****

**RANI CHENNAMMA UNIVERSITY, BELAGAVI**

**INTERNSHIP REPORT SUBMITTED IN PARTIAL FULFILMENT OF**

**THE REQUIREMENTS OF BACHELOR OF COMPUTER APPLICATION, SCIENCE AND COMMERCE DEGREE COLLEGE, NIDASOSHI.**

**Submitted by**

**Mr. Fariyaz M Patil(U15CK21S0012)**

**Under the Guidance of**

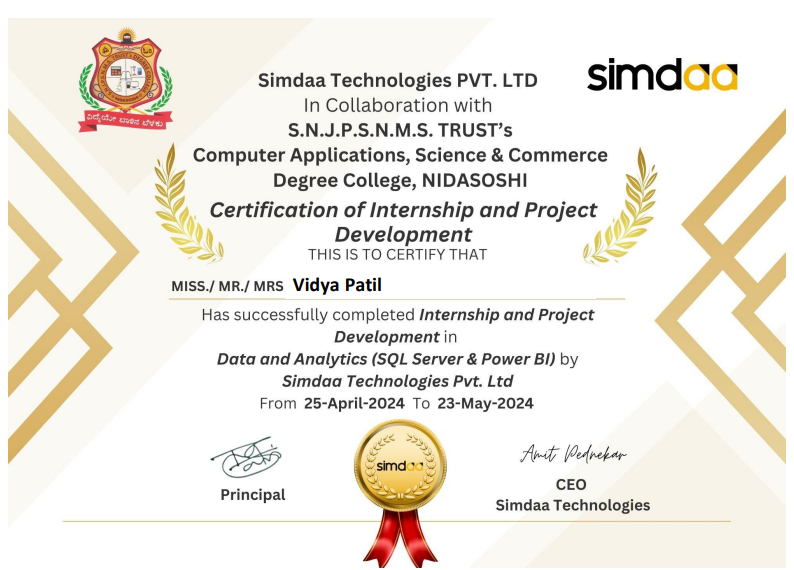
**Prof. Shilpa K Patil**

****

**S.N.J.P.S.N.M.S. Trust’s**

**BACHELOR COMPUTER APPLICATION, SCIENCE AND COMMERCE DEGREE COLLEGE NIDASOSHI.**

**2023-2024**



**Internship Report**

**For On Job Training**

**at**

**SJPN College, Nidasoshi**

**SQL SERVER**

**Date- 26 Apr 2024**

# Data & Database concepts

Data is collection of different types of information organized in structured way

Data is about the Fact with associated context

It can be in the form of numbers, words, images, or any other format that can be stored, retrieved, and used by a computer or other electronic device.

Data can be raw and unprocessed, or it can be organized, cleaned, and transformed into a more meaningful format.

Data can be stored in a variety of formats, such as in a database, spreadsheet, or text file

# • Consider an Example

* Customer name is Sreya Patil.Birthdate is 1-06-1997, located in Kolhapur and working in Simdaa Technology.
* Same informartion I want to store for above 10000 customers, then it is lengthy – So instead of repeating same we arrange this in structured manner

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| **Customer Name** | **Birth Date** | **Location** | **Company** |
| Sreya Patil | 1-06-1997 | Kolhapur | Simdaa Technology. |

# • Types of Data

1. Qualitative data

-Descriptive data

1. Quantitative Data -Numerical data a) Discrete data

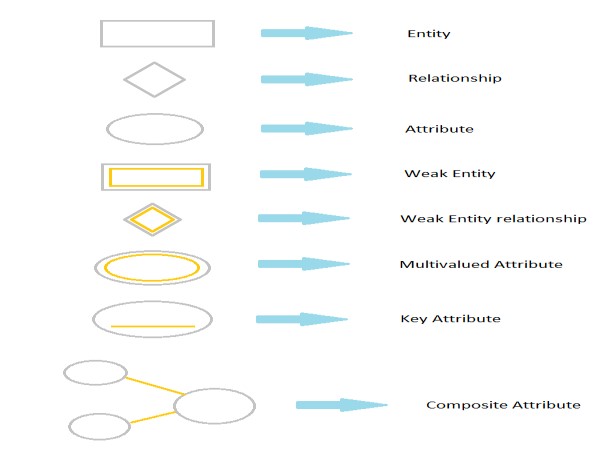
b) Continuous data

# Entity & Entity Relationship

In SQL, an entity is a table that stores data. For example, a customer entity might store customer information, such as their name, address, and phone number

The Entity Relational Model is a model for identifying entities to be represented in the database and representation of how those entities are related. The ER data model specifies enterprise schema that represents the overall logical structure of a database graphically.

# Symbols used in ER model



# Example of ER model

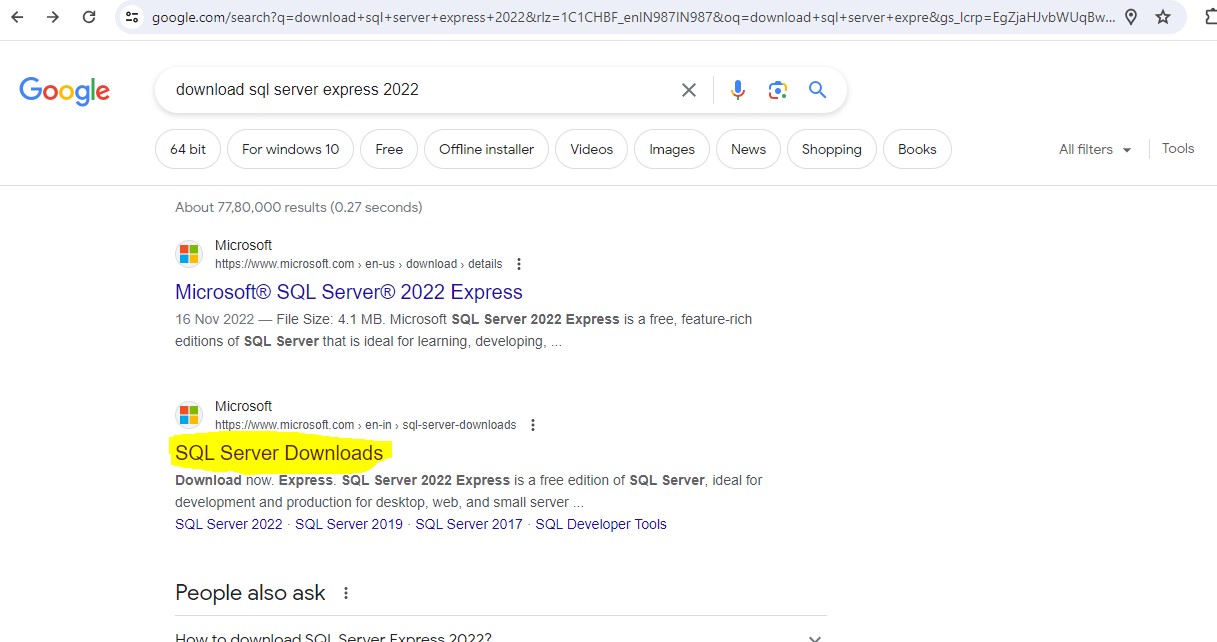
Customer & Account are the Entity

Cust Name,Address, CustID, Acc\_No, Account\_type, Balance are the attributes of the Entity

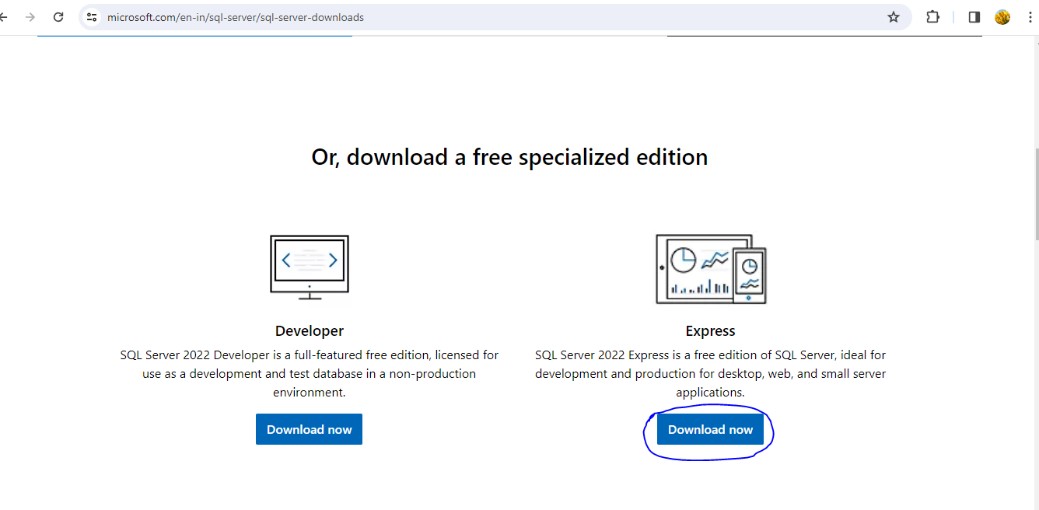
**Date : 29 Apr 2024**

**SQL Server Installation**

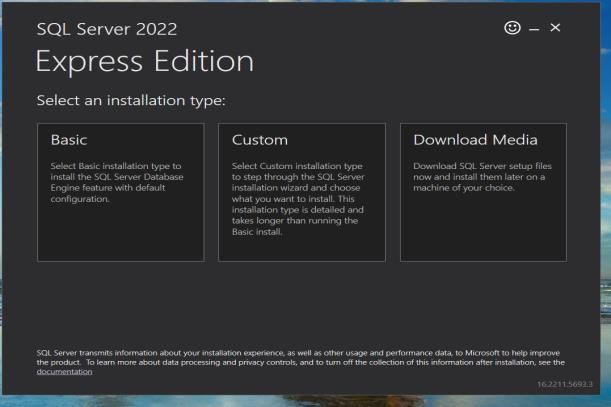
# Download SQL server instance from Google Chrome



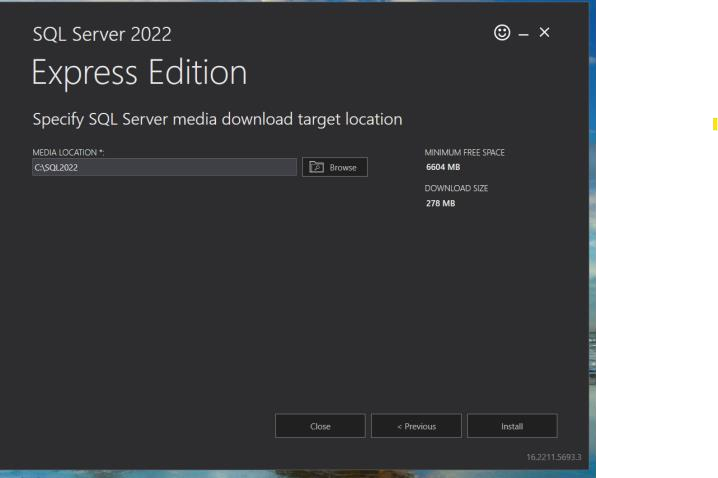
# Download SQL Server Express Edition



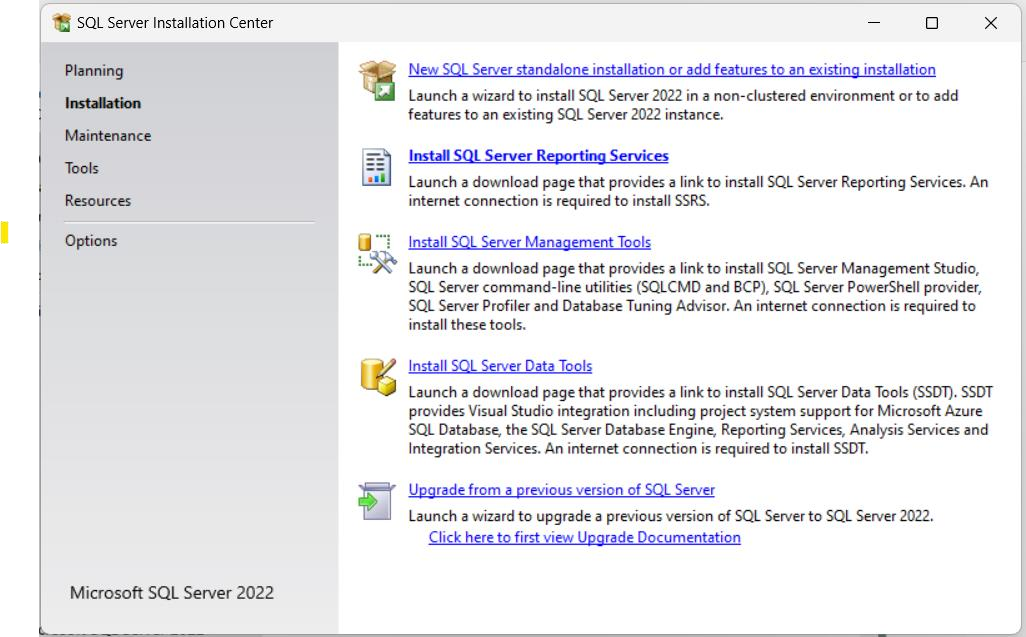
# It will ask for Installation Type. Click on Custom



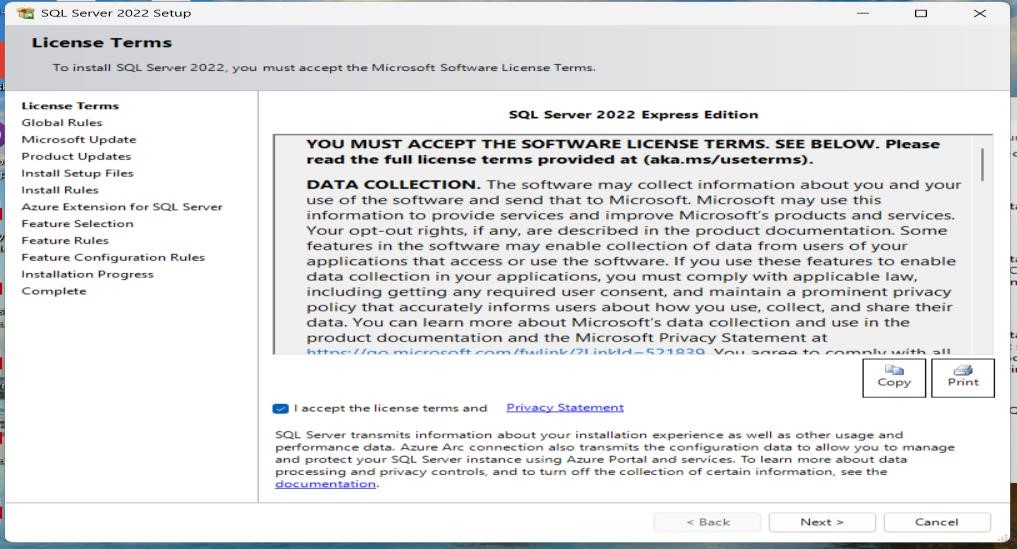
# Select the path for download and click on Install button



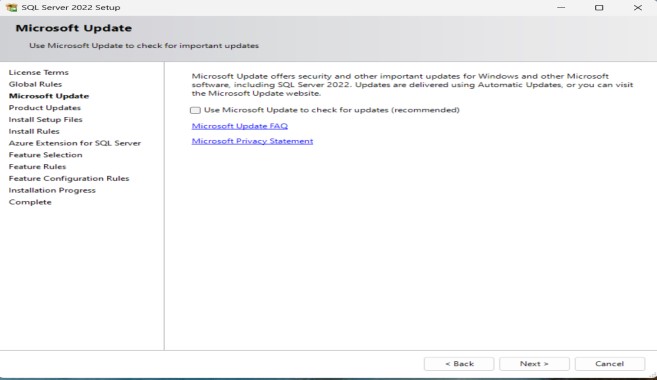
# Select New SQL Server standalone installation or add features to an existing installation



