Unit-2

Import Data in PowerBI

Import Data From Files

- To import data from a file into Power BI,

- you can follow these steps

1. Open Power BI.

2. Click on "Get Data".

3. Select the type of file you want to import, such as Excel, CSV, or Text.

4. Choose the specific file or files you want to import.

5. Configure any additional settings, such as specifying a delimiter for CSV files.

Import Data from Relational Data Source

- **[Get Data From Relational Data Sources In Power BI](#)**

- If your organization utilizes a relational database for sales, **Power BI Desktop** offers the option to establish a direct connection to the database, eliminating the need for using exported flat files.

- This direct connection enables you to closely monitor business progress, discern trends, forecast sales figures, devise budgets, and establish performance indicators and targets. Power BI Desktop is compatible with numerous relational databases, whether they are hosted in the cloud or on-premises.

- Let's take a scenario where the Sales team at Tailwind Traders has requested that you connect to the organization's on-premises **SQL Server** database and get the sales data into **Power BI Desktop** so you can build sales reports.

- **SQL Database >> Power BI >> Report**

- **Connect to Data in a Relational Database**

- To access data from your relational database in Power BI Desktop, you can utilize the "**Get Data**" feature and choose the relevant option corresponding to your relational database. In this instance, the **SQL Server** option is selected, as depicted in the provided screenshot. Adjacent to the "**Get Data**" button, there are
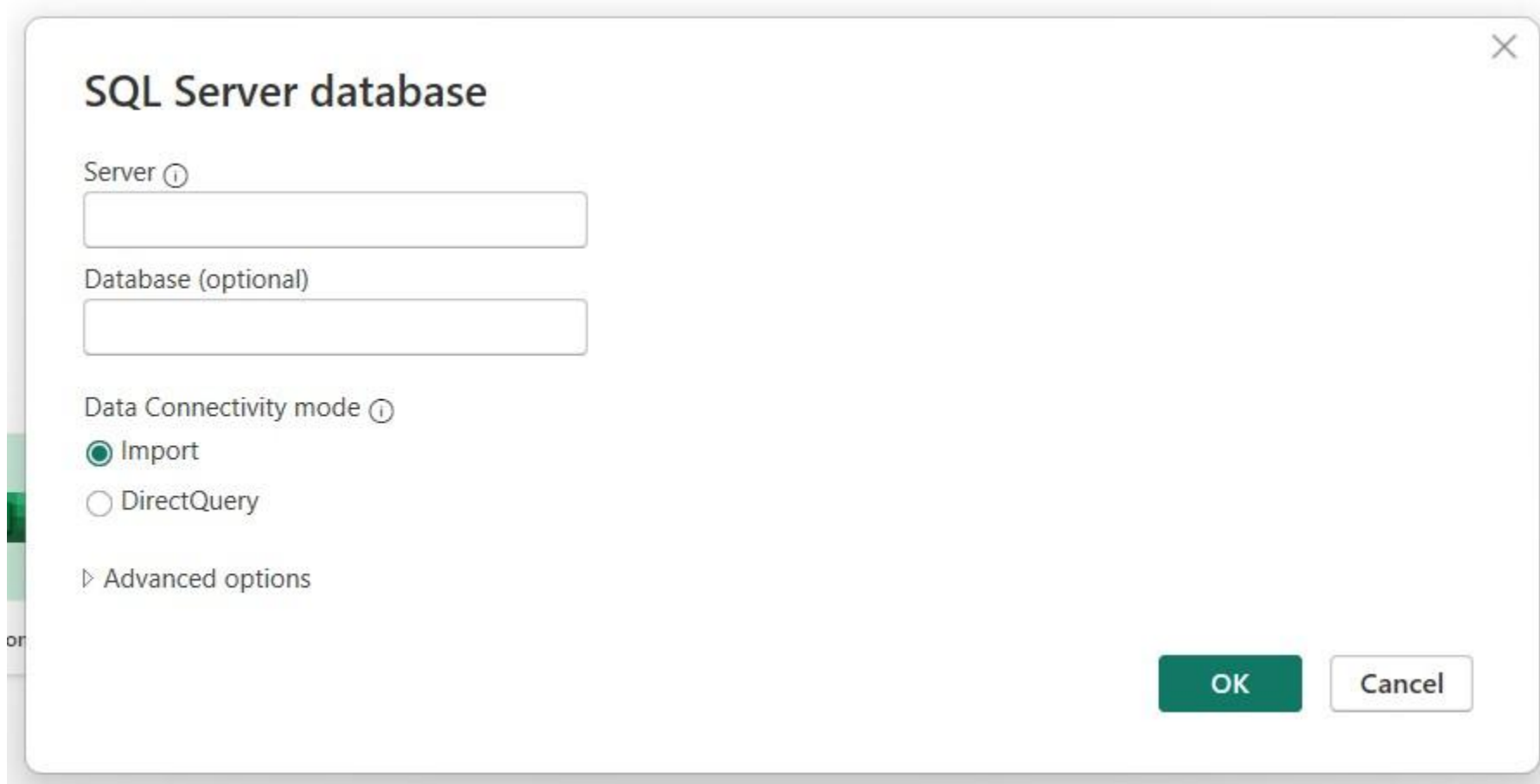
Steps:
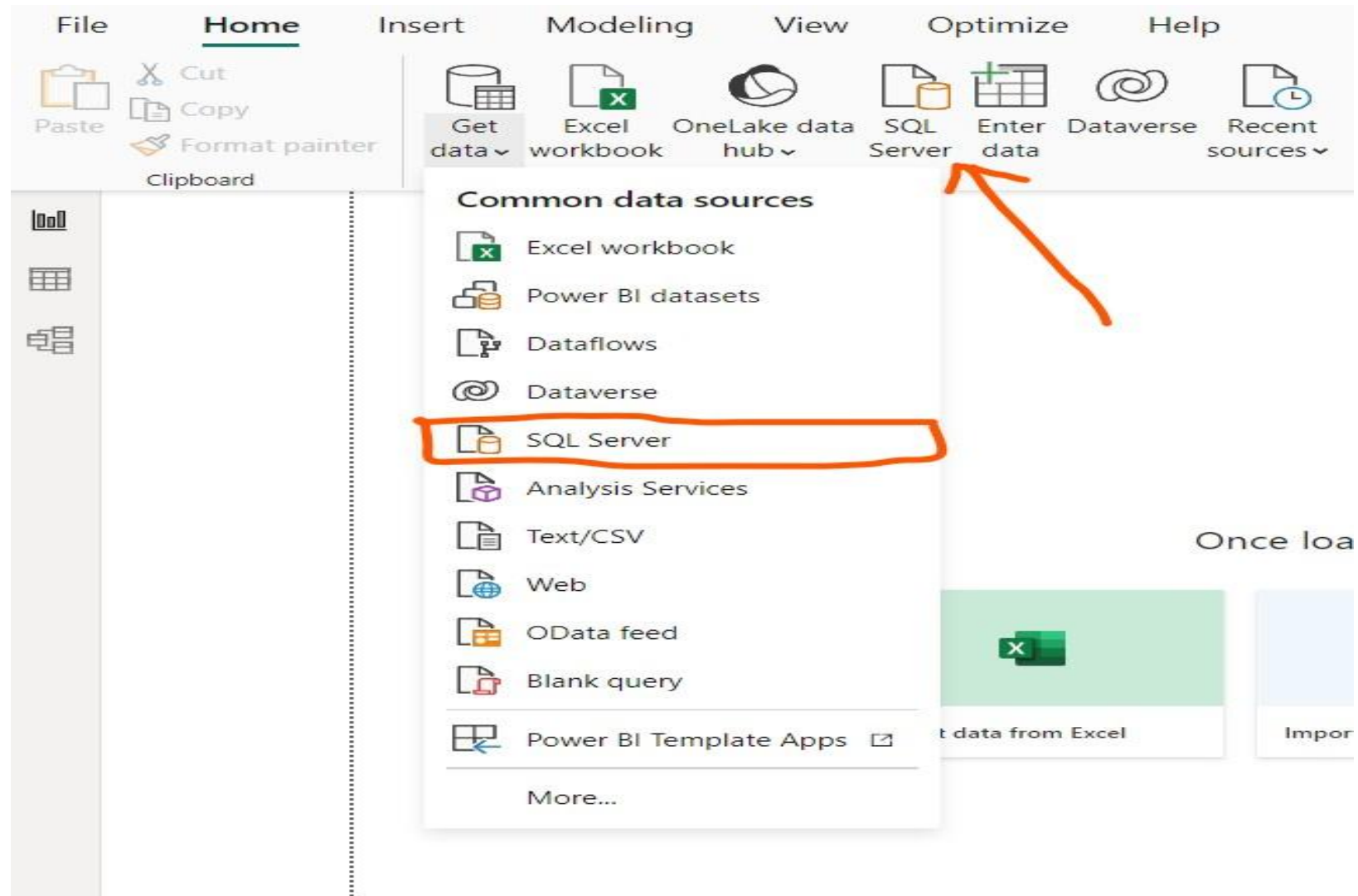
Contd..

