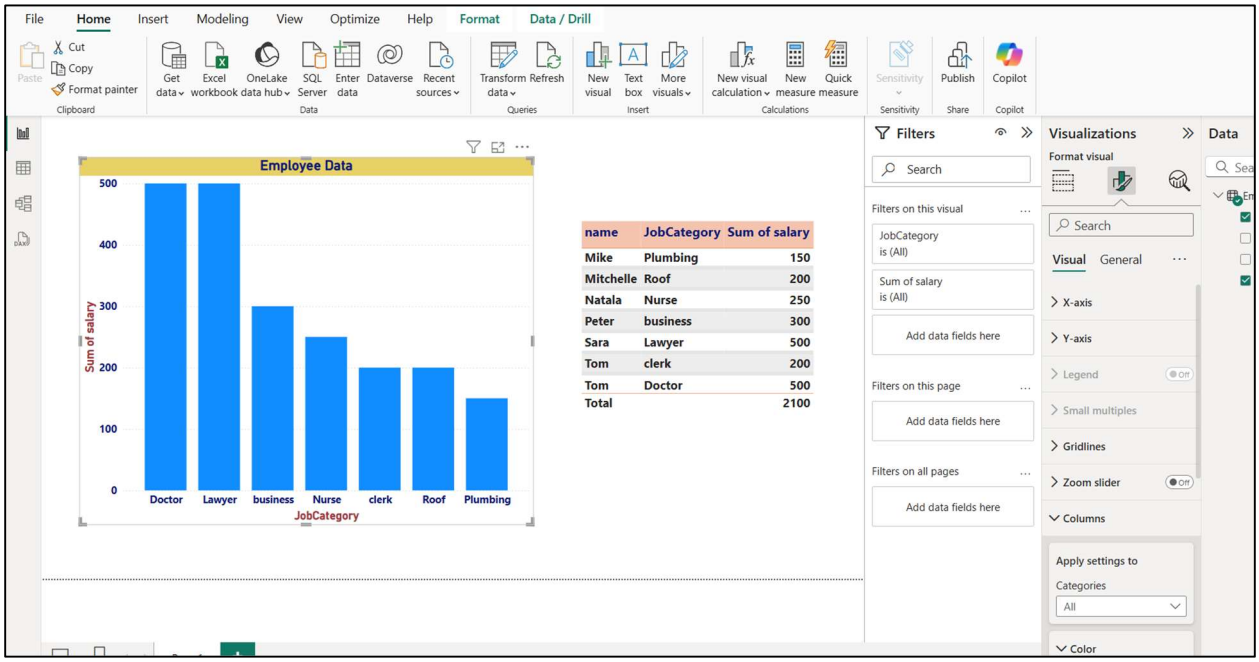


- Connect to the database in Power BI, and then select the **employee** table.



name	lastname	salary	JobCategory
Tom	Kelle	500	Doctor
Mike	Moore	150	Plumbing
Sara	Davis	500	Lawyer
Natala	Smith	250	Nurse
Mitchelle	Taylor	200	Electrician
Tom	Anderson	200	clerk
Peter	Williams	300	business

STEP 2: NOW WE CAN CREATE A POWER BI REPORT USING VARIOUS VISUALIZATION COMPONENTS.

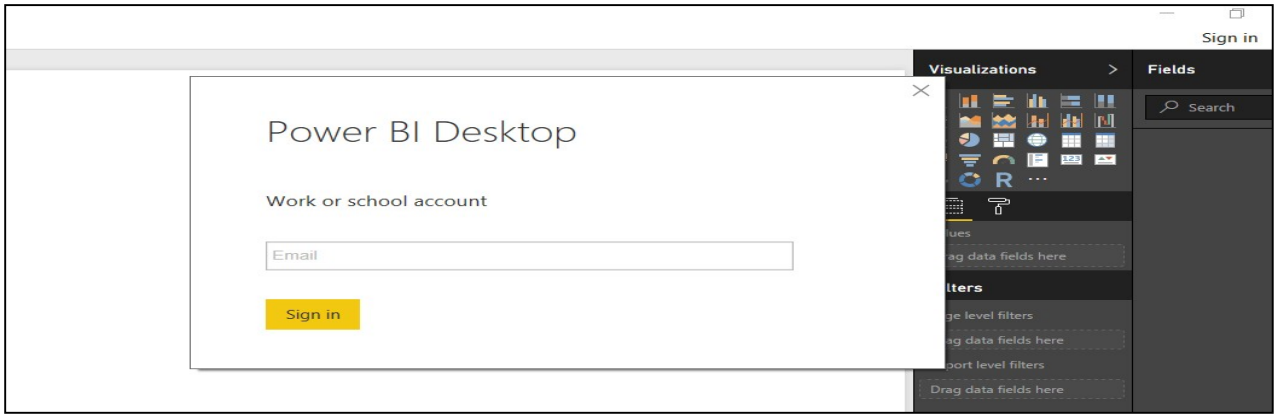


name	JobCategory	Sum of salary
Mike	Plumbing	150
Mitchelle	Roof	200
Natala	Nurse	250
Peter	business	300
Sara	Lawyer	500
Tom	clerk	200
Tom	Doctor	500
Total		2100

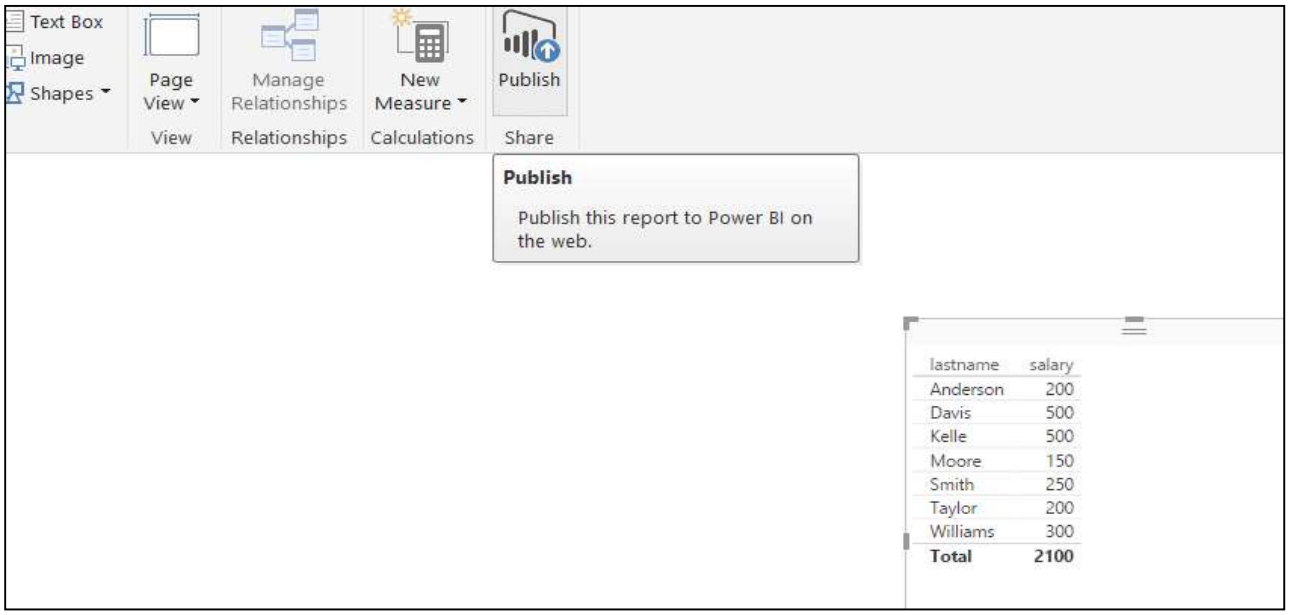
- Once we have Power BI desktop, Power BI app and the account set up, we then follow these next steps to publish dashboard to Power BI app.