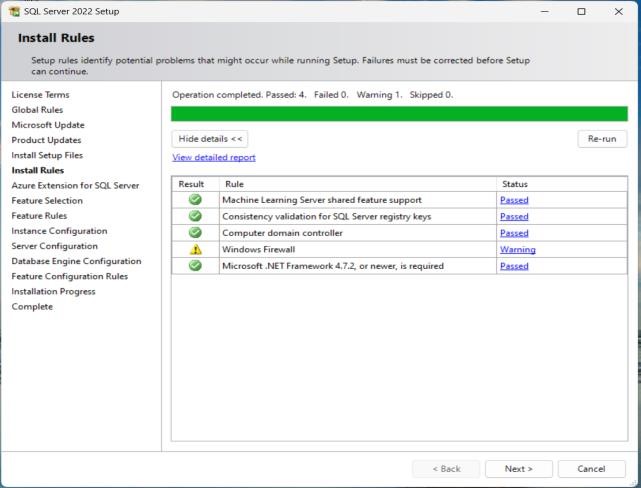
# Accept the terms and condition by clicking on I accept the licence terms and condition check box and click on Next button



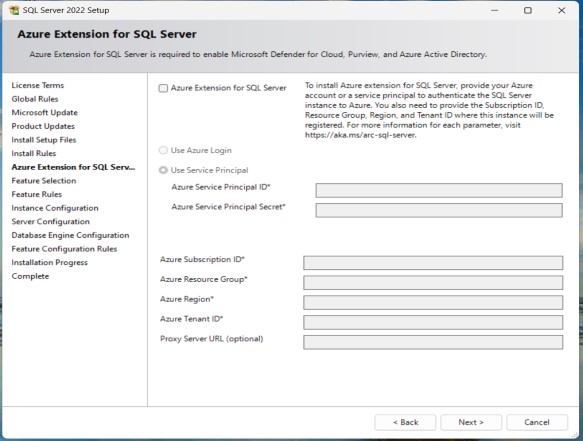
# Click on Next button



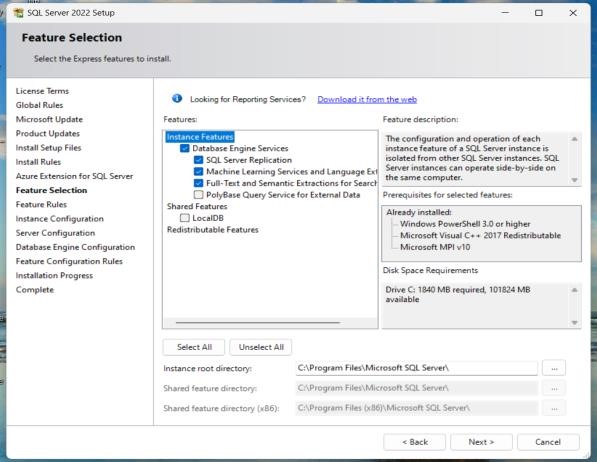
# Click on Next button



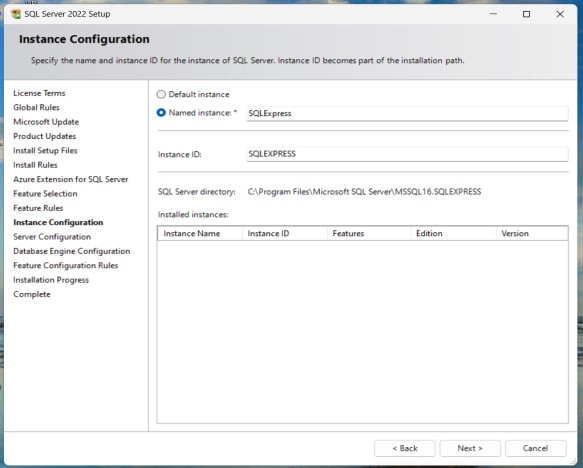
# Unselect Azure Extension for SQL Server check box and click on Next button



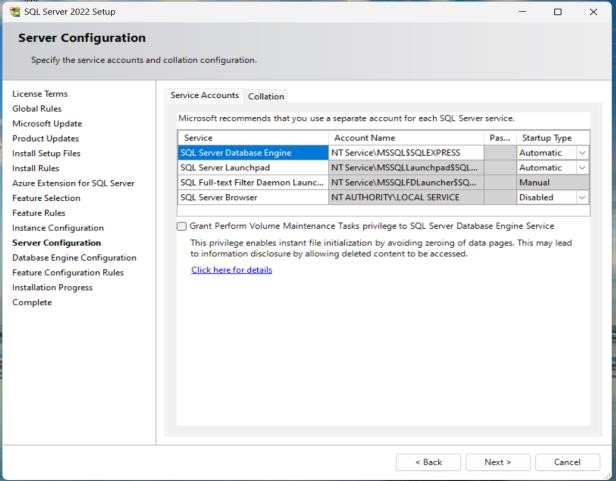
# Click on Next Button



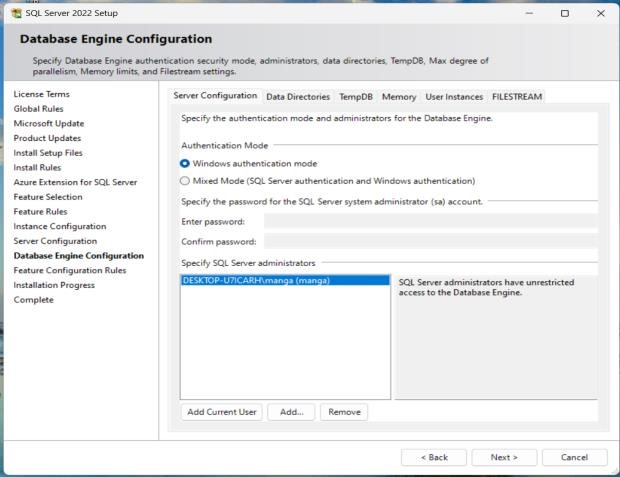
# Set the Named instance and click on Next button



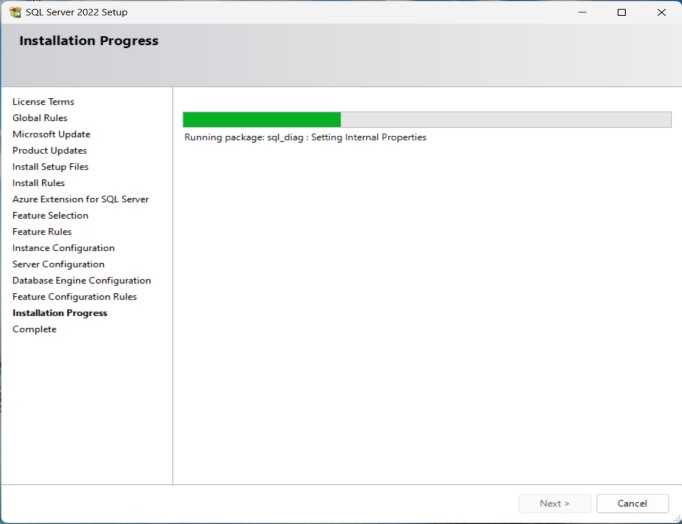
# Click on Next button



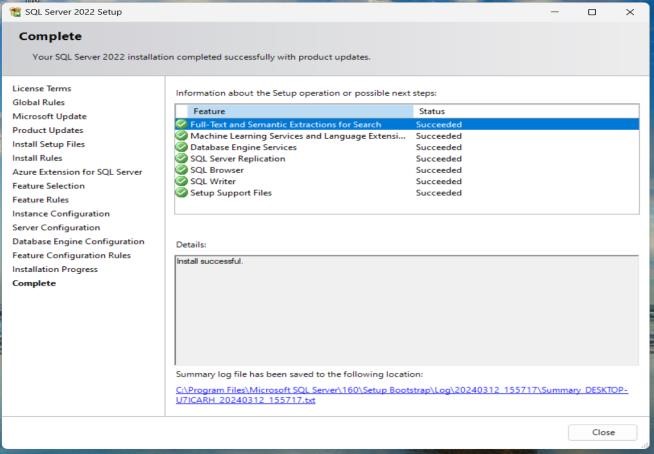
Select Windows Authentication mode and server



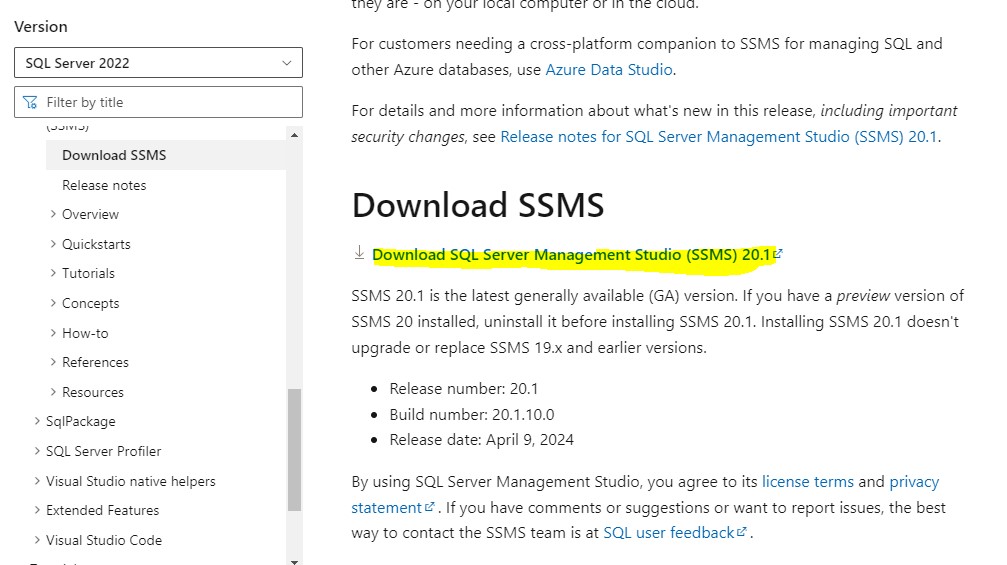
# Installation in inprogress



# After the installation process click on Close button



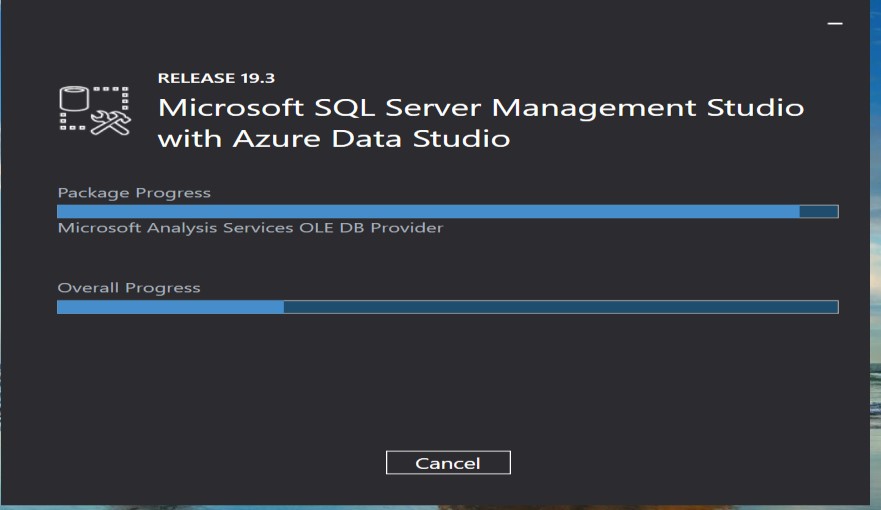
# Download SSMS 20.13



# Set the path where you want to install SSMS and click on INSTALL button



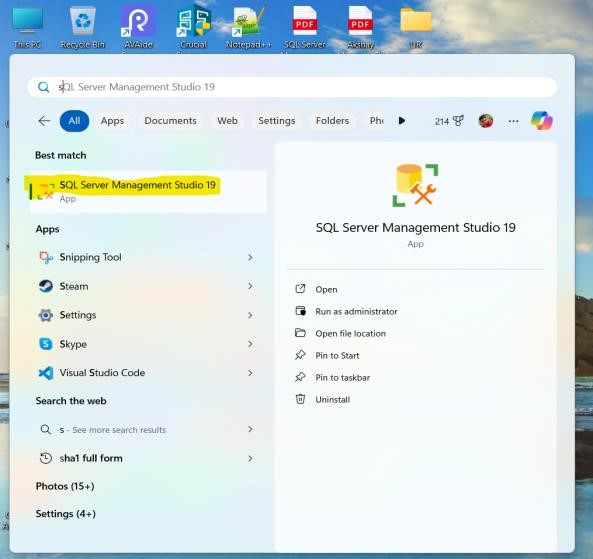
# Installing SSMS



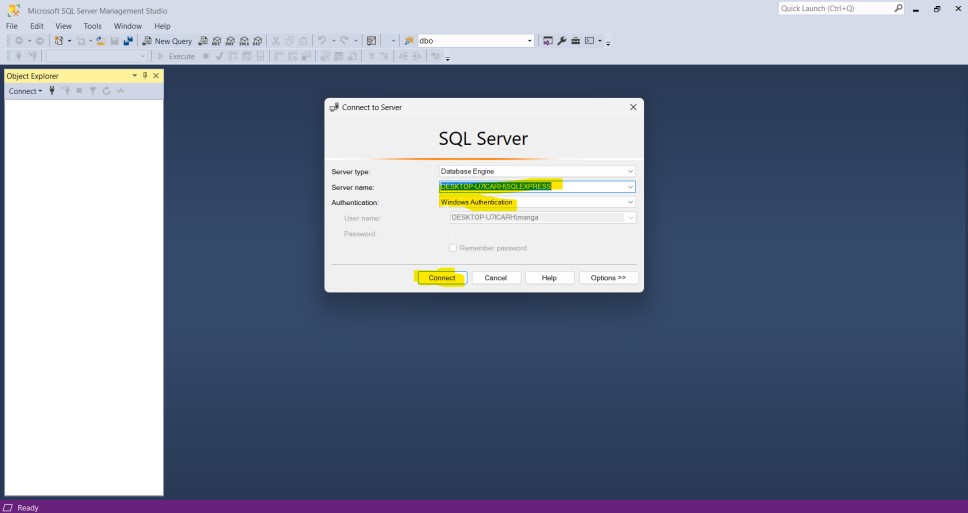
# After completion setup click on Close button



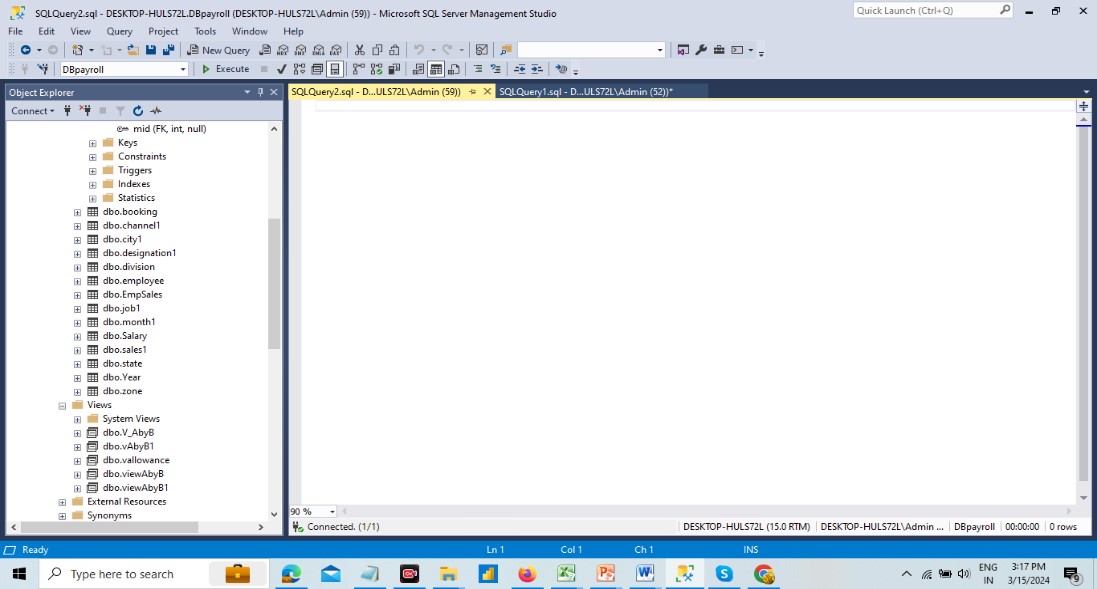
**Open SQL Server Management Studio by clicking on Search ot Start button on Windows.**



**Login to SSMS set Server name and Authentication and click on Connect button.**



# SQL Server Screen



**Date 30 Apr 2024 ( SQL Introduction)**

**Database** - A **database** is a collection of data that is organized and stored in a specific format, making it easily accessible, manageable, and reliable. It is a software system that allows users to create, read, update, and delete data in an efficient and secure manner.