- The next action involves entering the database server name and a database name in the SQL Server database window. Within the data connectivity mode, you'll encounter two options: Import (which is the default and recommended choice) and **DirectQuery**. Typically, the Import option is selected. While there are additional advanced options present in the SQL Server database window, they can be disregarded for the current configuration.

Contd..

- After you've added your server and database names, you'll be prompted to sign in with a username and password. You'll have three sign-in options:

- **Windows** – Use your Windows account (Azure Active Directory credentials).

- **Database** – Use your database credentials. For instance, SQL Server has its own sign-in and authentication system that is sometimes used. If the database administrator gave you a unique sign-in to the database, you might need to enter those credentials on the Database tab.

- **Microsoft account** – Use your Microsoft account credentials. This option is often used for Azure services.

- Select a sign-in option, enter your username and password, and then select Connect.

- **Select Data to Import**

- Once the database is successfully connected to Power BI Desktop, the Navigator window showcases the available data within your data source, which, in this case, is the SQL database. You have the option to choose a table or entity, preview its contents, and ensure that the accurate data will be loaded into the Power BI model.

- Select the checkbox(es) corresponding to the table(s) you wish to import into Power BI Desktop, and then proceed by choosing either the "**Load**" or "**Transform Data**" option.

- **Load** – Automatically load your data into a Power BI model in its current state.

- **Transform Data** – Open your data in Microsoft Power Query, where you can perform actions such as deleting unnecessary rows or columns, grouping your data, removing errors, and many other data quality tasks.

Navigator

Display Options ▾

▸ 📁 ▬▬▬▬▬▬▬ [2]
  ☐ ▦ ▬▬▬▬
  ☐ ▦ ▬▬▬▬▬

No items selected for preview

Load   Transform Data   Cancel

- **Import Data by Writing an SQL Query**

- An alternative method for importing data is by writing an SQL query to specifically outline the tables and columns required.

- To compose your SQL query, navigate to the SQL Server database window, input your server and database names, and then click the arrow adjacent to "**Advanced options**" to reveal additional settings. Within the "**SQL statement**" box, formulate your query statement, and then click "**OK**." In this illustration, the Select SQL statement is employed to load the ID, NAME, and SALESAMOUNT columns from the SALES table

SQL Server database

Database (optional)

Data Connectivity mode ⓘ
- ● Import
- ○ DirectQuery

▲ Advanced options

Command timeout in minutes (optional)

SQL statement (optional, requires database)

```
SELECT ID, NAME, SALESAMOUNT FROM SALES;
```

- ☑ Include relationship columns
- ☐ Navigate using full hierarchy
- ☐ Enable SQL Server Failover support

OK   Cancel

Contd..

- Write an SQL Statement

- As mentioned earlier, you have the option to import data into your Power BI model by utilizing an SQL query. SQL, or Structured Query Language, is a standardized

- programming language employed for managing relational databases and executing various data management operations.