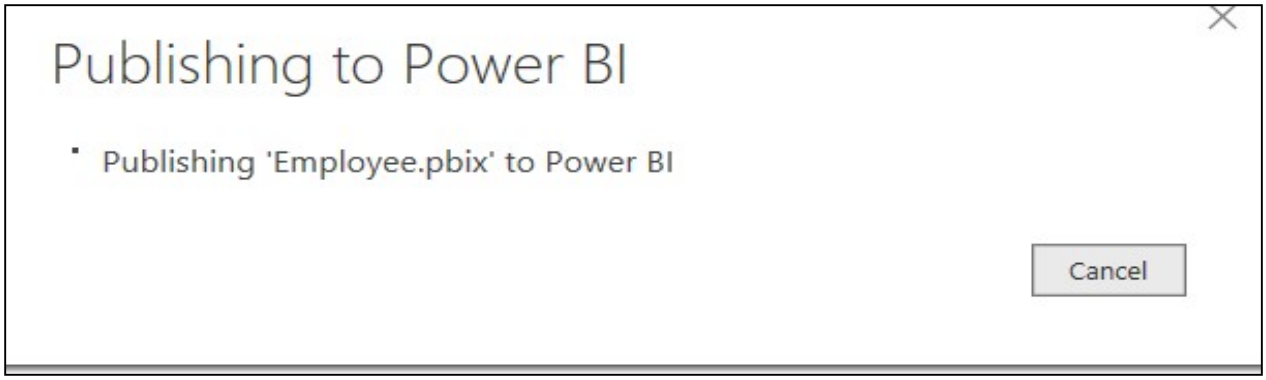
STEP 3: PUBLISH THE APP.

First, open Power BI desktop and sign in to Power BI desktop.

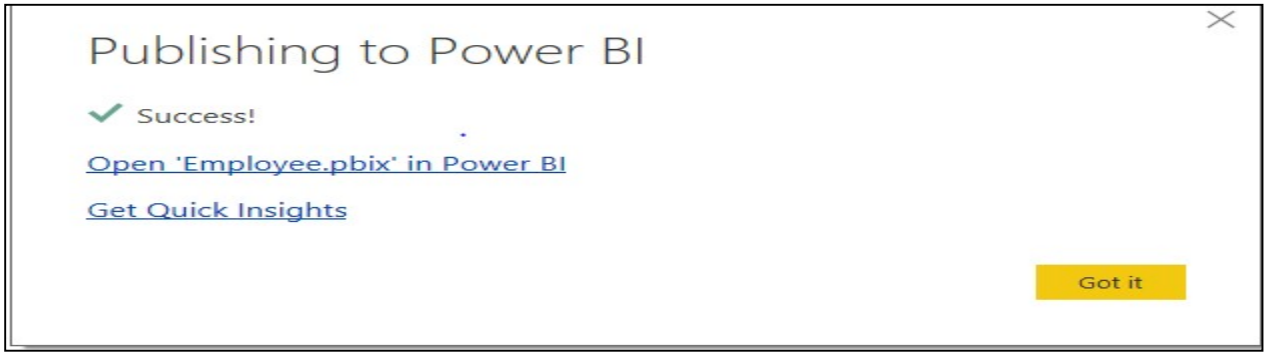


To sign in, use your org/school account in power BI desktop and in Power BI app. From '*home*' tab, click on '*publish*' button.





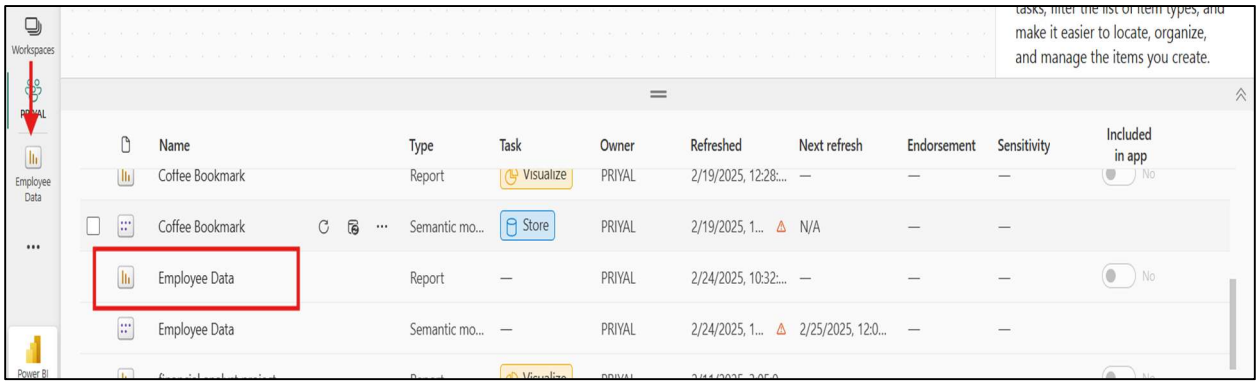
Now Power BI desktop reports that the dashboard is published.



STEP 4: VERIFYING THAT THE PUBLISHED REPORT HAS BEEN PROPERLY PUBLISHED

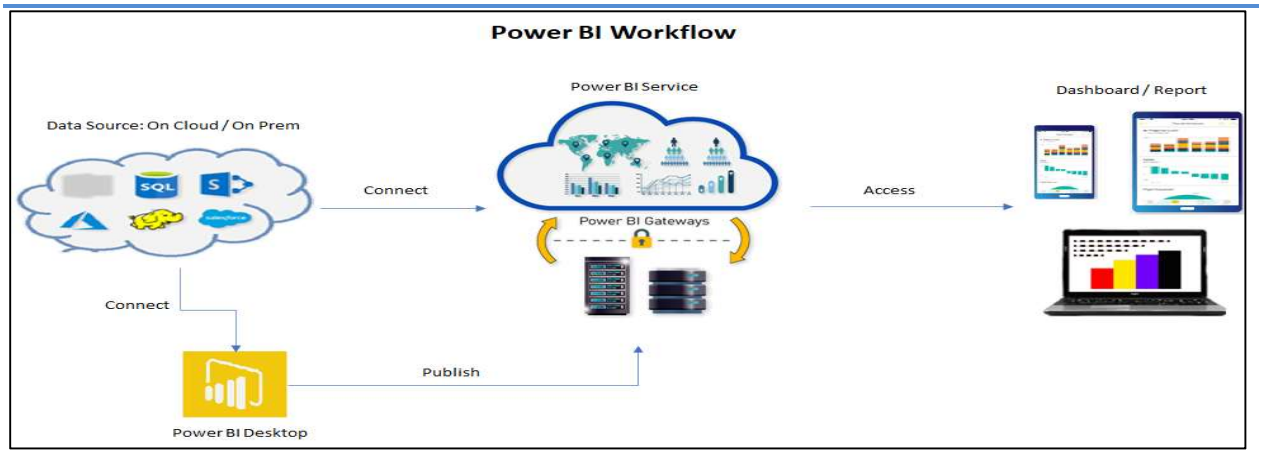
We now verify that all is working. Follow these steps to make sure that we can see our Power BI desktop report in the Power BI app.

- Open your browser and type app.Power BI.com; then login to your account. We will need to use the same account on Power BI desktop and Power BI app so that the document created In Power BI desktop is avilable in the Power BI app.
- In the left panel, you will see '*My workspace*'.