**DBMS** - A database management system (DBMS) is a software program that allows users to interact with a database. It allows users to create, read, update, and delete data in the database, as well as manage the database's security and performance

**RDBMS** – RDBMS stands for Relational Database Management System. It is a type of database management system that uses the relational model to store and manage data.

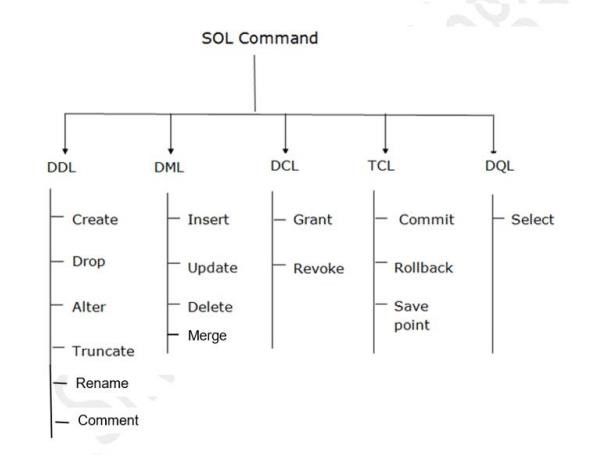
In a relational database, data is organized into tables, also known as relations. Each table has a set of rows and columns, where each row represents a record and each column represents a field. The columns in a table are called attributes and the rows are called tuples. Relationships between tables are established using keys.

# • SQL Introduction

SQL (Structured Query Language) is a standard programming language that is used to communicate with and manipulate relational databases. It is used to insert, update, delete, and retrieve data from a database, as well as to create and modify the database's structure.

SQL is used to create, alter and delete database objects, such as tables, indexes, views and procedures. It's also used to insert, update, delete and query data stored in tables.

# • SQL Commands



# SQL Commands

DDL(Data Definition Language): It deals with descriptions of the database schema and is used to create and modify the Structure of database object in database Eg. Create, Drop, Truncate, Alter, Rename & Comment

DML(Data manipulation Language): It deals with manipulation of data present in **database**

E.g. Insert, Update & Delete

DCL - Data control language is used to control data access and security of table data in SQL. DCL includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions, and other controls of the database system.

E.g. GRANT, REVOKE

DQL(Data Query Language) - DQL stands for Data Query Language. It is used to fetch the data from the database and perform queries on that data.

E.g. SELECT- This command allows getting the data out of database to perform operations with it. TCL - The TCL commands are as follows:

E.g. COMMIT, ROLLBACK, SAVEPOINT

**Date - 1 May 2024**

## Constraint

In the context of databases, a constraint is a rule that is applied to a table or column to enforce data consistency and integrity. Constraints ensure that data conforms to specific rules or conditions, preventing invalid or inconsistent data from being entered into the database. There are several types of constraints:

1. PRIMARY KEY: Ensures that each row has a unique identifier.
2. FOREIGN KEY: Establishes relationships between tables.
3. UNIQUE: Ensures that each value in a column is unique.
4. NOT NULL: Ensures that a column cannot contain null values.
5. CHECK: Verifies that data meets specific conditions or rules.
6. DEFAULT: Provides a default value for a column when no value is specified.

# Create table-

Example

CREATE TABLE DEPT\_TBL(

DEPT\_ID int PRIMARY KEY,

DEPT\_NAME varchar(50) NOT NULL,

MANAGER int)

CREATE TABLE EMPLOYEE\_TBL(

EMP\_ID int PRIMARY KEY,

EMP\_NAME varchar(50) NOT NULL,

CITY varchar(50),

DOB date,

DOJ date,

GENDER char(1) DEFAULT 'M' ,

DEPT\_ID int,

FOREIGN KEY (DEPT\_ID) REFERENCES DEPT\_TBL(DEPT\_ID) )

CREATE TABLE SALARY\_TBL(

SID int PRIMARY KEY,

SALARY bigint,

EMP\_ID int,

FOREIGN KEY (EMP\_ID) REFERENCES EMPLOYEE\_TBL(EMP\_ID) )

## Insertion of Records-

INSERT INTO DEPT\_TBL VALUES

(1, 'Finance', 2),

(2, 'Production', 3),

(3, 'Development', 1),

(4, 'Executive', 4),

(5, 'HR', 5),

(6, 'Marketing', 6)

INSERT INTO EMPLOYEE\_TBL VALUES

(101,'Sachin Kurapati', 'Mumbai','09/25/1991','03/22/2011','M',2),

(102,'Ashutosh Rajpurohit', 'Banglore', '6/15/1988','6/5/2010', 'M',1),

(103,'Yogesh Mukhtar Ali', 'Haidrabad', '6/19/1983', '12/25/2014', 'M',3),

(104, 'Deepak Rath', 'Pune', '9/14/1982','6/25/2013', 'M',1),

(105, 'Srinivas Sharma', 'Panjim', '3/12/1975','8/25/2010', 'M',5),

(106, 'Kaustubh Shinde', 'Pune', '6/15/1973','6/26/2012', 'M', 3),

(107, 'Amit Sonawane', 'Kolhapur', '7/20/1967', '4/7/2018', 'M',4),

(108, 'Shailesh Katkar','Hubli', '11/15/1992', '12/3/2012', 'M',6)

INSERT INTO DEPT\_TBL VALUES

(1, 'Finance', 2),

(2, 'Production', 3),

(3, 'Development', 1),

(4, 'Executive', 4),

(5, 'HR', 5),

(6, 'Marketing', 6)

**Date - 2 may 2024**

## SELECT statement

select \* from DEPT\_TBL select \* from EMPLOYEE\_TBL

select \* from SALARY\_TBL

## Update statement-

UPADTE salary\_tbl SET salary=salary+100

UPADTE salary\_tbl SET salary=salary+500 WHERE emp\_id=103

SELECT \* FROM SALARY\_TBL

## Delete Statement

DELETE FROM EMP\_TBL

DELETE FROM EMPLOYEE\_TBL WHERE emp\_id=108

## SELECT statement using Where clause

SELECT \* FROM EMPLOYEE\_TBL WHERE emp\_id=104

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL WHERE emp\_id=104

SELECT \* FROM EMPLOYEE\_TBL WHERE emp\_id>103

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL WHERE emp\_id<=104

## SELECT statement using Distinct clause

SELECT DISTINCT CITY FROM EMPLOYEE\_TBL

SELECT DISTINCT SALARY FROM SALARY

## SELECT statement with IN Clause

SELECT \* FROM EMPLOYEE\_TBL WHERE emp\_id IN (101,104,106,108)

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL WHERE DEPT\_ID IN (1,3)

## SELECT Statement with BETWEEN Clause

SELECT \* FROM EMPLOYEE\_TBL WHERE emp\_id BETWEEN 102 AND 106

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL WHERE emp\_id BETWEEN 102 AND 106 **Arithmetic Operator**

SELECT salary+500 FROM SALARY\_TBL

SELECT salary\*0.20 AS Bonus\_sal FROM SALARY\_TBL

SELECT Salary+(salary\*0.20) AS Total\_sal FROM SALARY\_TBL

## Logical Operator

SELECT \* FROM EMPLOYEE\_TBL WHERE emp\_id =104 OR city='Mumbai‘

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL WHERE emp\_id =104 OR city='Mumbai'

SELECT \* FROM EMPLOYEE\_TBL WHERE emp\_id =104 AND city='Mumbai‘

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL WHERE emp\_id =104 AND city='Mumbai'

## Wildcard Character

\_ Represents a single character

% Represents zero or more characters

SELECT \* FROM EMPLOYEE\_TBL WHERE emp\_name like 'S%‘

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL WHERE city like '%e‘

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL WHERE emp\_name like '\_a%'

SELECT \* FROM EMPLOYEE\_TBL WHERE city like '%a%‘

SELECT \* FROM EMPLOYEE\_TBL WHERE city like '\_\_i%'

## Order by Clause

ASC- Ascending

DESC – Descending

SELECT \* FROM EMPLOYEE\_TBL ORDER BY DEPT\_ID

SELECT emp\_id, emp\_name, city FROM EMPLOYEE\_TBL ORDER BY city, emp\_id DESC

SELECT \* FROM EMPLOYEE\_TBL ORDER BY DOB DESC ,emp\_name

SELECT \* FROM SALARY\_TBL ORDER BY salary desc

## Group By Clause

SELECT dept\_id, count(emp\_id) 'Total' FROM EMPLOYEE\_TBL GROUP BY dept\_id

SELECT City, count(emp\_id) AS TOTAL\_employee FROM Employee\_tbl GROUP BY City;

## Having Clause

SELECT emp\_id, sum(Salary) as Total\_Salary FROM salary\_tbl GROUP BY emp\_id HAVING SUM(Salary) >= 50000;

**Date - 3 May 2024**

**Joins:**

A join clause is used to combine rows from two or more tables.

**Implicit join** – Join two tables by using where clause

## The SQL Join using WHERE Clause (Implicit Join) Example-

select employee\_tbl.emp\_id, emp\_name, city, salary from employee\_tbl , salary\_tbl where employee\_tbl.emp\_id=salary\_tbl.emp\_id

**Explicit Join** - Joins two or more tables using Join Example

select employee\_tbl.emp\_id, emp\_name, city, salary

from employee\_tbl t1 join salary\_tbl t2 on t1.emp\_id=t2.emp\_id

**Types of join:**

1. Inner join – Inner join display matched records from the table.

Example-

1. select a.emp\_id, emp\_name, city, dept\_name from employee\_tbl a inner join dept\_tbl b on a.dept\_id=b.dept\_id

1. select a.emp\_id, emp\_name, city, dept\_name,salary from employee\_tbl a

inner join dept\_tbl b on a.dept\_id=b.dept\_id inner join salary\_tbl c on a.emp\_id=c.emp\_id

2.Outer join

Left Outer Join – The left join operation returns all record from left table and matching records from the right table. On a matching element not found in right table, NULL is represented in that case

Example-

1. select a.emp\_id, emp\_name, city, dept\_name from employee\_tbl a left outer join dept\_tbl b on a.dept\_id=b.dept\_id

1. select a.emp\_id, emp\_name, city, salary from employee\_tbl a

left outer join salary\_tbl b on a.emp\_id =b.emp\_id

where salary>=50000 order by a.emp\_id

Right Outer Join- The right join operation returns all record from right table and matching records from the left table. On a matching element not found in left table, NULL is represented in that case.

Example-

1. select a.emp\_id, emp\_name, city, dept\_name from employee\_tbl a right outer join dept\_tbl b on a.dept\_id=b.dept\_id

1. select a.emp\_id, emp\_name, city, salary from employee\_tbl a

right outer join salary\_tbl b on a.emp\_id =b.emp\_id where salary>=50000

order by a.emp\_id

Full Outer Join - The full outer Join keyword returns all records when there is a match in left or right table records.

Example-

1. select a.emp\_id, emp\_name, city, dept\_name from employee\_tbl a full outer join dept\_tbl b on a.dept\_id=b.dept\_id

1. select a.emp\_id, emp\_name, city, salary from employee\_tbl a

full outer join salary\_tbl b on a.emp\_id =b.emp\_id

where salary>=50000

order by a.emp\_id

3.Cross Join- Cartesian product of both the tables.

Example

SELECT A.emp\_id, emp\_name, city, Dept\_name

FROM EMPLOYEE\_TBL A CROSS JOIN DEPT\_TBL B

## SubQuery

In SQL a Subquery can be simply defined as a query within another query. In other words we can say that a Subquery is a query that is embedded in WHERE clause of another SQL query. Example

select employee.emp\_id, emp\_name, city from employee where emp\_id in (select emp\_id from Salary\_tbl where salary between 50000 and 90000)

**Date- 4 May 2024**

Normalization- Normalization is the process of organizing the attributes of the database to reduce the data redundancy.( repeated data in the database) **Purpose of normalization:**

1. Eliminating data redundancy in a table .
2. Enhance data integrity in the table.
3. Remove insert, update, and delete anomaly.

**Advantages of normalization:**

1.avoid data quality issues. 2.reduce data redundancy,

3.improve data analysis.

**4.and enhance data security.**

**Types of normal forms:**

1. **1 normal form** : - Every cell should contain atomic Value or multivalued attribute. Every column must belongs to same domain
2. **2 Normal form**: It is to be in 1F. Should not contain partial dependency. Non- Prime attribute is dependent on Candidate key(prime attribute).
3. **3 Normal form**: It is in 2NF. There is No transitive dependency(one non-prime attribute finds another non prime attribute is called transitive dependency). 3NF reduces duplication. Functional dependency from X → Y

Either X is a Super Key or Y is a Prime attribute

1. **Boyce–Codd normal form (BCNF) :** BCNF is newly defined version of 3NF. Every Functional dependency from X → Y . Either X is the super key of table Or Y is a Prime attribute.

**Date – 6 May 2024**

**SET Operators -** SET operators are special type of operators which are used to *combine the result of two queries*

**Union** - Union will be used to combine the result of two select statements. Duplicate rows will be eliminated from the results obtained after performing the UNION operation.

Example

1. select pid, pname, price from product1 union select pid, pname, price from product2

1. select pid, pname, price from product1 where pid>=2 union

select pid, pname, price from product2;

**Union All** - This operator combines all the records from both the queries.Duplicate rows will be not be eliminated from the results obtained after performing the UNION ALL operation.

Example

1. select pid, pname, price from product1 union all select pid, pname, price from product2

1. select pid, pname, price from product1 where pid>=2 union all

select pid, pname, price from product2;

**Intersect** - It is used to combine two SELECT statements, but it only returns the records which are common from both SELECT statements.

Example

1. select pid, pname, price from product1 Intersect select pid, pname, price from product2

1. select pid, pname, price from product1 where pid>=2 Inersect select pid, pname, price from product2;

**Except**- It displays the rows which are present in the first query but absent in the second query with no duplicates.

Example

1. select pid, pname, price from product1 Except select pid, pname, price from product2

1. select pid, pname, price from product1 where pid>=2 Except

select pid, pname, price from product2;

**Date - 8 May /2024**

IT firm Assignment

**Date – 9 May 2024**

Normalization Assignment

**Date-6 May 2024**

**Power BI**

Power BI is an analytics service provided by Microsoft which helps to visualize the data and share insights.

It converts data from different sources to build interactive dashboards and business intelligence reports.

Microsoft Power BI is a collection of software services, apps, and connectors that work together to turn the unrelated sources of data into interactive insights.

## Power BI Architecture



**Power BI Desktop:** It is a free software provide by Microsoft which allows us to create and publish interactive reports and dashboards. We can extract data from multiple data sources.

**Power BI services:** It is cloud-based platform provided by Microsoft that allows users to share, collaborate, and access Power BI reports and dashboards from anywhere, using a web browser or a mobile device. It can access cloud-based as well as on-premises data sources. Users can collaborate with each other using shared workspaces

There are two types of data sources from where data is fetched and loaded into Power BI desktop as well as services.

**On-Premise data source:**

* It is the data stored locally within an organizations infrastructure such as databases, file servers, or other internal systems. Power BI desktop and service require gateway to connect to on-premise data source.

**On-cloud Data source:**

* It refers to data that is stored in the cloud, such as in cloud-based databases, cloud storage, or Softwareas-a-Service (SaaS) applications.

**Gateway**- it is a secure bridge that enables secure connection between on-premises data sources and Power BI. It allows data to be refreshed and accessed securely from within Power BI.

There are two different versions in gateway:

**Standard Gateway**- It is used by multiple users within the organization. It is installed on server within an organizations network. It allows users to share access to data sources and manage data source credentials, refresh schedules, and other gateway settings.

**Personal Gateway**- It is used by individual users to access their personal data sources such as excel, CSV, and other files stored on local machine. It can only be used by the user who installed it and is not suitable for sharing data sources or managing access credentials

**Workspace-** It is a container for dashboards, reports, datasets, data marts, and dataflows in Power BI. Workspace provide a centralized location for users to create, edit, and share dashboards, reports, datasets, and dataflows. There are two types of workspaces.

1. **My workspace- I**t is the personal workspace for any Power BI user to work with their own content. Only we have access to our My workspace. We can share dashboards and reports from our My workspace. It contains datasets, workbooks, reports, and dashboards.
2. **Shared workspace/Workspaces-** Workspace that can be shared with many users (with paid users). Multiple users can collaborate on reports and dashboards within a shared workspace, and share item with other members of the organization. It contains datasets, workbooks, dataflow, reports, and dashboards.

**Dataset**- A dataset is a collection of data that we import or connect to that is used for creating reports, dashboards in Power BI.

**Dataflow**:- Data flows are a way to extract, transform, and load data from various sources into Power BI for analysis and reporting. Data flows are created and managed only in workspaces but not My workspace.

**Datamart**- A Datamart is a specialized data repository designed to serve a particular business function, department, or group.

It is a subset of large data warehouse that focuses on a specific business area or subject matter.

## Deployment Pipeline –

* It is a series of automated steps that is used for deploying reports and dashboards from a development or staging environment to a production environment.
* Deployment pipelines enable creators to develop and test Power BI content in the power BI service, before the content is consumed by users.
* The content types include reports, dashboards, datasets, and dataflows.
* Deployment pipeline has three stages:
  + **Development**: This stage is used to design, build, and upload new content with fellow creators. This is the first stage in deployment pipelines.
  + **Test**: we can enter the test stage after all the needed changes to the content are done. The modified content is uploaded to this test stage.
  + **Production**: after testing the content, the production stage is used to share the final version of your content with business users across the organization.

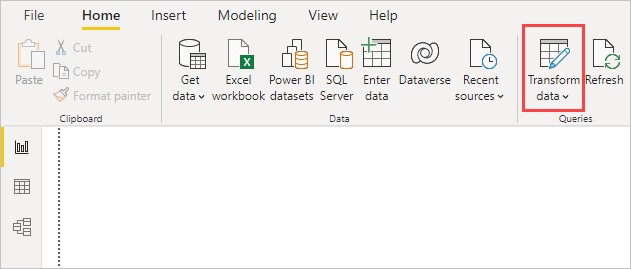
## Connectivity Modes

In Power BI, data connectivity mode refers to how Power BI Desktop connects to and retrieves data from its data sources.

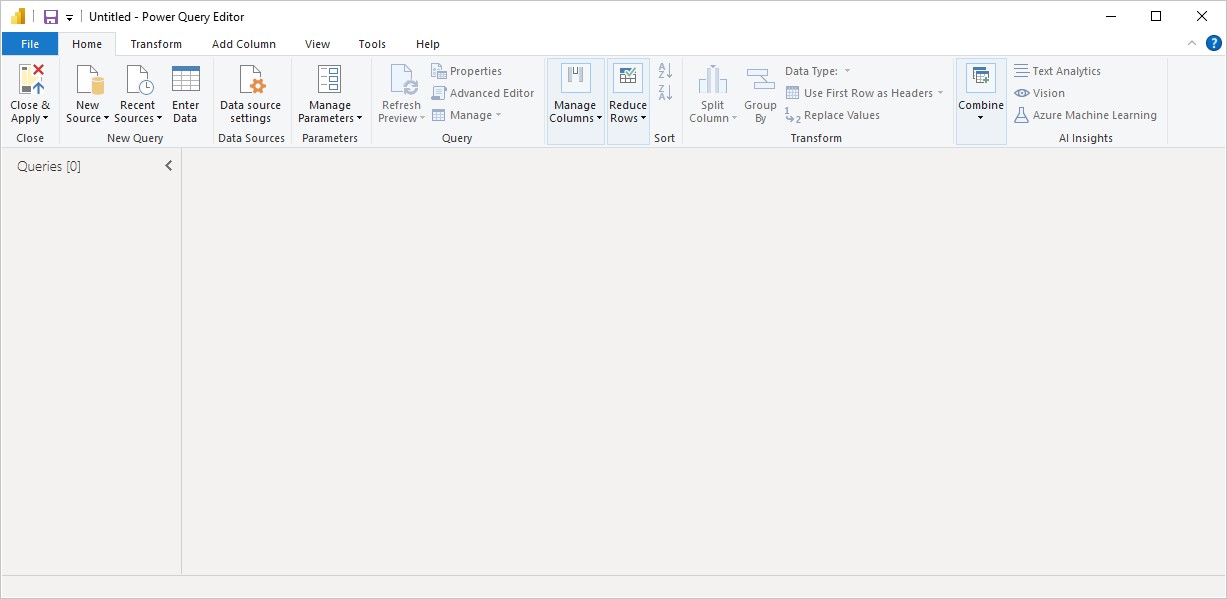
* Import
* Direct query
* Composite mode

## Power Query Editor

* To get to Power Query Editor, select **Transform data** from the **Home** tab of Power BI Desktop. • With no data connections, Power Query Editor appears as a blank pane, ready for data



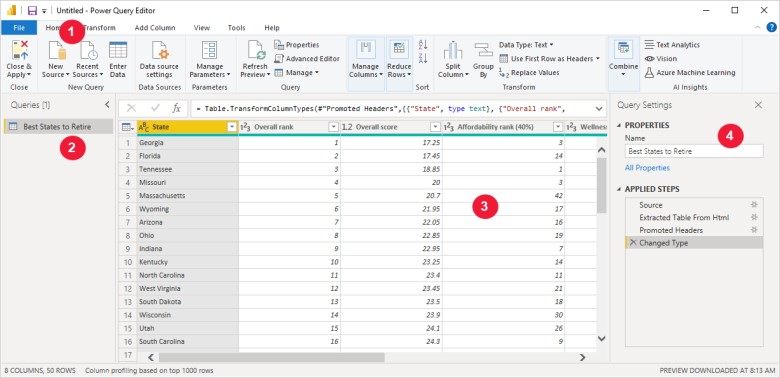
* After a query is loaded, Power Query Editor view becomes more interesting.
* If you connect to a Web data source using the New Source button in the top left, Power Query Editor loads information about the data, which you can then begin to shape.



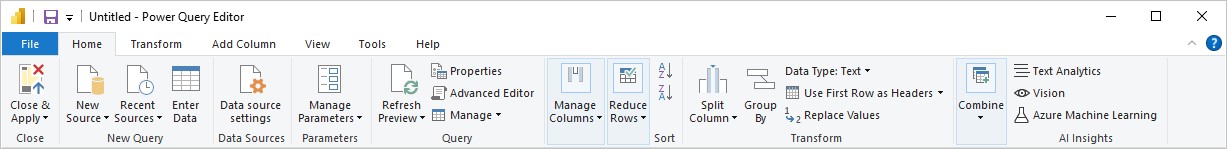
Here's how Power Query Editor appears after a data connection is established:

1. In the ribbon, many buttons are now active to interact with the data in the query.
2. In the left pane, queries are listed and available for selection, viewing, and shaping.
3. In the center pane, data from the selected query is displayed and available for shaping.
4. The Query Settings pane appears, listing the query's properties and applied steps.

Each of these four areas will be explained later: the ribbon, the Queries pane, the Data view, and the Query Settings pane.

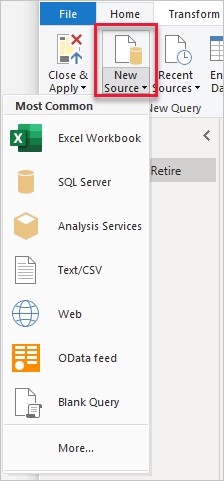


The ribbon in Power Query Editor consists of four tabs: **Home, Transform, Add** **Column, View, Tools,** and **Help.**



The **Home** tab contains the common query tasks.

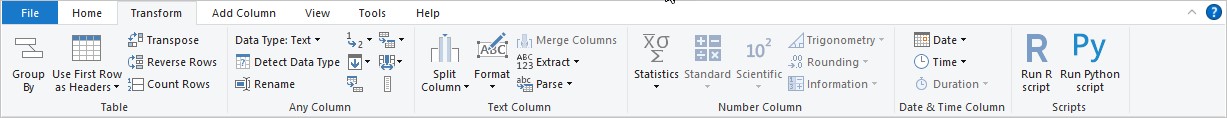
To connect to data and begin the query building process, select **New Source**. A menu appears, providing the most common data sources.



# Transform

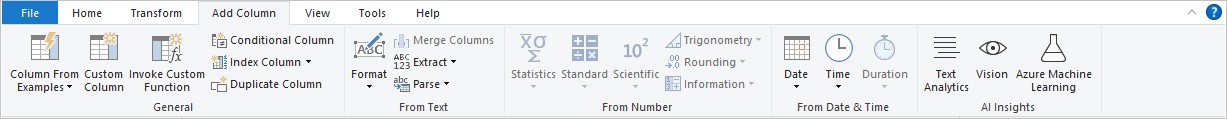
The **Transform** tab provides access to common data transformation tasks, such as:

* Adding or removing columns
* Changing data types
* Splitting columns
* Other data-driven tasks



# Add

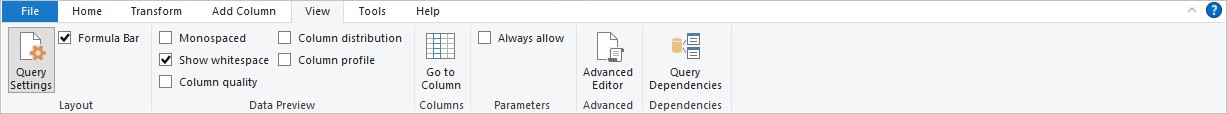
The **Add Column** tab provides more tasks associated with adding a column, formatting column data, and adding custom columns. The following image shows the **Add Column** tab.



# View

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

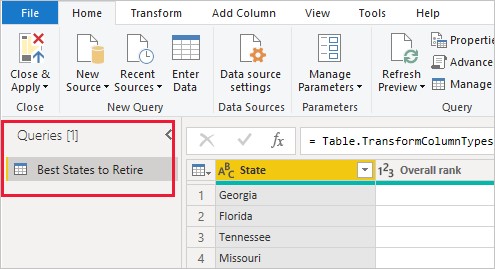
It's useful to know that many of the tasks available from the **ribbon** are also available by right-clicking a column, or other data, in the center pane.



# The left (Queries) pane

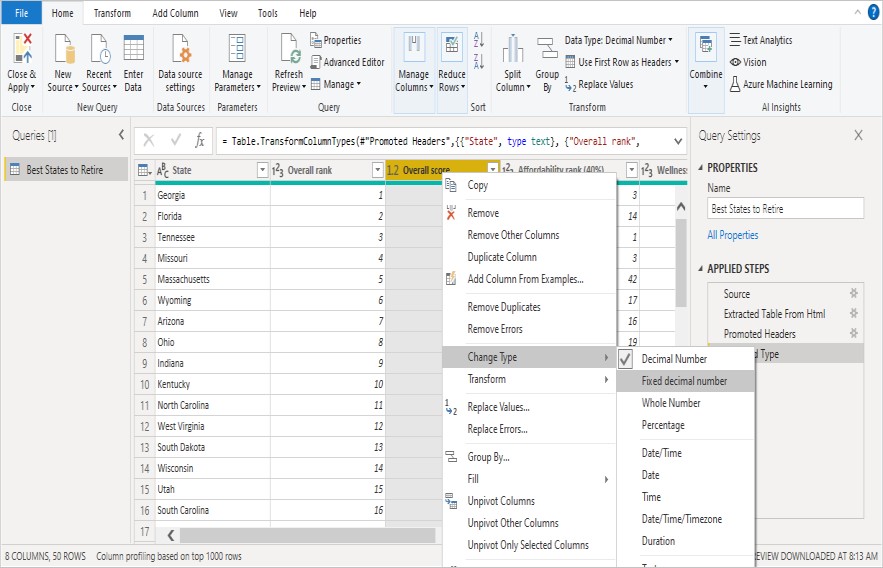
* The left pane, or **Queries** pane, displays the number of active queries and the name of the query.
* When you select a query from the left pane, its data is displayed in the center pane, where you can shape and transform the data to meet your needs.

The following image shows the left pane with a query.



# The center (Data) pane

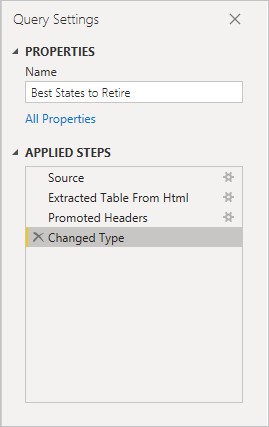
* In the center pane, or **Data** pane, data from the selected query is displayed.
* This pane is where much of the work of the **Query** view is accomplished.
* The following image shows the Web data connection established earlier.
* The **Overall score** column is selected, and its header is right-clicked to show the available menu items.
* Notice that many of these items in the right-click menu are the same as buttons in the ribbon tabs.
* When you select a right-click menu item (or a ribbon button), the query applies the step to the data. It also saves step as part of the query itself.
* The steps are recorded in the **Query Settings** pane in sequential order, as described in the next section.



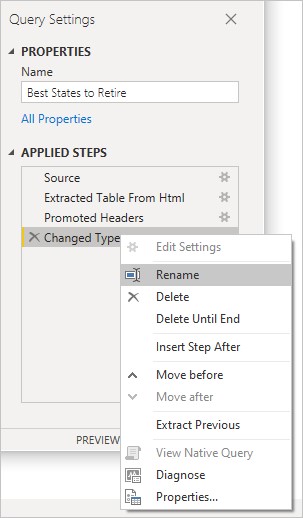
# The right (Query Settings) pane

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

As more shaping steps are applied to the query, they're captured in the **Applied Steps** section



* It's important to know that the underlying data isn't changed. Rather, Power Query Editor adjusts and shapes its view of the data.
* It also shapes and adjusts the view of any interaction with the underlying data that occurs based on Power Query Editor's shaped and modified view of that data.
* In the **Query Settings** pane, you can rename steps, delete steps, or reorder the steps as you see fit.
* To do so, right-click the step in the **Applied Steps** section, and choose from the menu that appears. All query steps are carried out in the order they appear in the **Applied Steps** pane



# Advanced Editor

* The **Advanced Editor** lets you see the code that Power Query Editor is creating with each step.
* It also lets you create your own code in the Power Query M Formula language.

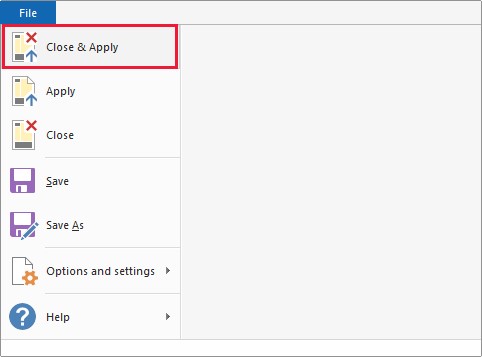
To launch the advanced editor, select **View** from the ribbon, then select **Advanced Editor.**

* A window appears, showing the code generated for the selected query. You can directly edit the code in the **Advanced Editor** window. To close the window, select the **Done** or Cancel button

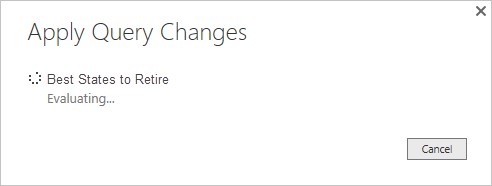


# Saving your work

When your query is where you want it, select **Close & Apply** from Power Query Editor's **File** menu. This action applies the changes and closes the editor.



* As progress is made, Power BI Desktop provides a dialog to display its status.
* When you're ready, Power BI Desktop can save your work in the form of a .pbix file.