- Under '*My Workspace*', you will see a few options.
- *'Datasets'*: Our dataset name "*Employee*" will display under *Datasets* section.
- *'Reports'*: Our report name '*employee*' will show under *Reports* section. Open the Employee Report from File menu. You can print the report, and export the report to PowerPoint.
- *'Workbooks'*: If we deploy an Excel workbook to Power BI app, '*Excel file'* will show under the *'workbook'* section.
- '*Dashboards*': we can create a dashboard using one or more reports.

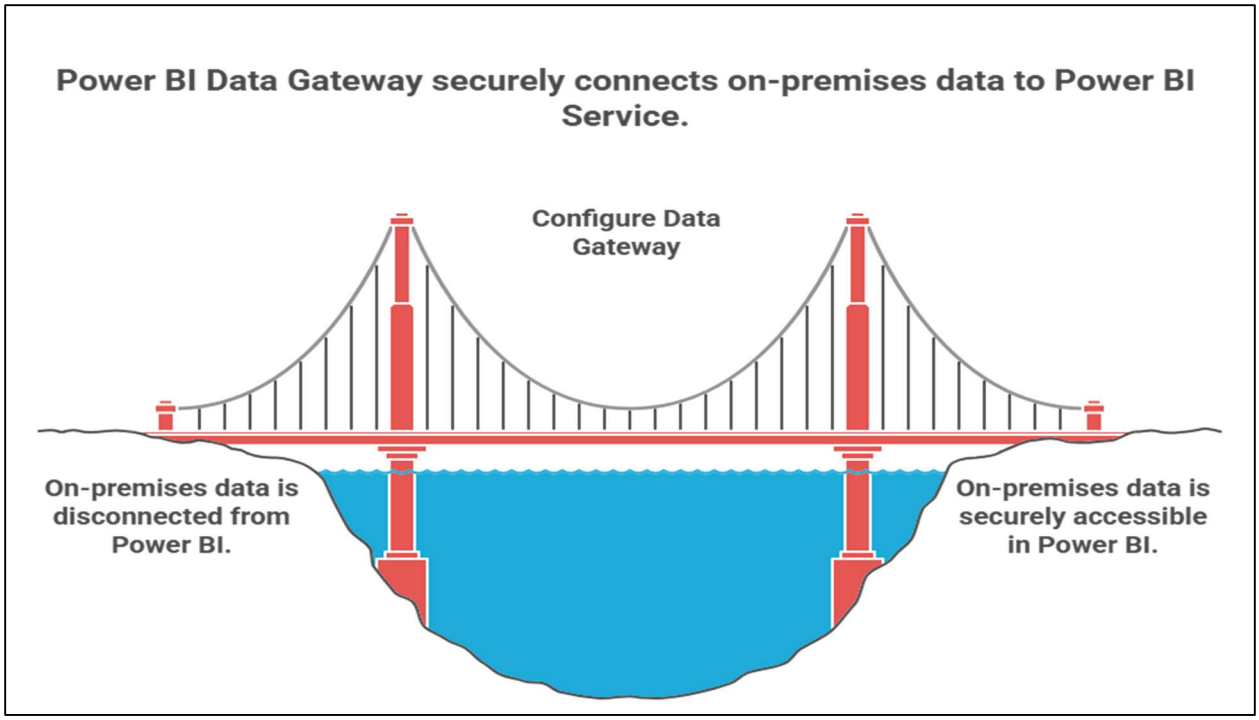
CONFIGURING A DATA GATEWAY IN POWER BI SERVICE



A **Data Gateway** in Power BI is used to securely connect **on-premises data sources** (such as SQL Server, MySQL, Oracle, or file-based data) to **Power BI Service**, enabling **data refreshes** and **live queries** without moving the data to the cloud. It acts as a **bridge** between on-premises data and Microsoft cloud services like Power BI, Power Automate, Power Apps, and Azure. The gateway ensures **secure data transfer** by encrypting data and preventing direct exposure of internal databases to the internet. It allows **scheduled refreshes** for imported datasets and supports **DirectQuery** and **Live Connection** for real-time data access. This is essential for organizations needing **hybrid data solutions** while maintaining data security and compliance.

Configuring a **Data Gateway** in Power BI Service is an essential step to keep your on-premises data sources refreshed and connected to Power BI. Here's a step-by-step guide on how to set up and configure the gateway:

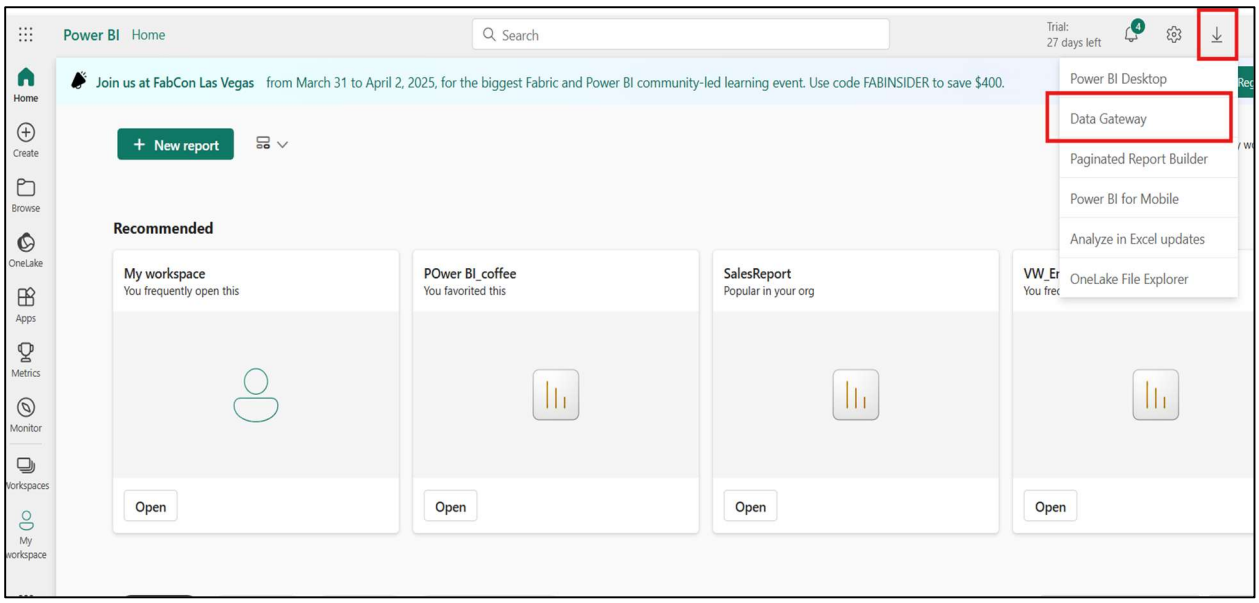
Note : A Data Gateway in Power BI securely connects on-premises data sources to the Power BI Service, enabling data refreshes and live queries without moving data to the cloud. It encrypts data transfers, ensuring security and compliance. The gateway supports both scheduled refresh and real-time connections for hybrid data solutions.