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



## Calculated Column

**What is Calculated Column ?**

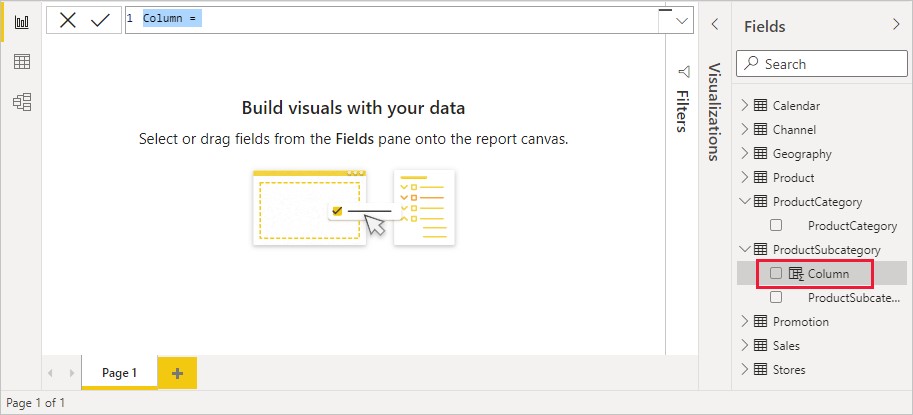
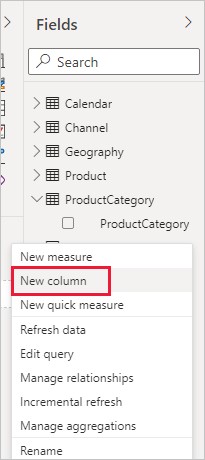
* Sometimes the data you’re analyzing doesn’t contain a particular field that you need to get your desired results. Calculated columns are useful for this situation.
* Calculated columns use Data Analysis Expressions (DAX) formulas to define a column’s values.

This tool is useful for anything from putting together text values from a couple of different columns to calculating a numeric value from other values.

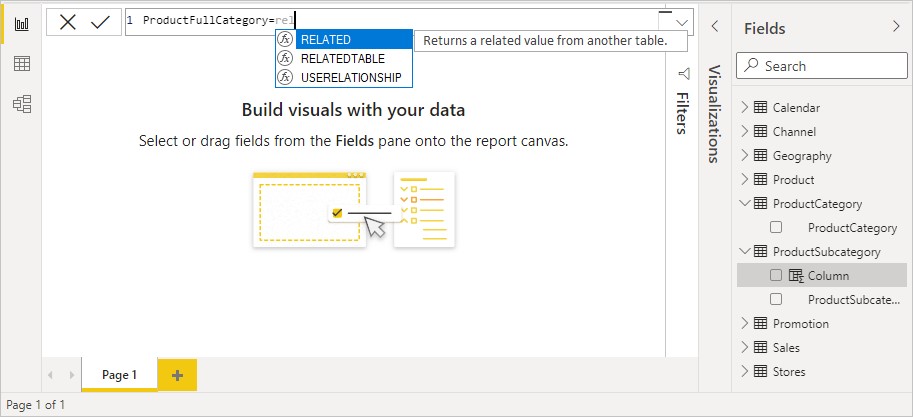
## Create a Calculated Column

* To create your new column in the table, right-click or select the ellipsis next to in the Fields pane, and choose New column from the menu.

When you choose New column, the Formula bar appears along the top of the Report canvas, ready for you to name your column and enter a DAX formula.



* By default, a new calculated column is named Column. If you don’t rename it, new columns will be named **Column 2, Column 3**, and so on. You want your column to be more identifiable, so while the **Column** name is already highlighted in the formula bar, rename it by typing , and then type an equals (=) sign.
* You want the values in your new column to start with the name in the field. Because this column is in a different but related table, you can use the RELATED function to help you get it.



* After the equals sign, type **r**. A dropdown suggestion list shows all of the DAX functions beginning with the letter R. Selecting each function shows a description of its effect. As you type, the suggestion list scales closer to the function you need. Select **RELATED**, and then press **Enter**.
* An opening parenthesis appears, along with another suggestion list of the related columns you can pass to the **RELATED** function, with descriptions and details of expected parameters.
* You want the column from the table. Select, press **Enter**, and then type a closing parenthesis.

**Date – 8-5-2024**

## Data Modelling

* Data modelling is the process of analyzing and defining all the different data types your business collects and produces, as well as the relationships between those bits of data.
* By using text, symbols, and diagrams, data modelling concepts create visual representations of data as it’s captured, stored, and used at your business.
* As your business determines how data is used and when, the data modelling process becomes an exercise in understanding and clarifying your data requirements.

## The benefits of data modeling

By modelling your data, you can document what types of data you have, how you use it, and the data management requirements surrounding its usage, protection, and governance. The benefits of data modelling include:

* Creating a structure for collaboration between your IT and business teams.
* Revealing opportunities for improving business processes by defining data needs and uses.
* Saving time and money on IT and process investments through appropriate planning.
* Reducing errors (and error-prone redundant data entry) while improving data integrity.
* Increasing the speed and performance of data retrieval and analytics by planning for capacity and growth.
* Setting and tracking target key performance indicators tailored to your business objectives. It's not just about the results of data modelling, but how you get those results.

## Data modelling concept examples

Now that you know what data modelling is and why it’s important, let’s look at the three different types of data modelling concepts as examples.

## Conceptual data modelling

* A conceptual data modelling defines the overall structure of your business and data.
* Used for organizing business concepts, your conceptual data model is defined by your business stakeholders and data engineers or architects.
* For instance, you may have customer, employee, and product data and each data bucket, known as entities, has relationships with other entities.
* Both the entities and the entity relationships are defined within your conceptual data model.

## Logical data modelling

* A logical data model builds upon the conceptual data model with specific attributes of data within each entity and the relationships between those attributes.
* For instance, Customer A buys Product B from Sales Associate C.
* This is your technical model of the rules and data structures as defined by data engineers, architects, and business analysts, helping drive decisions about what physical model your data and business require.

## Physical data modelling

* A physical data model is your specific implementation of the logical data model created by database administrators and developers.
* It is developed for a specific database tool and data storage technology, and with data connectors to serve the data throughout your business systems to users as needed.
* This is the “thing” the other models have been leading to—the actual implementation of your data estate.

## How data modelling concepts impact analytics

* Data modelling, data science, and data analytics all go hand-in-hand you need a quality data model to get the most impactful data analytics for effectual business intelligence that'll inform your future decisionmaking.
* The process of creating a data model involves forcing each business unit to look at how they contribute to their holistic organizational goals.
* Plus, a solid data model means optimized analytics performance, no matter how large and complex your data estate is or will become.
* With all your data clearly defined, analyzing exactly the data you need becomes much easier.
* Because you’ve already set up the relationships between data attributes within your data model, it’s simple to analyze and see impacts as you change processes, prices, or staffing.

## How to choose a data modelling tool

* The good news is a quality business intelligence tool will include all the Data modelling tools you need, other than the specific software products and services you choose to create your physical model.
* So, you’re free to choose the one that suits your business needs and existing infrastructure best. Ask yourself these data modelling best practise questions when evaluating a data analytics tool for its data modelling and analytics potential.

**Is this data modelling tool intuitive?**

* The technical team implementing the data model might be able to handle any tool you throw at them.
* But your business strategists and everyday analytics users basically, your business as a whole aren’t going to get optimum value out of your data modelling tool if it’s not easy to use.
* You'll want a data modelling tool with an intuitive, straightforward user experience that can help your team with data storytelling and data dashboards.

**How does this data modelling tool perform?**

* Another important attribute is performance meaning speed and efficiency, which translate into the ability to keep the business running smoothly as users run analyses.
* The best planned data model isn’t really the best if it can’t perform under the stress of real-world conditions which hopefully involve business growth and increasing volumes of data, retrieval, and analysis.

**Does this data modelling tool require maintenance?**

* If every change to your business model requires cumbersome changes to your data model, your business won’t get the best results out of that model or its associated analytics.
* Look for a data modelling tool that makes maintenance and updates easy, so your business can pivot as needed while still having access to the most up-to-date data.

**Will your data be secure with this data modelling tool?**

* Government regulations require that you protect your customer data, but the viability of your business requires protecting all your data as the valuable asset it is.
* Make sure the data modelling tools you choose have strong security measures built-in, including controls for granting access to those who need it and blocking those who don’t.

# Relationship

## Advanced Data Modeling Techniques in Power BI

Advanced data modeling techniques in Power BI include working with active and inactive relationships, leveraging USERELATIONSHIP measures, and optimizing key columns for better performance.

Mastering these advanced techniques allows you to enhance your data analysis capabilities and create even more powerful and efficient data models in Power BI.

## Active and Inactive Relationships in Power BI

Active and inactive relationships in Power BI allow for multiple connections between tables, enabling users to switch between different relationships for specific calculations.

In Power BI, only one active relationship can exist between two tables at a time. Active

relationships are used to [filter data](https://www.indeed.com/career-advice/career-development/what-is-data-filtering) from one table to another, while inactive relationships prevent data from flowing between tables.

To utilize inactive relationships, you can create USERELATIONSHIP measures or complete an additional Date dimension table.

Understanding and working with active and inactive relationships in Power BI allows you to create data models with greater flexibility in calculations and visualizations, thereby providing a more comprehensive view of your data landscape.



## USERELATIONSHIP Measures

USERELATIONSHIP measures in Power BI enable users to activate inactive relationships for specific calculations using DAX expressions. This can be particularly useful when you need to perform calculations using a relationship that is not active by default.

The USERELATIONSHIP function in Power BI can be utilized to substitute another relationship for a calculation in lieu of the active relationship.

# DAX

## What Is DAX

* DAX is a formula language specifically designed for Power BI and other Microsoft data analysis tools.
* It allows you to manipulate, transform, and calculate new data fields based on your existing data.

**Why Use DAX in Power BI?**

While Power BI offers built-in functionalities for data manipulation, DAX provides more flexibility and control. Here are some key benefits:

* **Calculated Columns**: Create new data fields based on existing ones using formulas.
* **Measures**: Develop custom calculations that summarize your data for deeper analysis.
* **Data Filtering and Slicing**: Apply advanced filtering logic to your data using DAX expressions.
* **Calculated Tables**: Construct entirely new tables based on complex calculations and relationships.

## Learning DAX Basics

Getting started with DAX doesn't require advanced coding knowledge. Power BI offers a user-friendly formula bar where you can build DAX expressions using functions, operators, and references to your data fields.

Here are some fundamental DAX functions to explore:

* **SUM, AVERAGE, MIN, MAX:** Perform basic calculations on your data.
* **COUNT, COUNTROWS**: Count the number of rows or items in your data.
* **FILTER, CALCULATE:** Apply advanced filtering logic to specific data subsets.
* **DATE, YEAR, MONTH:** Work with and manipulate date and time data. **Beyond the Basics:**

Power BI offers a vast library of DAX functions catering to complex data manipulation needs. As you progress, you can explore more advanced functionalities like:

* **Time Intelligence Functions**: Perform calculations involving dates and time periods.
* **Logical Functions:** Use IF statements and other logic to create conditional calculations.

**Relationships and Hierarchies:** Model complex relationships between your data tables for advanced analysis

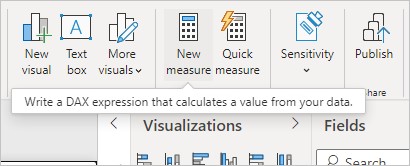
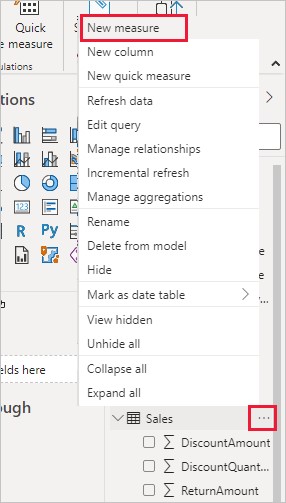
# Measures

## Create a measure

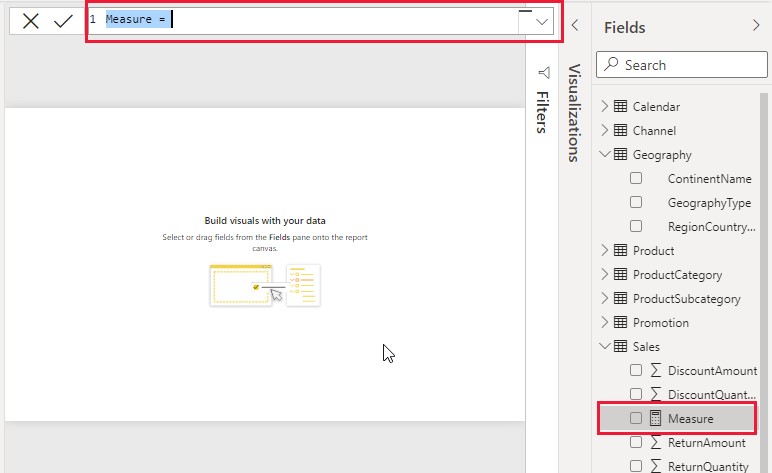
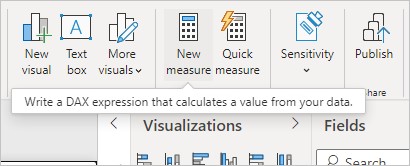
Suppose you want to analyze your net sales by subtracting discounts and returns from total sales amounts. For the context that exists in your visualization, you need a measure that subtracts the sum of DiscountAmount and ReturnAmount from the sum of SalesAmount. There's no field for Net Sales in the **Fields** list, but you have the building blocks to create your own measure to calculate net sales.

To create a measure, follow these steps:

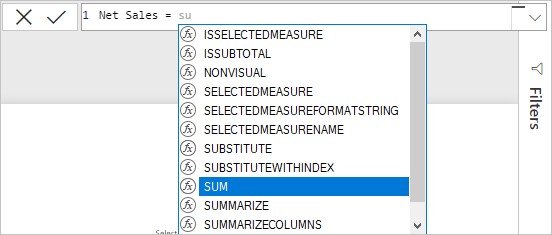
* In the **Fields** pane, right-click the **Sales table**, or hover over the table and select **More options** (...).
* From the menu that appears, choose **New measure**. This action saves your new measure in the Sales table, where it's easy to find.

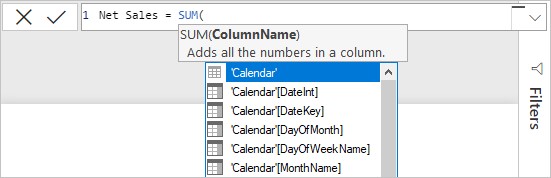


* You can also create a new measure by selecting New Measure in the Calculations group on the Home tab of the Power BI Desktop ribbon.
* The formula bar appears along the top of the report canvas, where you can rename your measure and enter a DAX formula.
* By default, each new measure is named Measure. If you don’t rename it, new measures are named Measure 2, Measure 3, and so on. Because we want this measure to be more identifiable, highlight Measure in the formula bar, and then change it to Net Sales.

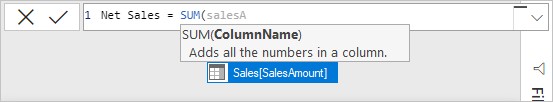


* Begin entering your formula. After the equals sign, start to type Sum. As you type, a drop-down suggestion list appears, showing all the DAX functions, beginning with the letters you type. Scroll down, if necessary, to select **SUM** from the list, and then press **Enter**.
* An opening parenthesis appears, along with a drop-down suggestion list of the available columns you can pass to the SUM function.





* Expressions always appear between opening and closing parentheses. For this example, your expression contains a single argument to pass to the SUM function: the SalesAmount column. Begin typing SalesAmount until Sales(SalesAmount) is the only value left in the list.
* The column name preceded by the table name is called the fully qualified name of the column. Fully qualified column names make your formulas easier to read.
* Select Sales[SalesAmount] from the list, and then enter a closing parenthesis.

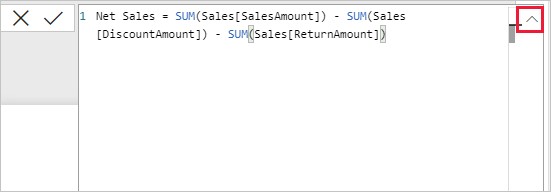
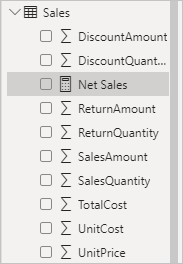


Subtract the other two columns inside the formula

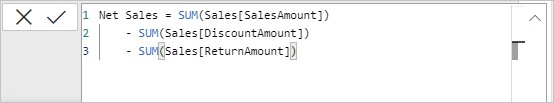
* 1. After the closing parenthesis for the first expression, type a space, a minus operator (-), and then another space.
  2. Enter another SUM function, and start typing DiscountAmount until you can choose the Sales[DiscountAmount] column as the argument. Add a closing parenthesis.
  3. Type a space, a minus operator, a space, another SUM function with Sales[ReturnAmount] as the argument, and then a closing parenthesis.



* 1. Press Enter or select Commit (checkmark icon) in the formula bar to complete and validate the formula.
  2. The validated Net Sales measure is now ready to use in the Sales table in the Fields pane.
  3. If you run out of room for entering a formula or want it on separate lines, select the down arrow on the right side of the formula bar to provide more space.
  4. The down arrow turns into an up arrow and a large box appears.



* Separate parts of your formula by pressing Alt + Enter for separate lines, or pressing Tab to add tab spacing.



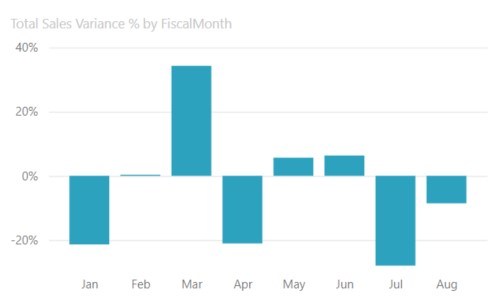
**Date – 9 May 2024 Ctarts-**

All of these visualizations can be added to Power BI reports, specified in Q&A, and pinned to dashboards. **Area charts: Basic (Layered) and Stacked**



The basic area chart is based on the line chart with the area between the axis and line filled in. Area charts emphasize the magnitude of change over time, and can be used to draw attention to the total value across a trend. For example, data that represents profit over time can be plotted in an area chart to emphasize the total profit. On the other hand, stacked area charts display the cumulative total of multiple data series stacked on top of each other, showing how each series contributes to the total.

## Bar and column charts

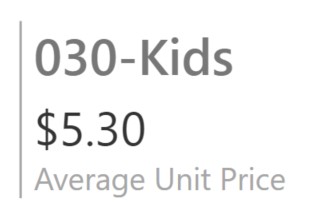




Bar charts are the standard for looking at a specific value across different categories.

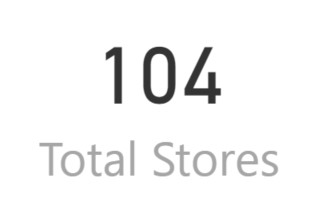
**Cards**

## Multi row



Multi row cards display one or more data points, one per row.

## Single number



Single number cards display a single fact, a single data point. Sometimes a single number is the most important thing you want to track in your Power BI dashboard or report, such as total sales, market share year over year, or total opportunities.

## Combo charts

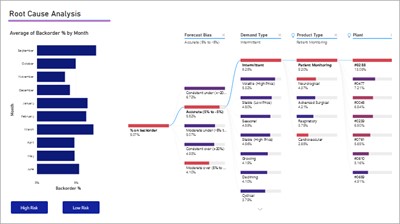


A combo chart combines a column chart and a line chart. Combining the two charts into one lets you make a quicker comparison of the data. Combo charts can have one or two Y axes, so be sure to look closely.

Combo charts are a great choice:

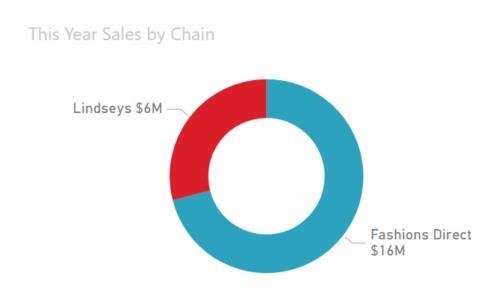
* When you have a line chart and a column chart with the same X axis.
* To compare multiple measures with different value ranges.
* To illustrate the correlation between two measures in one visual.
* To check whether one measure meets the target, which is defined by another measure.
* To conserve canvas space.

## Decomposition tree



The decomposition tree visual lets you visualize data across multiple dimensions. It automatically aggregates data and enables drilling down into your dimensions in any order. It is also an artificial intelligence (AI) visualization, so you can ask it to find the next dimension to drill down into based on certain criteria. This capability makes it a valuable tool for ad hoc exploration and conducting root cause analysis.

## Doughnut charts

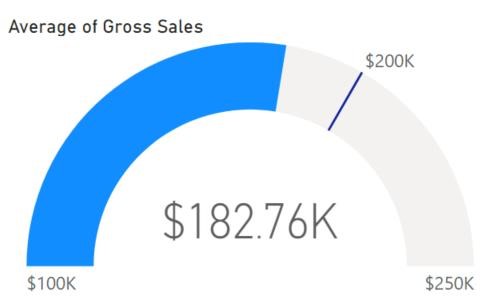


Doughnut charts are similar to pie charts. They show the relationship of parts to a whole. The only difference is that the center is blank and allows space for a label or icon. **Funnel charts**



Funnels help visualize a process that has stages, and items flow sequentially from one stage to the next. One example is a sales process that starts with leads and ends with purchase fulfillment.

For example, a sales funnel that tracks customers through stages: Lead > Qualified Lead > Prospect > Contract > Close. At a glance, the shape of the funnel conveys the health of the process you're tracking. Each funnel stage represents a percentage of the total. So, in most cases, a funnel chart is shaped like a funnel -- with the first stage being the largest, and each subsequent stage smaller than its predecessor. A pear-shaped funnel is also useful -- it can identify a problem in the process. But typically, the first stage, the "intake" stage, is the largest. **Gauge charts**



A radial gauge chart has a circular arc and displays a single value that measures progress toward a goal. The goal, or target value, is represented by the line (needle). Progress toward that goal is represented by the shading. And the value that represents that progress is shown in bold inside the arc. All possible values are spread evenly along the arc, from the minimum (left-most value) to the maximum (right-most value).

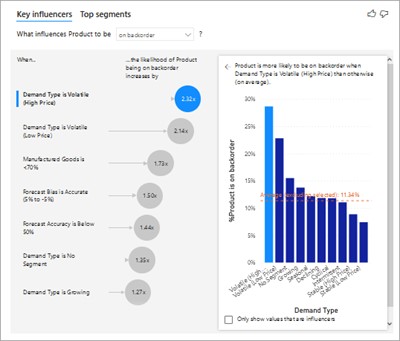
In the example, we are a car retailer, tracking our Sales team's average sales per month. Our goal is 200,000 and represented by the location of the needle. The minimum possible average sales is 100,000 and we set the maximum as 250,000. The blue shading shows that we're currently averaging approximately $180,000 this month.

Luckily, we still have another week to reach our goal.

Radial gauges are a great choice to:

* Show progress toward a goal.
* Represent a percentile measure, like a KPI.
* Show the health of a single measure.
* Display information that can be quickly scanned and understood.

## Key influencers chart



A key influencer chart displays the major contributors to a selected result or value.

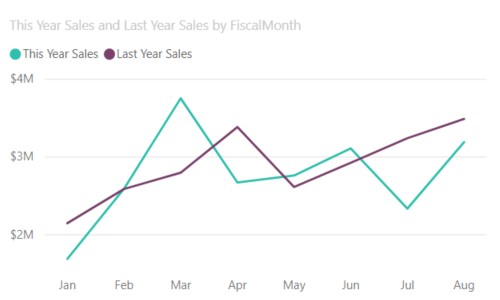
Key influencers are a great choice to help you understand the factors that influence a key metric. For example, *what influences customers to place a second order* or *why were sales so high last June*. **KPIs**



A Key Performance Indicator (KPI) is a visual cue that communicates the amount of progress made toward a measurable goal.

KPIs are a great choice:

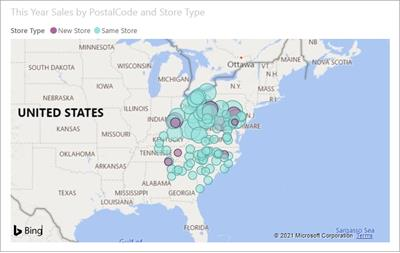
* To measure progress (what am I ahead or behind on?).
* To measure distance to a metric (how far ahead or behind am I?). **Line charts**



Line charts emphasize the overall shape of an entire series of values, usually over time.

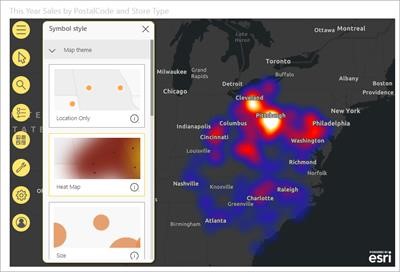
**Maps**

## Basic map



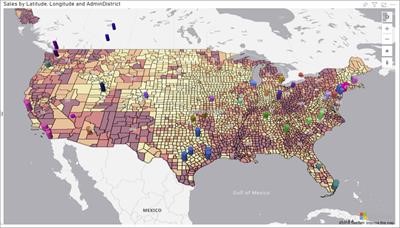
Use a basic map to associate both categorical and quantitative information with spatial locations.

## ArcGIS map

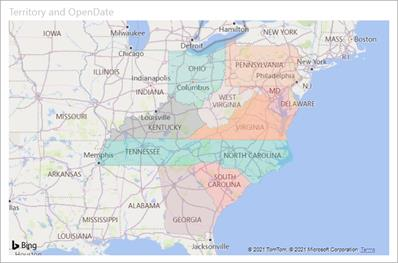


The combination of ArcGIS maps and Power BI takes mapping beyond the presentation of points on a map to a whole new level. The available options for base maps, location types, themes, symbol styles, and reference layers creates gorgeous informative map visuals. The combination of authoritative data layers (such as census data) on a map with spatial analysis conveys a deeper understanding of the data in your visual.

## Azure map

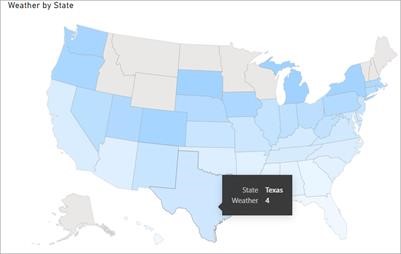


## Filled map (Choropleth)

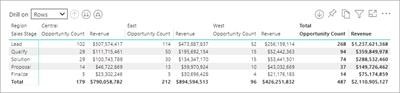


A filled map uses shading or tinting or patterns to display how a value differs in proportion across a geography or region. Quickly display these relative differences with shading that ranges from light (less-frequent/lower) to dark (more-frequent/more).

## Shape map



Shape maps compare regions on a map using color. A shape map can't show precise geographical locations of data points on a map. Instead, its main purpose is to show relative comparisons of regions on a map by coloring them differently. **Matrix**

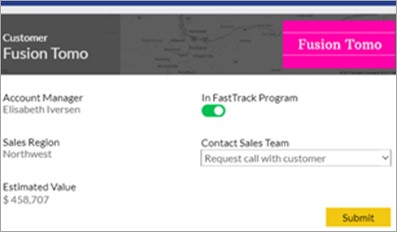


The matrix visual is a type of table visual (see [Tables](https://learn.microsoft.com/en-us/power-bi/visuals/power-bi-visualization-types-for-reports-and-q-and-a#tables) in this article) that supports a stepped layout. A table supports two dimensions, but a matrix makes it easier to display data meaningfully across multiple dimensions. Often, report designers include matrixes in reports and dashboards to allow users to select one or more element (rows, columns, cells) in the matrix to cross-highlight other visuals on a report page. **Pie charts**



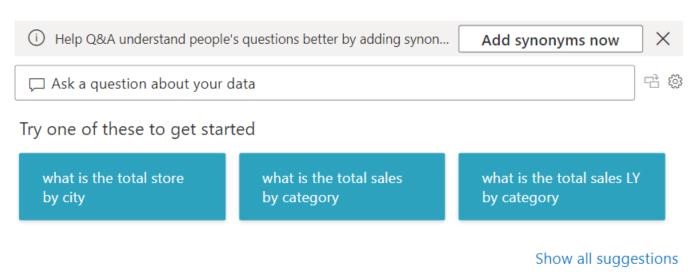
Pie charts show the relationship of parts to a whole.

## Power Apps visual



Report designers can create a Power App and embed it into a Power BI report as a visual. Consumers can interact with that visual within the Power BI report.

## Q&A visual



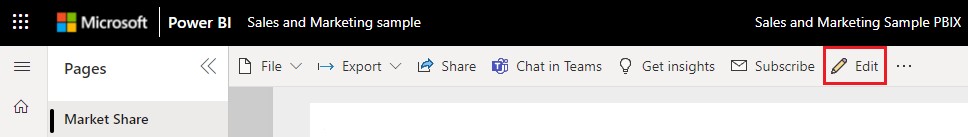
The Q&A visual provides users with a text box to query data in Power BI reports. Users can use natural language to query data, and the Q&A visual interprets the query and provides an appropriate visualization.

For example, if a user asks a question like "What were the product sales in 2019?" the Q&A visual queries the relevant data and create an appropriate visualization to display the results. This visualization can be in many different formats, such as a chart or a table.

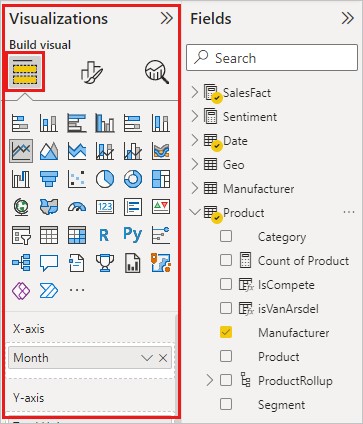
**xFormat Visuals**

### Formatting report visualizations

* If you have edit permissions for a report, there are numerous formatting options available. In Power BI reports, you can change the color of data series, data points, and even the background of visualizations. You can change how the x-axis and y-axis are presented. You can customize the data labels. You can even format the font properties of visualizations, shapes, and titles. Power BI provides you with full control over how your reports appear.
* To get started, open a report in Power BI Desktop or the Power BI service. Both provide almost identical formatting options. In the Power BI service, you need to select **Edit** from the menu bar.

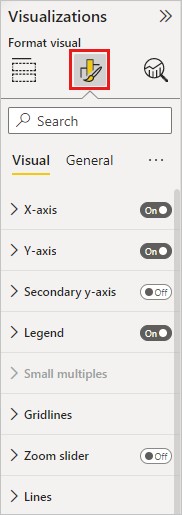


* When you’re editing a report and you select a visualization, the Visualizations pane appears. Use this pane to change visualizations.
* Directly below the Visualizations pane, there are three icons: the Fields icon (a stack of bars), the Format icon (a paint brush), and the Analytics icon (a magnifying glass).
* In the image below, the Fields icon is selected, indicated by a yellow bar below the icon.