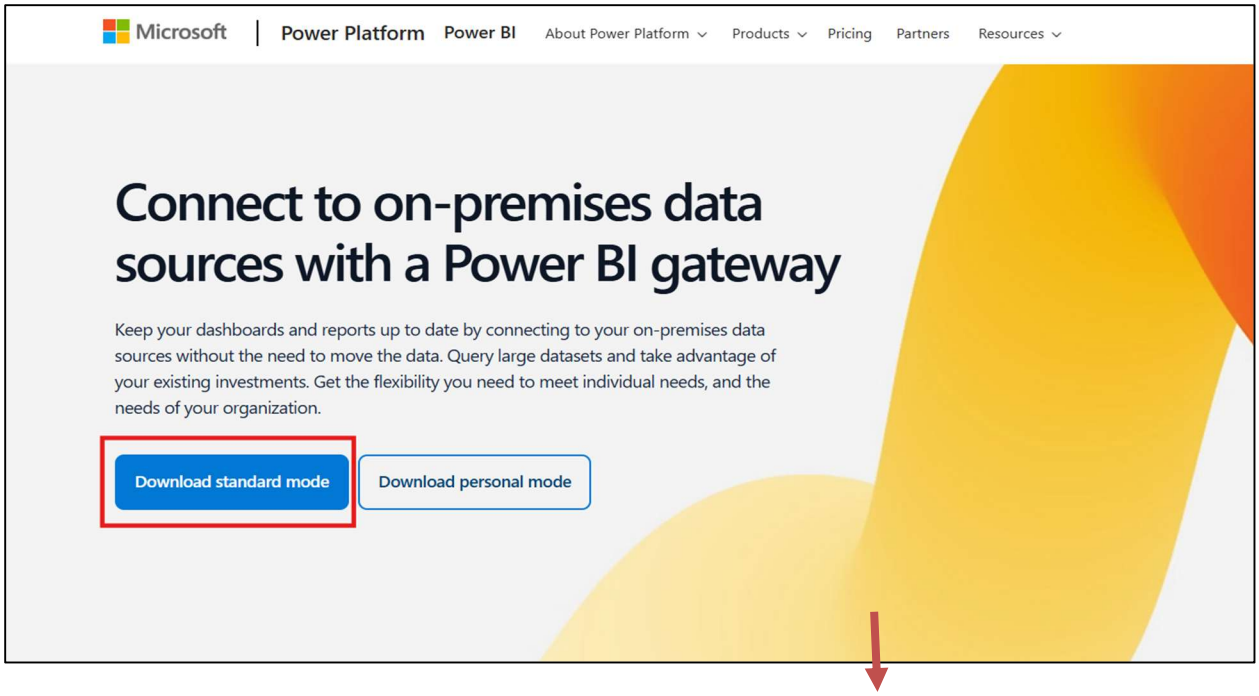


STEP 1: INSTALL THE ON-PREMISES DATA GATEWAY

Download the Gateway:

- Go to the https://www.microsoft.com/en-us/power-platform/products/powerbi/gateway
- Download the **On-Premises Data Gateway**.

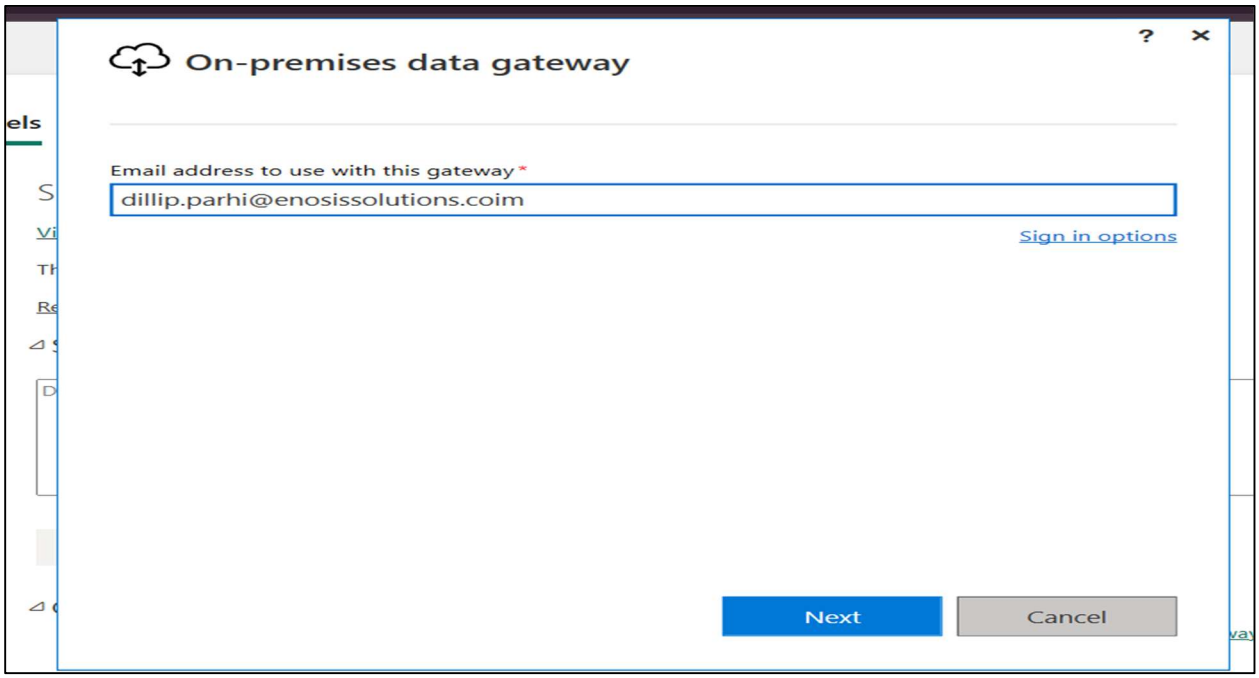


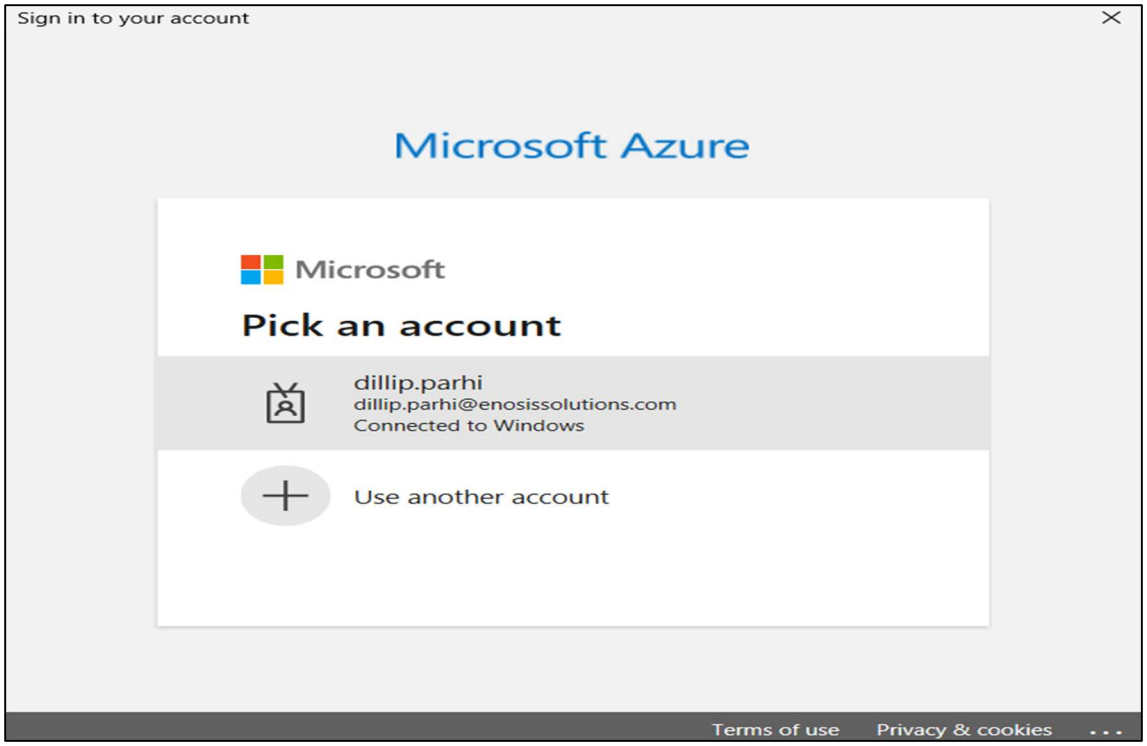


Install the Gateway:

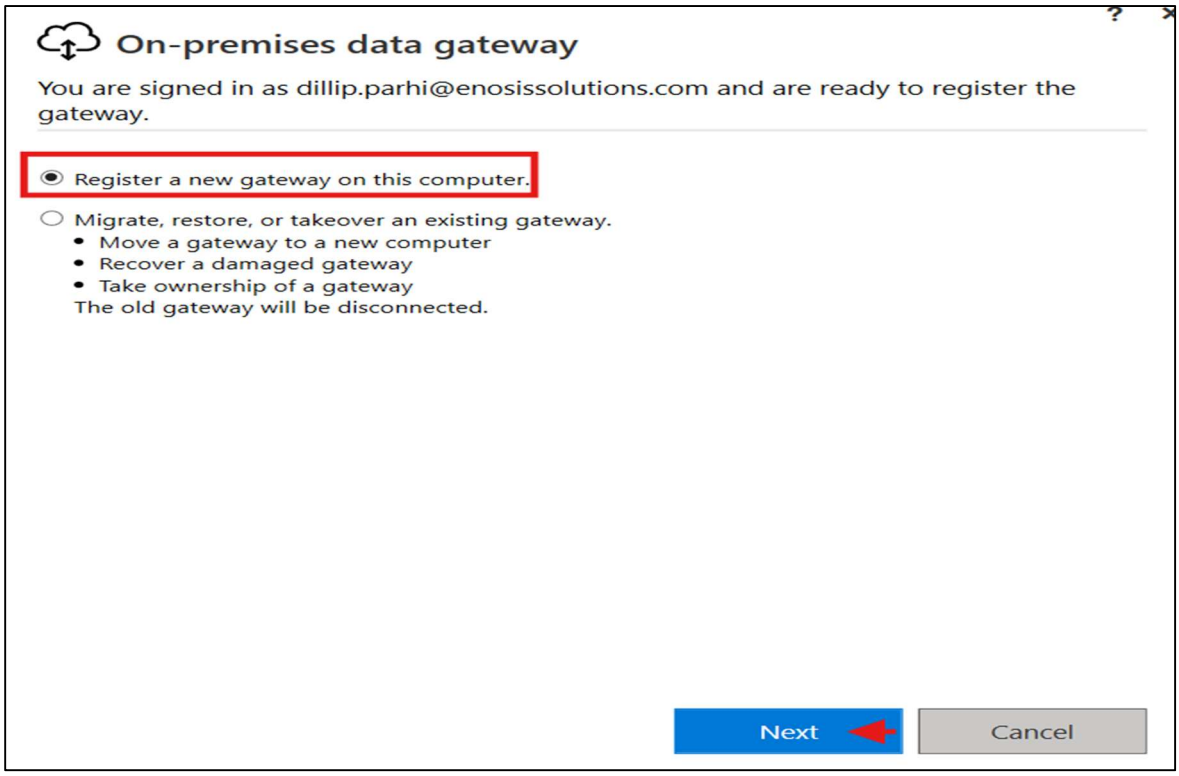
Step 1: Run the downloaded installer.

- Need to login in the ON-premises data gateway

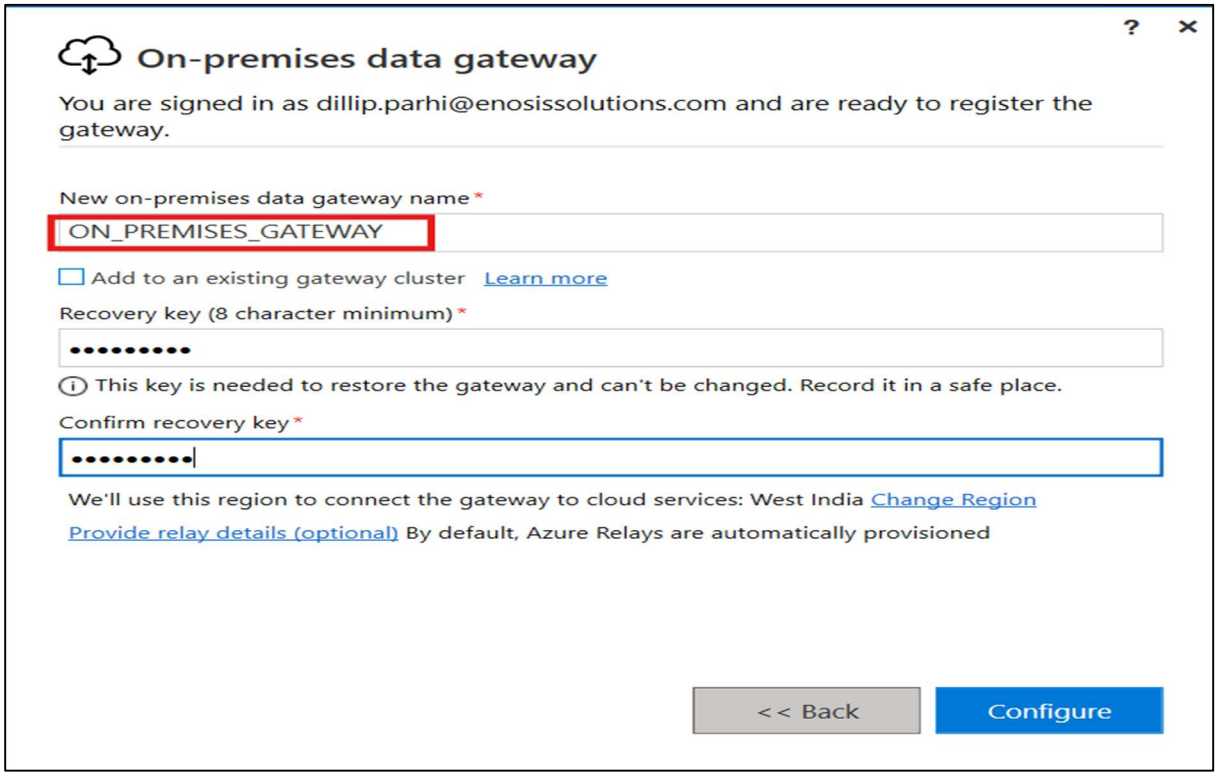




Step 2: Register the Gateway



Step 3: Now for registration create the New On-premises Data Gateway Name and set the Recovery key to it.



Once installed, launch the **On-Premises Data Gateway** application.