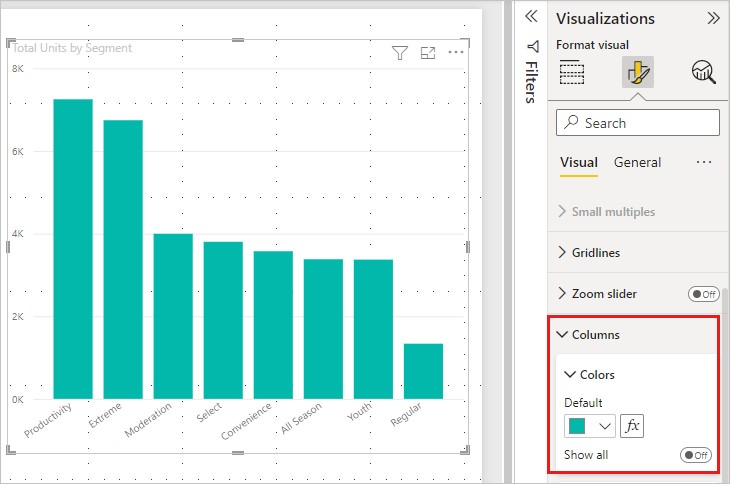
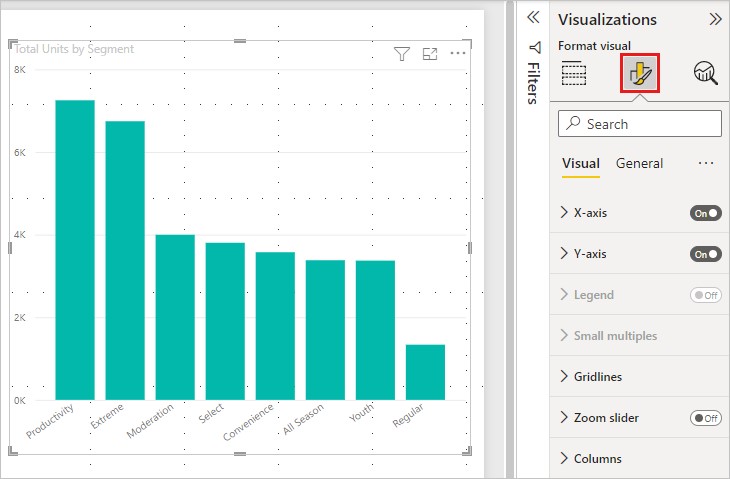
* When you select Format, the area below the icon displays the customizations available for the currently selected visualization.
* You can customize many elements of each visualization. The options available depend on the visual selected. Some of those options are:

|  |  |
| --- | --- |
| * Legend * X-axis * Y-axis * Data colors * Data labels * Total labels * Shapes * Plot area * Title | * Background * Lock aspect * Border * Shadow * Tooltip * Visual header * Shapes * Position * Zoom |

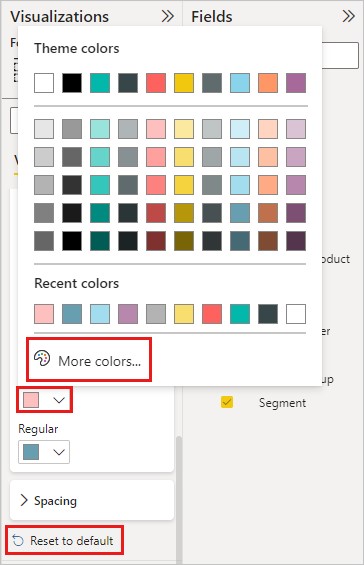
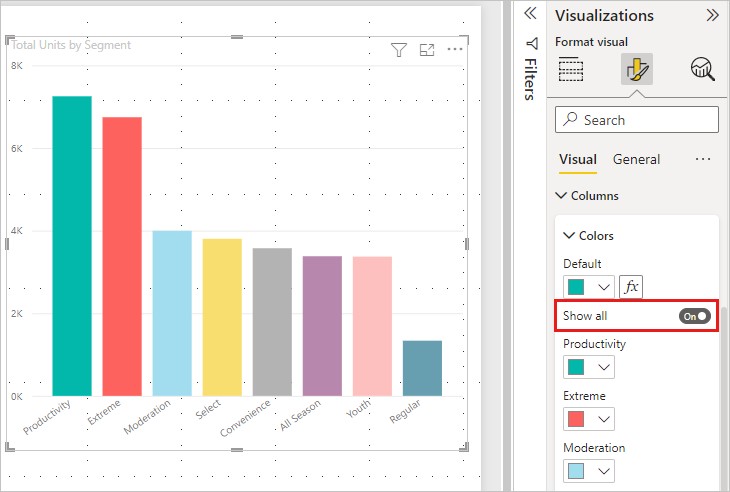


Let’s walk through the steps necessary to customize colors on a visualization.

* 1. Select a visualization to make it active.
  2. Select the paint brush icon to open the Formatting tab. The Formatting tab displays all the formatting elements available for the selected visual.



Change Show all to On, and select different colors for columns, rows, and lines, depending on the visual type.



Here are a few tips for working with colors. The numbers in the following list are also shown on the following image, indicating where you can access and change these useful elements.

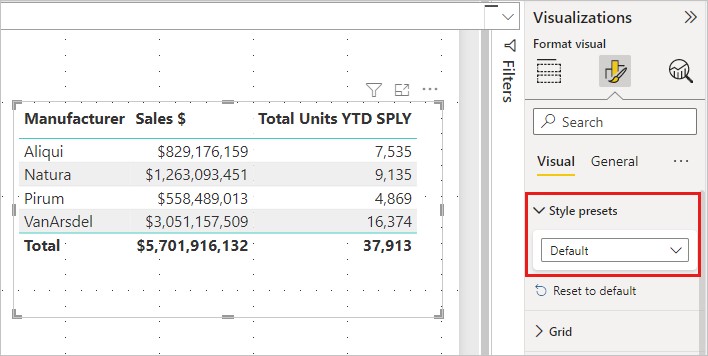
1. Don’t like the color? No problem, just select the down-arrow to open the color palette and select a new one.
2. Don't like any of the color changes? Select Reset to default from the bottom of the Data color section, and your colors revert to the default settings.
3. Want a color you don’t see in the palette? Just select More colors..., and choose from the spectrum.

### Apply a style to a table

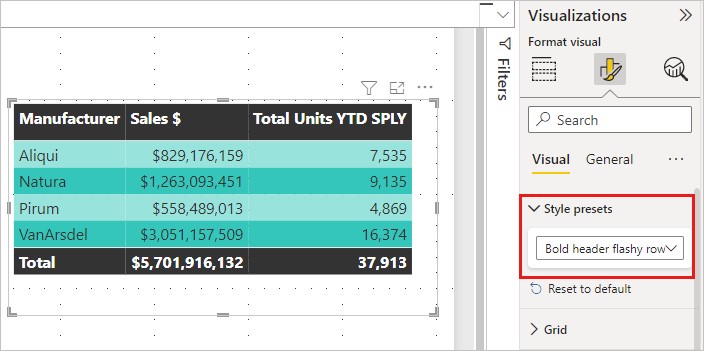
Some Power BI visualizations have a Style option. One click applies a full set of formatting options to your visualization, all at once.

1. Select a table or matrix to make it active.

Open the Formatting tab and select Style presets.

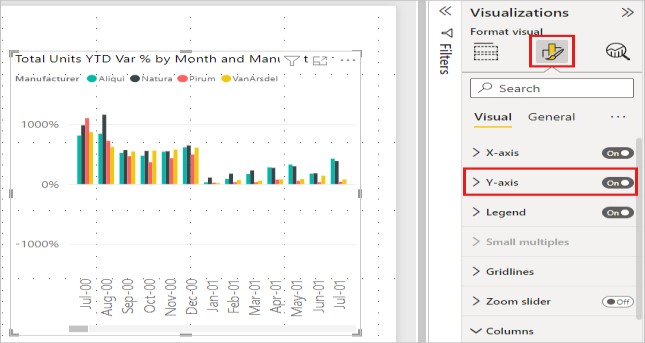


Select a style from the dropdown.



### Change axis properties

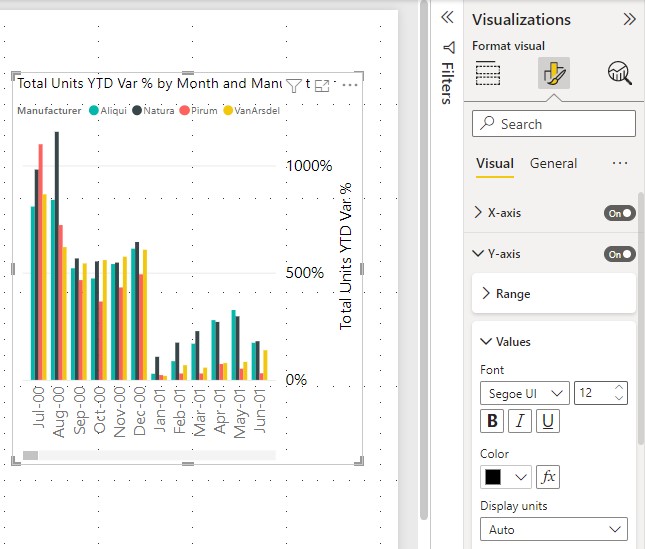
It’s often useful to modify the X-axis or the Y-axis. Similar to working with colors, you can modify an axis by selecting the down-arrow icon to the left of the axis you want to change, as shown in the following image.



The following example illustrates making the following formatting changes to the Y axis:

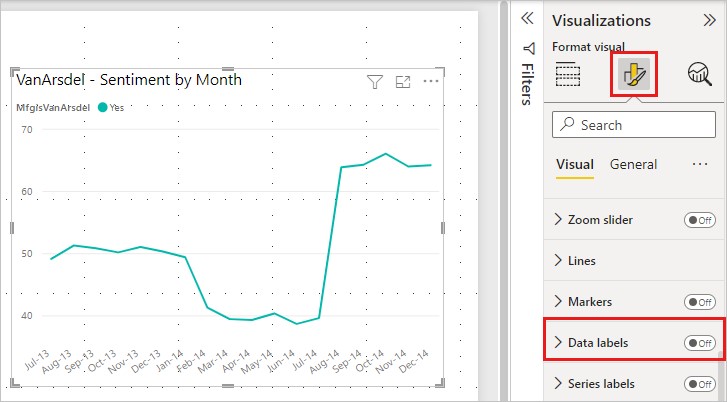
* Move the labels to the right side of the visualization.
* Change the starting value to zero.
* Change the label font color to black.
* Increase the label font size to 12.
* Add a Y-axis title.

You can remove the axis labels entirely, by toggling the radio button beside **X-Axis** or **Y-Axis**. You can also choose whether to turn axis titles on or off by selecting the radio button next to **Title**

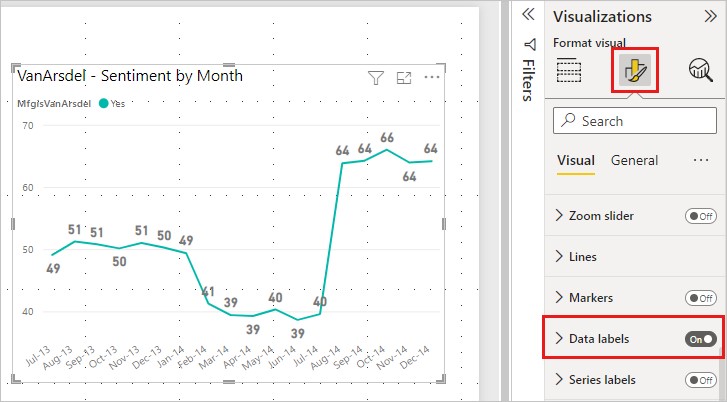


### Add data labels

**Here is the before picture.**



**Here is the after picture.**



### Add and format data labels

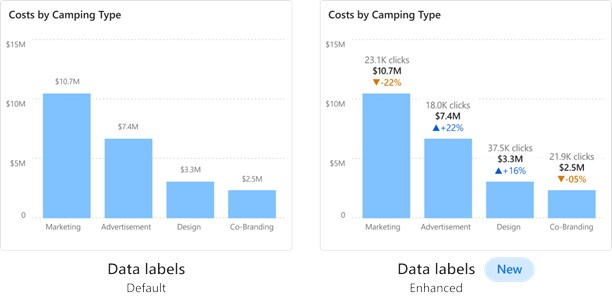
1. Select the visualization to make it active, and open the Formatting tab.
2. Select Data labels and turn them On.
3. Make these formatting changes:
   1. Increase the font size to 12.
   2. Change the font family to Arial Black.
   3. Turn Show background to On and Background color to white with a Transparency of 5%.

Depending on the visual and its size, data labels might not display. If your data labels don't display, try making the visual larger or viewing it in full screen mode.

These are just a few of the formatting tasks that are possible. Open a report in Editing mode and explore the Formatting pane to create beautiful and informative visualizations

### Customize data labels

Data labels have some enhanced formatting capabilities, giving you extended customization choices. For example, you can integrate additional metrics on your charts to improve your data analysis experience. These options are available for columns, bars, lines, and ribbon charts.

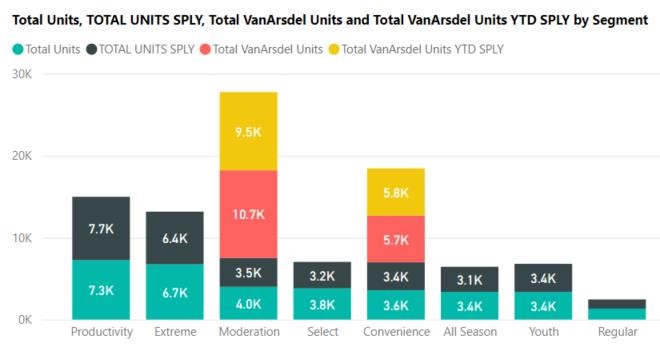


* Title: The new Title card feature enables you to display legend fields on your data labels, so you can omit the legend for a cleaner look. You can also customize data label titles using different data fields, adjustable font style, color, and transparency.
* Value: Spot the active field for your data label in the field well. Modifying it to another field is easy, with customization options like font, color, transparency, and more, plus a new feature for blank value display.
* Detail: The new detail card feature lets you add a secondary metric to your data labels. Choose any field to add an extra value, with the full range of formatting options.
* Visual label layout: Choose between a sleek single-line data label or a multi-line data label, as needed.

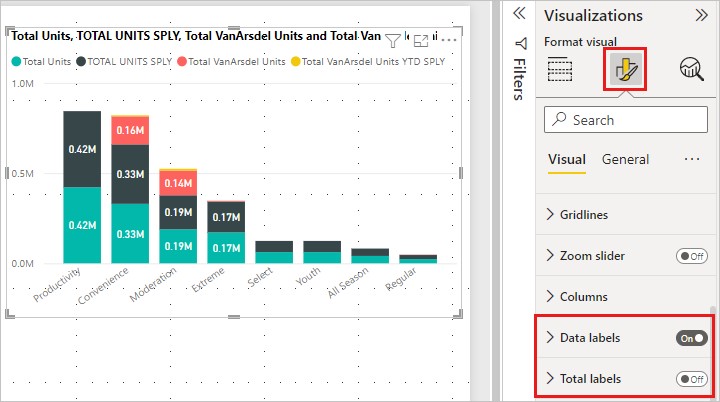
### Add total labels

* One last formatting example before you start exploring on your own. Let's add total labels to a stacked column chart. Total labels are available for stacked charts, combo charts, and area charts.
* When you turn on total labels, Power BI displays the aggregate, or total, of the data. Let's look at an example.

Here, this stacked column chart has data labels showing the value of each portion of each full stack. This view is the default

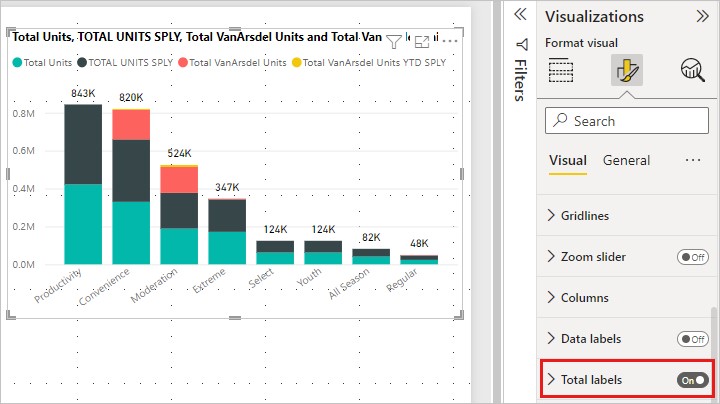


By opening the visual in Editing view, you can change the display for data labels and total labels. Select the visual to make it active and open the Formatting pane. Scroll down to **Data labels** and **Total labels**. **Data labels** is On and **Total labels** is Off.



* Turn **Data labels** Off, and turn **Total labels** On. Power BI now displays the aggregate for each column.

These are just a few of the formatting tasks that are possible. Open a report in Editing mode and have fun exploring the Formatting pane to create beautiful and informative visualizations



**Drill through**

Drill Through in Power BI is a powerful feature that helps you dive deeper into your data without leaving your main report. This tool enhances your ability to analyze and understand specific details, making your reports more interactive and insightful.

|  |
| --- |
| With **Drill Through** in Power BI reports, you can create a destination target page in your report that  focuses on a specific entity such as a supplier, customer, or manufacturer.  When your report readers right-click a data point in other source report pages, they drill through to the target page to get details that are filtered to that context. For example: selecting a specific store in a bar chart from the main dashboard can send the viewer to a details page that shows a variety of visualizations based on the selected store. |

**Date - 10 May 2024**

### Conditional Formatting

In Power BI, you can apply conditional formatting in many different ways, such as dynamically formatting individual elements in your visuals or changing the font color of data labels and chart titles.

Power BI offers an easy way of applying basic conditional formatting to change the background and font color of cells in a table or matrix. You can also add data bars (which fill the cell with a bar proportional to the value it represents) and icons (such as arrows or shapes) based on the values in the cells of a table or matrix.