- **Sign in to Power BI:**

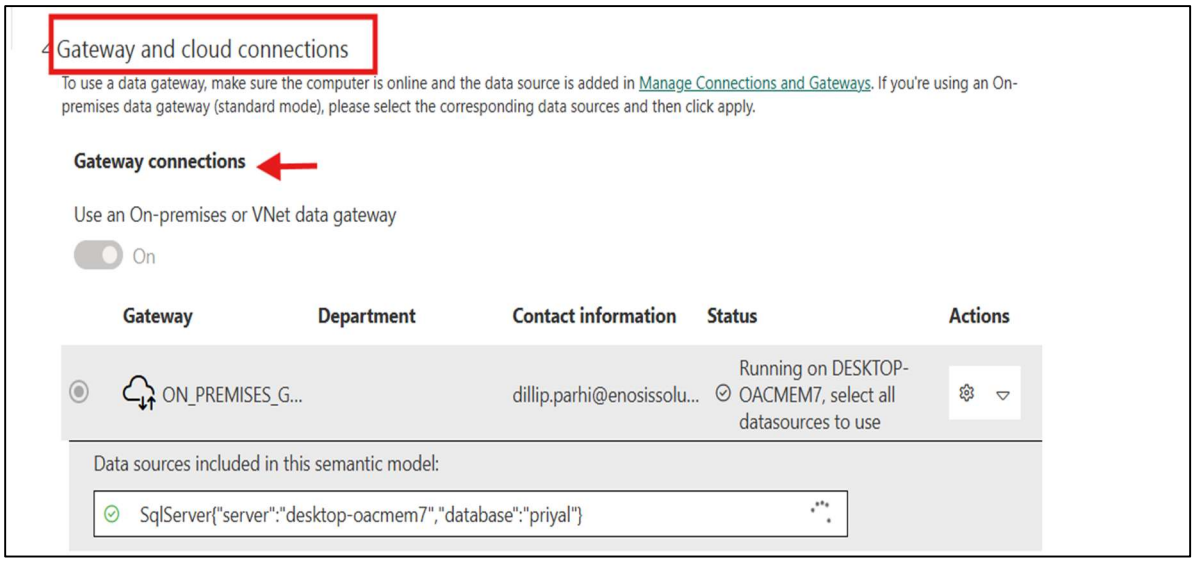
  1.Use your organizational Power BI account to sign in.

  2.After sign-in, you'll be asked to either create or register a gateway recovery key. Keep this key secure, as it's needed if you want to manage or recover the gateway later.

STEP 2: REGISTER THE GATEWAY IN POWER BI SERVICE

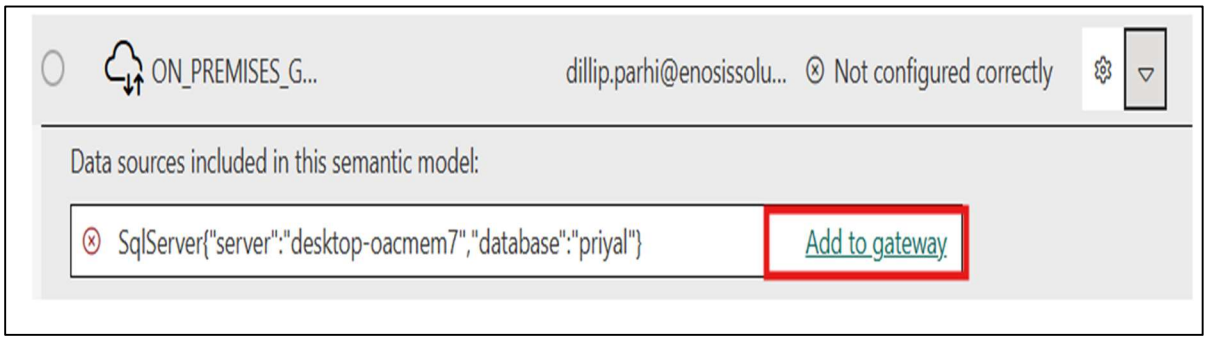
Configure the Gateway:

1. After installation, the gateway will automatically register with your Power BI account.
2. You can go to the **Power BI Service** (https://app.powerbi.com), and under the settings, you'll see the **Gateways** option in the **Admin Portal**.



Add the Gateway to the Power BI Service:

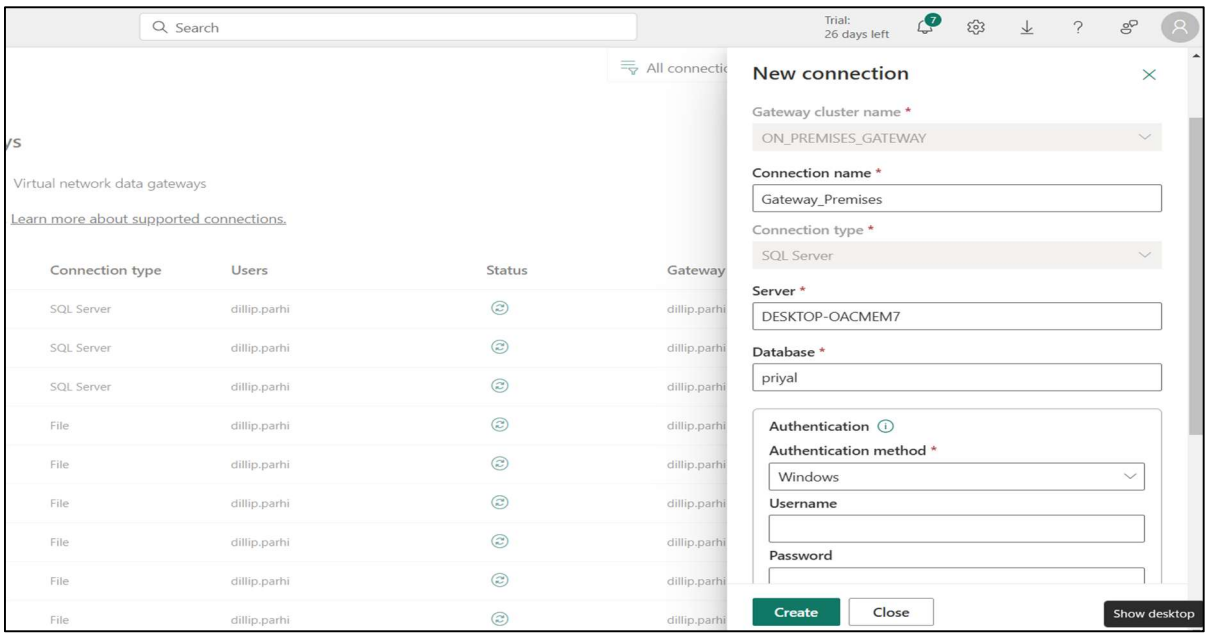
1. Navigate to **Settings > Manage Gateways**.
2. Here, you can see the newly registered gateway.
3. Click on **Add Data Source** to connect your on-premises data sources to the gateway.



STEP 3: CONFIGURE DATA SOURCES

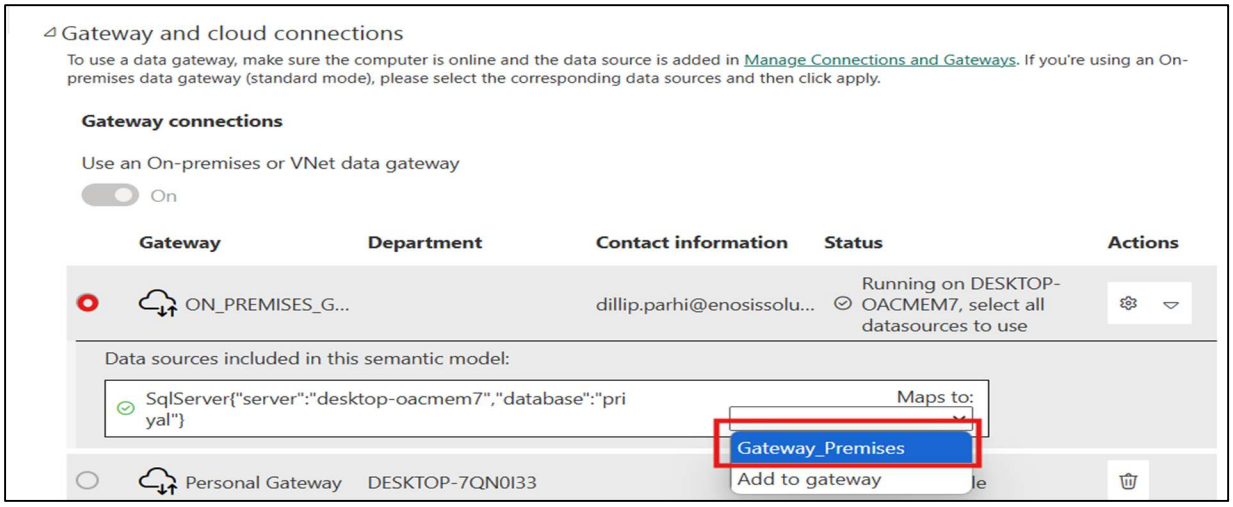
Configure A Data Source:

1. After adding the gateway, you need to connect your on-premises data sources (e.g., SQL Server, Oracle, etc.).
2. In the **Manage Gateways** section, click on your gateway and then click **Add Data Source**.
3. Choose the type of data source you are connecting to (e.g., SQL Server).
4. Enter the connection details, such as:
5. **Data Source Name**: A descriptive name for the data source.
6. **Connection Information**: Server name, database name, authentication type, and credentials.
7. You can also configure additional settings depending on the data source type.