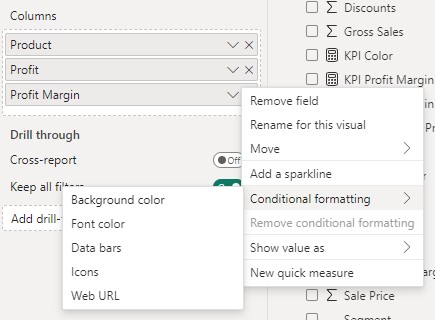
The conditions for formatting can be simple (e.g., values above or below a certain threshold) or complex, involving DAX expressions with highly specific formatting rules.

Power BI evaluates each data point against the specified conditions and applies the formatting styles automatically whenever the underlying data changes.

## Conditional Formatting in Power BI

These are the main types of conditional formatting you can apply to the table or matrix visuals in Power BI:

* Background color and font color
* Data bars
* Icons
* Web URLs



## Background color and font color

These options allow you to change the background color or font color of cells or text based on their values. Background color formatting is commonly used to create heat maps within tables or matrices, making it easier to spot high and low values through color gradients.

There are three format style options here. You can apply conditional formatting based on a field value, a gradient, or rules.

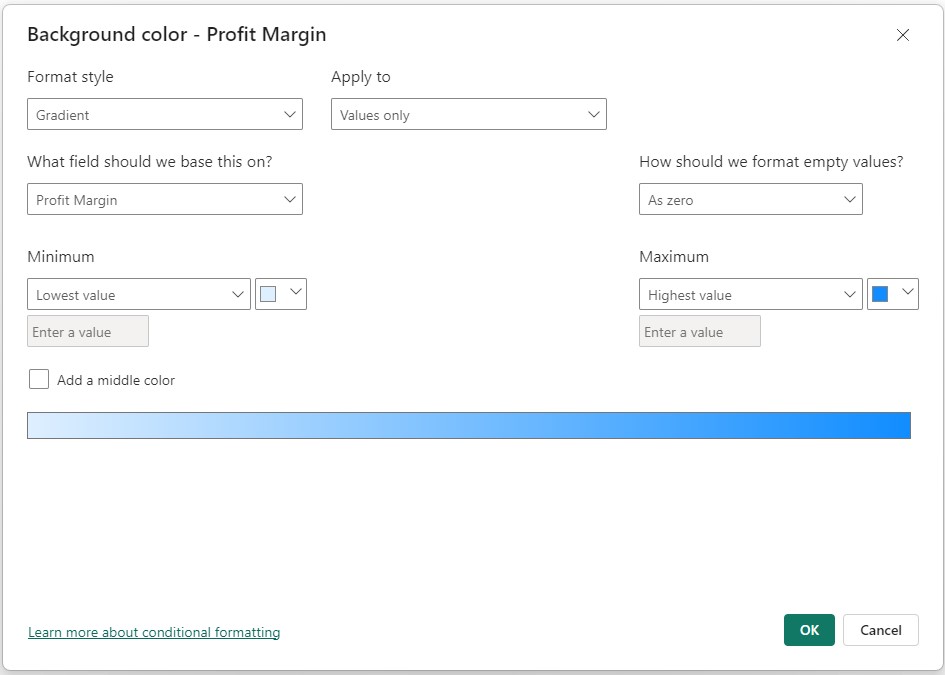


### *Field value*

Using this format style, you can apply colors based on the cell values or other fields in your data model. This option is more complex since it requires colors to be specified in a column or a DAX measure to work correctly.

### *Gradient*

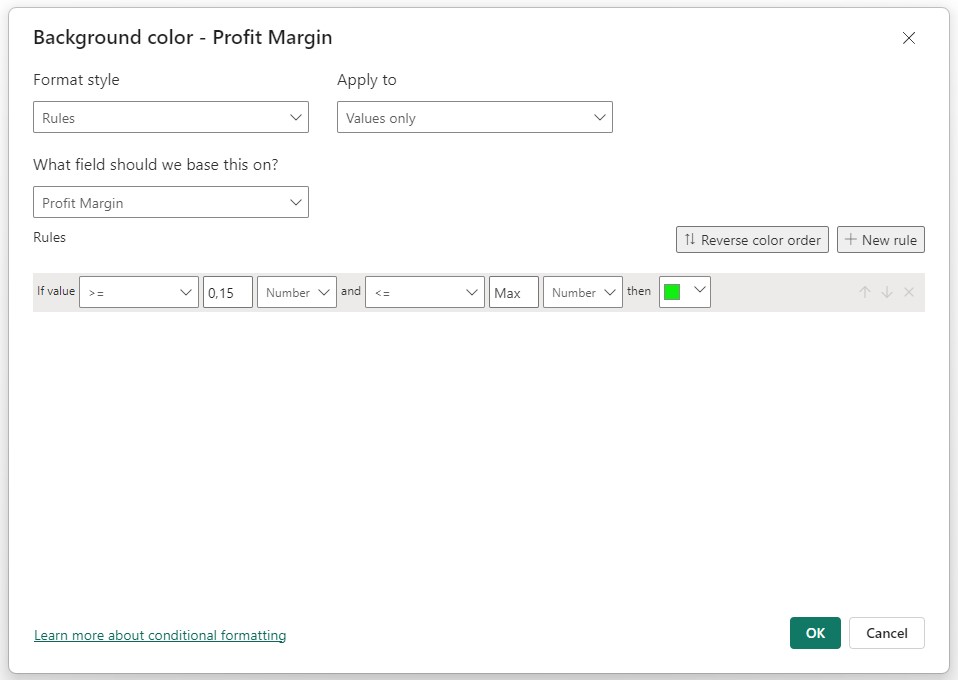
The gradient format style applies a color scale across the range of values, often from a low-value color through a midpoint (optionally) to a high-value color.



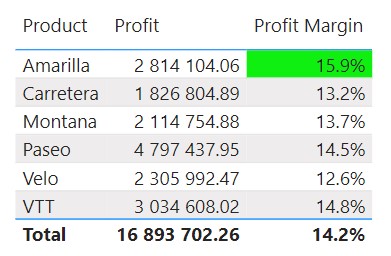
### *Rules*

Rules-based formatting lets you set specific conditions (e.g., greater than, less than, or equal to a specific value) for when to apply certain background colors or font colors. This method offers more control than the gradient and is good for comparing values against a target or predefined threshold.

You need to specify each rule individually with a color that should be applied when those rules are met. Remember to apply each rule in the correct order, especially if your rules overlap.

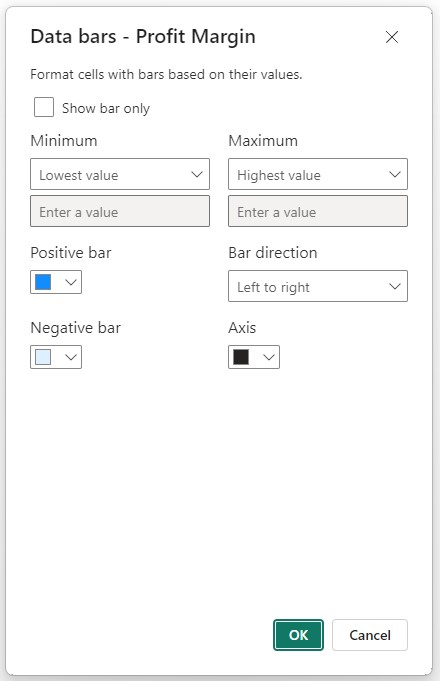


Here is an example of how the above rule would look in a table:

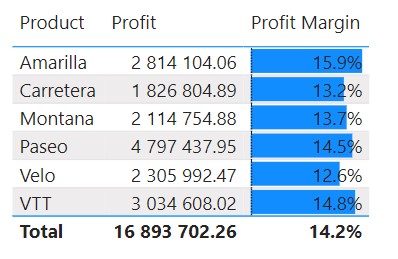


## Data bars

Data bars add a horizontal bar within the cell. The length of the bar represents the cell's value compared to other values in the column. This is a useful way of visually gauging and comparing data points directly within the cell.



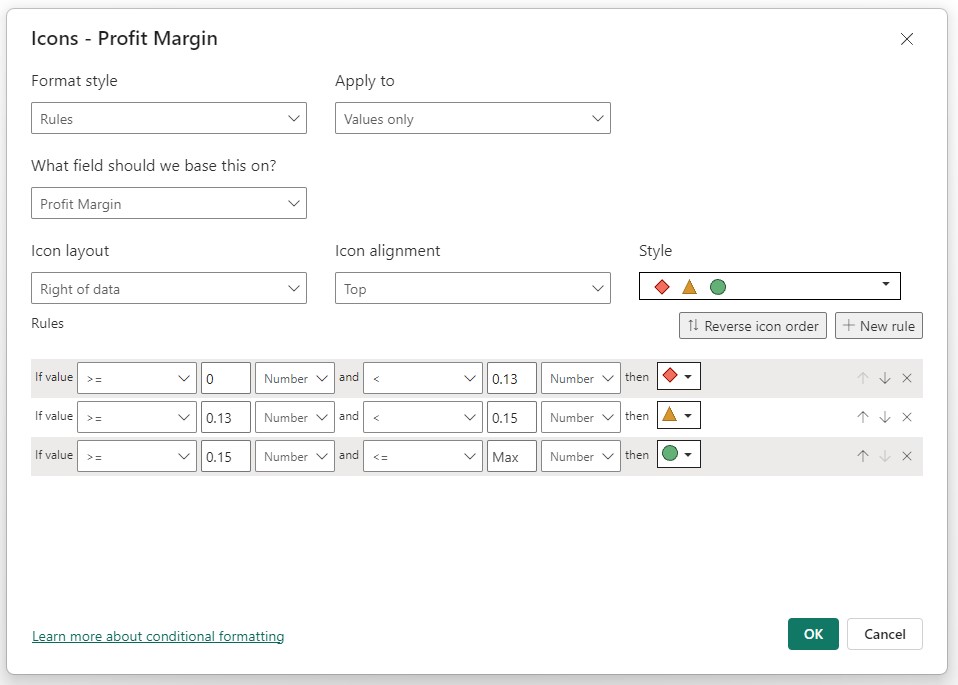
Here is an example of how the bars look in a table:



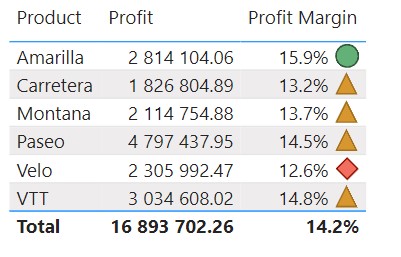
## Icons

Icon sets are applied the same way we did using the rules format style above, except you can add symbols or icons next to your data points based on the defined rules.

For example, you could use upward or downward arrows to indicate performance trends or, as in our example below, shapes to indicate whether certain targets have been met.



Here is an example of how the above icon rules look in our table:



## Web URLs

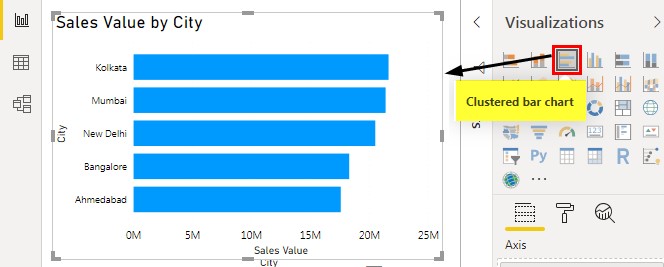
Although not a visual style per se, this type of conditional formatting allows you to add hyperlinks to your tables based on the URL data in another field. This can be useful for creating drill-through experiences or linking to external resources for additional context.

# Bookmark-

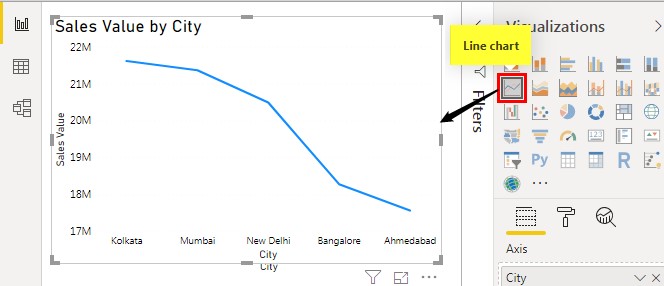
Bookmark is generally used with the web URL link to return to the bookmarked link immediately. A similar concept of the bookmark is integrated with Power BI as well. However, not to go to any URL link but to capture the current view of the visualization. Once you click on the created bookmarked icon, you can go back to the captured screen or visualization.

**Follow the below steps to create bookmarks in Power BI.**

1. Copy and paste the already created column chart on top of the same chart and change the chart type to “Clustered bar chart.”

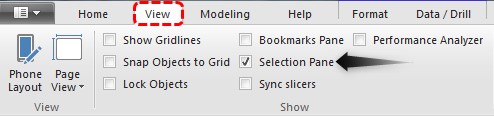


1. Copy one more, paste the existing chart, and change the chart type to “Line chart.”

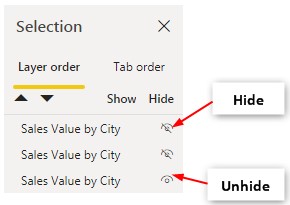


Now, we have three charts on top of each together.

1. Go to the “View” tab and check the box “Selection Pane” to view “Selection Pane” to the right.



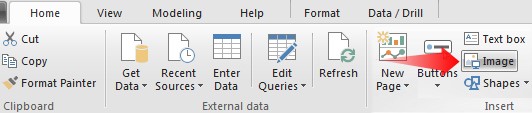
1. Now, from the “Selection Pane,” we can hide and unhide the charts we have created. Click on the “eye” icon in the “Selection Pane” to hide or unhide. We have hidden two “eye” icons. In the “Report” view, we can see only one chart, “Column Chart.”



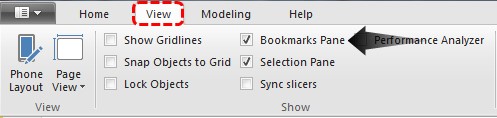
1. We have downloaded the following three types of chart visuals from the internet. And we need to insert these images into Power BI.



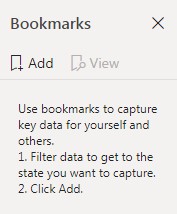
1. Go to the “HOME” tab, and click on “Image.” It will ask you to choose the image from the saved location, so choose your folder. It will insert one image simultaneously, so repeat the above step and insert all three images.



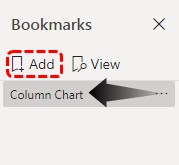
1. Go to the “View” tab and check the “Bookmarks Pane” box.



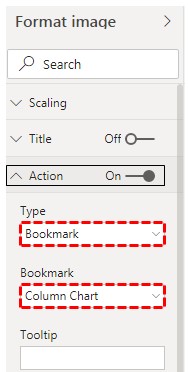
1. You can see the “Bookmarks Pane” on the right, just like the “Selection Pane.”



1. From “Bookmarks Pane,” click on “Add.” It will insert the bookmark as “Bookmark1.” Double click on this and change the name to “Column Chart.”

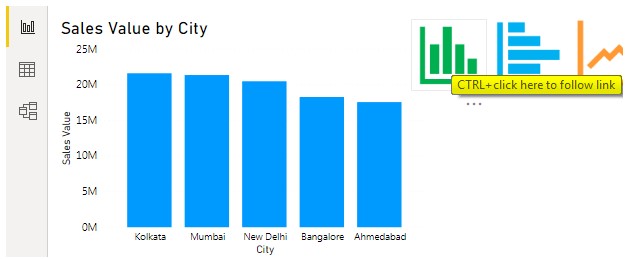


1. Now, select the “Column Chart.” It will open up the “Format Image” option to the right side. Under the “Format Image” option, turn on the “Action” option, and click on the down arrow to see further options under this. From the “Type” dropdown list, choose “Bookmark” it will open up one more “Bookmark” option from which we must choose the already named bookmark.



Like this, repeat the same for “Bar Chart” and “Line Chart.” Now, all our Power BI bookmarks are completed.

We now see “Line Chart” to see “Column Chart.” Hold the Ctrl key and click on the “Column Chart” image. It will take you to the “Column Chart” view.



Like this, we can create bookmarks to capture the moments and toggle back to those steps with just a click of the button.

**Date - 11 May 2024- 16 May 2024**

**HR Assgnment**

**Date - 17 May 2024 – 23 May 2024**

## Project