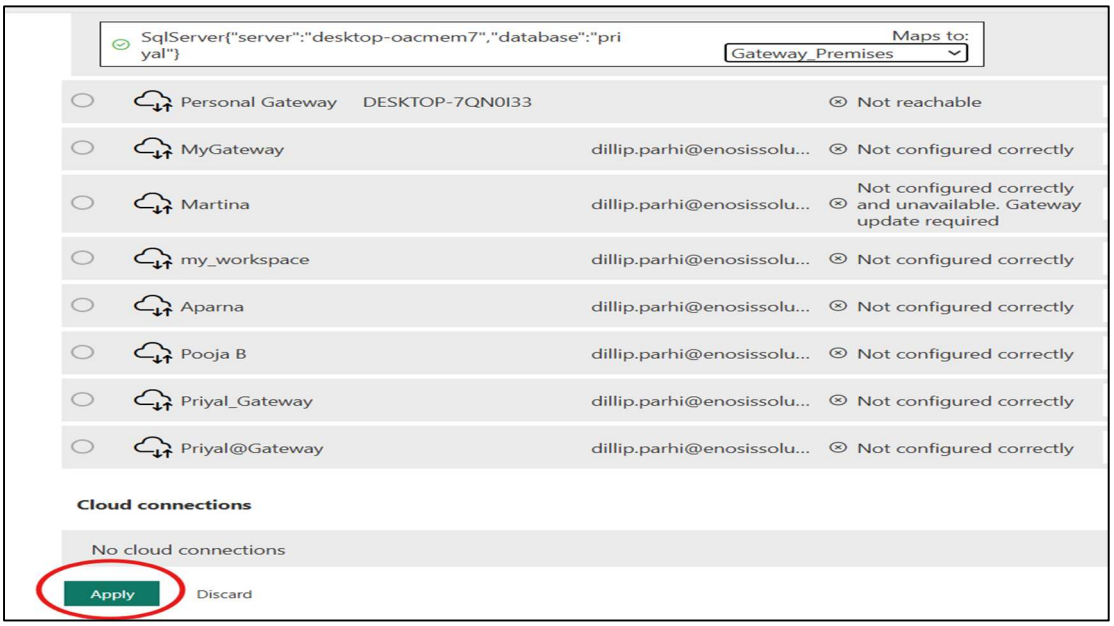
Map The Connection Name In The Gateway Created :

- mapping the connection name in the **Gateway** involves configuring the **data source mapping** to ensure that Power BI Service correctly connects to your on-premises data through the **On-premises Data Gateway**.



Click On Apply:

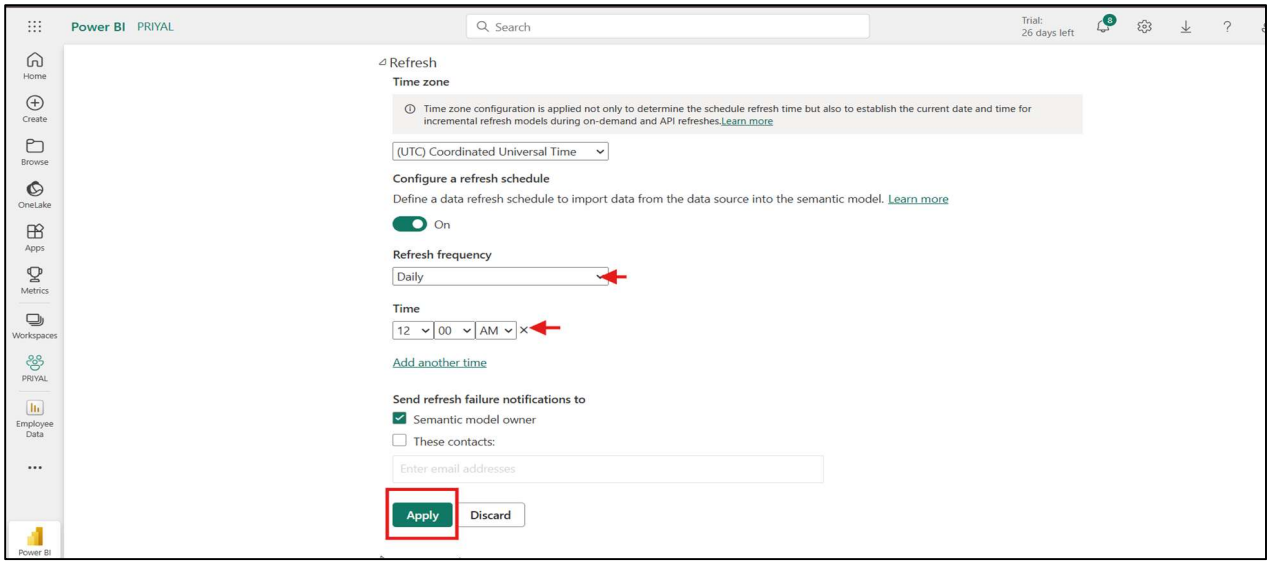
As u click on apply , the On_PREMISES_GATEWAY will be Configured with Power Bi Services.



STEP 4: USE THE GATEWAY IN POWER BI REPORTS

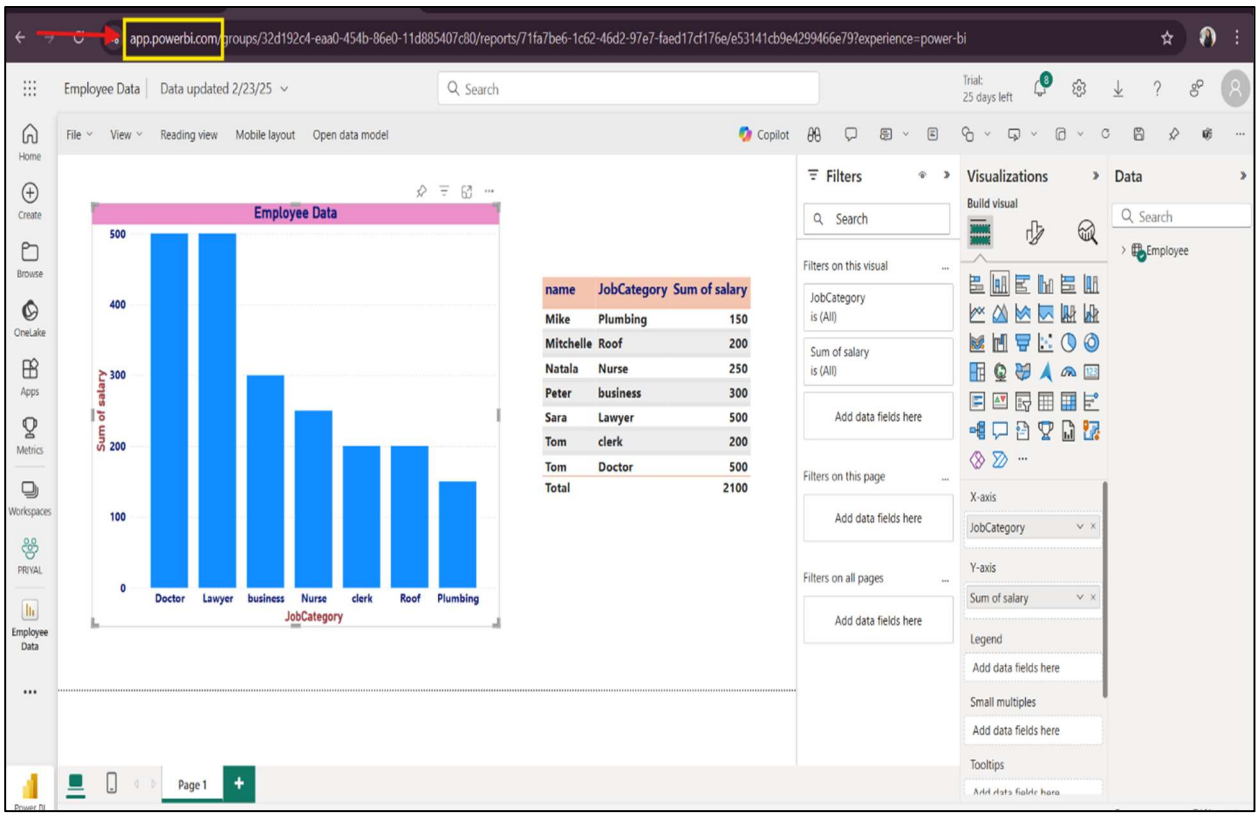
Configure Scheduled Refresh:

1. In Power BI Service, go to your dataset and click on **Schedule Refresh**.
2. Select the **Gateway** that you set up in the previous steps.
3. Set up the refresh schedule (daily, weekly, etc.) and configure the credentials if needed.

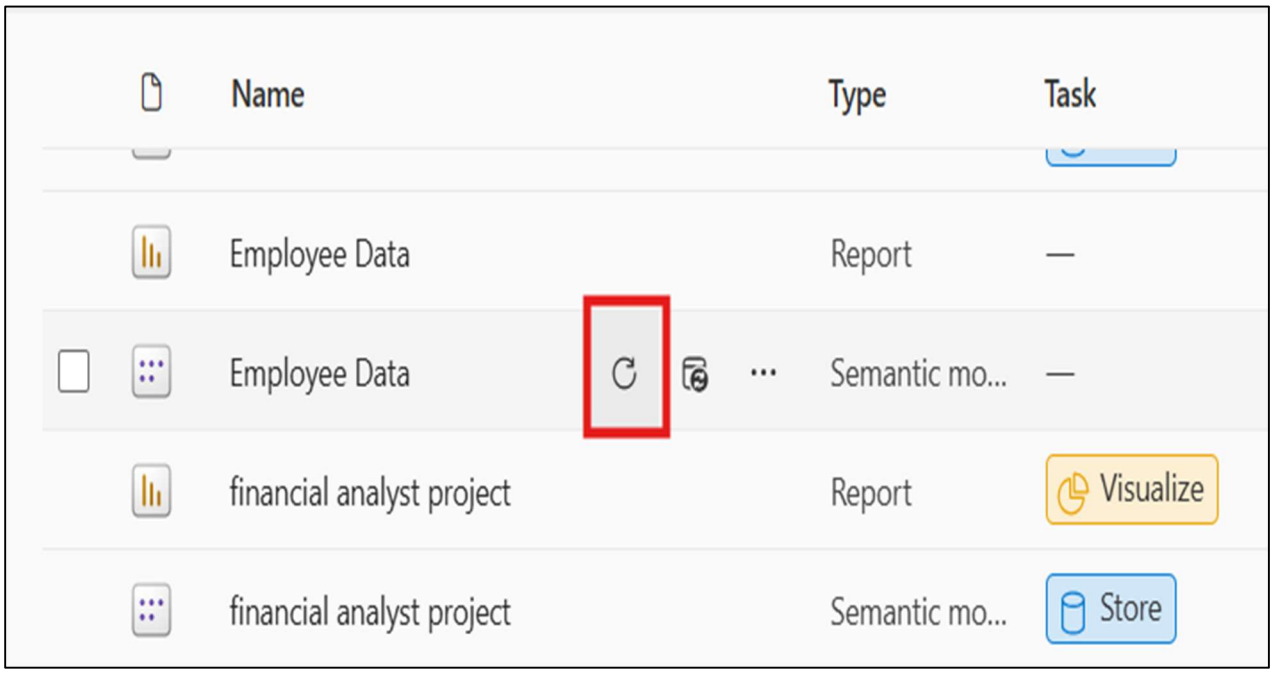


Publish Reports:

- Once the data source is configured and the gateway is linked, you can publish reports & dashboards in Power BI Service that will automatically use the gateway for data refresh.



Data Refresh:



STEP 5: MONITOR AND MANAGE THE GATEWAY

Monitor Gateway Performance:

1. You can view the performance, health, and activity of your gateway in the **Admin Portal**.
2. This allows you to track any issues, such as failed data refreshes or performance bottlenecks.

Manage Users and Permissions:

1. If you're using the **Standard mode**, you can add additional users to share the gateway for collaborative access.
2. Go to the **Manage Gateways** section and assign users to the gateway to enable shared access.

Note : The On-Premises Data Gateway in Power BI connects local data sources securely to the Power BI Service for real-time and scheduled refreshes. It is installed, registered, and configured in Power BI Service to link on-premises databases like SQL Server. Once set up, it enables publishing and auto-refresh of reports using live data. Administrators can monitor gateway performance, manage users, and ensure secure, continuous data access.

QUESTIONS

1.What is Deployment in Power BI and why is it important?
2.Explain the role of Power BI Gateway in connecting on-premises data with cloud services.
3.What is a Workspace in Power BI Service and how does it support collaboration?
4.How do Data Refresh and Subscriptions help organizations maintain updated reports in Power BI?
5.What is Row-Level Security (RLS) in Power BI and how does it protect sensitive data?

SUMMARY

Deployment in Power BI involves publishing reports from Power BI Desktop to the Power BI Service, enabling online access, sharing, and collaboration. Key components include the **Power BI Gateway**, which securely connects on-premises data to the cloud; **Power BI Apps**, which provide a web-based platform for viewing and distributing reports; and **Workspaces**, collaborative areas for managing dashboards, datasets, and reports. Major functionalities of deployment include **sharing reports** with stakeholders, setting up **subscriptions** to automate email delivery, performing **data refreshes** to keep reports current, and implementing **Row-Level Security (RLS)** to control data access based on user roles. Configuring the Data Gateway ensures secure, encrypted connections for hybrid environments, allowing both on-premises and cloud data to be seamlessly integrated. Once deployed, Power BI reports are accessible from anywhere, supporting real-time insights and enabling organizations to make timely, data-driven decisions.

Deployment in Power BI involves publishing reports from Power BI Desktop to Power BI Service for online access and collaboration.

Key components:

- Power BI Gateway – Bridges on-premises data with the cloud.
- Power BI App – Web-based platform for report viewing and sharing.
- Workspace – A collaborative space for managing dashboards, datasets, and reports.

Major functionalities:

- Share Reports – Distribute insights with stakeholders.
- Subscriptions – Automate email report delivery.
- Data Refresh – Keep reports up to date with real-time or scheduled refreshes.
- Row-Level Security (RLS) – Restrict data visibility based on user roles.