- Imagine a scenario where your database contains an extensive table encompassing sales data spanning several years. If, for instance, sales data from 2009 is irrelevant to your report, SQL proves advantageous. It enables you to load only the necessary set of data by specifying exact columns and rows in your SQL statement, importing them into your semantic model. SQL provides flexibility to join different tables, execute specific calculations, formulate logical statements, and filter data within your query.

- In the following example, a straightforward query is demonstrated where the ID, NAME, and SALESAMOUNT are selected from the SALES table.

- The SQL query initiates with a select statement, allowing you to choose the specific fields you wish to extract from your database. In this instance, the objective is to load the ID, NAME, and SALESAMOUNT columns.

- SELECT ID, NAME, SALESAMOUNT FROM

Contd..

- FROM specifies the name of the table that you want to pull the data from. In this case, it's the SALES table. The following example is the full SQL query:

- SELECT   ID, NAME, SALESAMOUNT   FROM   SALES

- When using an SQL query to import data, try to avoid using the wildcard character (*) in your query. If you use the wildcard character (*) in your SELECT statement, you import all columns that you don't need from the specified table.

- The following example shows the query using the wildcard character.

- SELECT * FROM SALES

Contd..

- The wildcard character (*) will import all columns within the Sales table. This method isn't recommended because it will lead to redundant data in your semantic model, which will cause performance issues and require extra steps to normalize your data for reporting.

- All queries should also have a WHERE clause. This clause will filter the rows to pick only filtered records that you want. In this example, if you want to get recent sales data after January 1st, 2020, add a WHERE clause. The evolved query would look like the following example.

- SELECT   ID, NAME, SALESAMOUNT   FROM   SALES   WHERE   OrderDate >= '1/1/2024'

- A recommended best practice is to refrain from executing such queries directly within Power BI. Instead, it is advisable to formulate queries like this within a view. A view, comparable to a table, is an object in a relational database. Views possess rows and columns, and they can incorporate nearly every operator in the SQL language. When Power BI leverages a view, it engages in query folding, a functionality of Power Query. The intricacies of query folding will be elaborated upon later, but in essence, Power Query optimizes data retrieval based on how the data will be utilized subsequently.

Contd..

- Open PowerBI Desktop
- Click on GetData
- Click on More
- Click on Database
- Select Oracle
- Give the Database Server Name as :desktop-g8dllsc
- Click on Load
- Get Data from Database.
- Select Chart And do visualization and data Analysis.

Note:Sql Plus to be installed in the System

Import Data from NOSQL Database

- **Import Data from NOSQL Database**
- **How to Integrate NoSQL Database into Power BI**

Contd..

- NoSQL databases have gained significant popularity in recent years due to their scalability, flexibility, and ability to handle large volumes of data. Power BI, on the other hand, is a powerful business intelligence tool that enables organizations to analyze and visualize data effectively. In this article, we will explore the various aspects of integrating NoSQL databases into Power BI, including a brief overview of NoSQL databases, an introduction to Power BI, the benefits of integration, and a step-by-step guide to the integration process.

Contd..

- **Understanding NoSQL Databases: A Brief Overview**

- NoSQL databases, as the name suggests, are non-relational databases that provide a flexible data model for storing unstructured, semi-structured, and structured data. Unlike traditional relational databases, NoSQL databases do not rely on predefined schemas, allowing for dynamic and agile data storage. They are designed to handle the growing volumes of data generated in today's digital landscape and are known for their scalability, high performance, and fault tolerance.

- **Introduction to Power BI: An Overview**

- Power BI is a suite of business analytics tools developed by Microsoft that allows users to visualize and analyze data from a wide range of sources. It provides organizations with a comprehensive platform for data analysis, exploration, and reporting. Power BI offers various features, including interactive dashboards, data connectors, data preparation capabilities, and natural language queries. It is widely used by businesses to gain valuable insights and make informed decisions based on their data.

- **Benefits of Integrating NoSQL Databases with Power BI**

- Integrating NoSQL databases with Power BI offers several advantages for organizations. Firstly, it enables businesses to harness the power of their NoSQL data by visualizing and analyzing it in a user-friendly and interactive manner. Power BI's data connectors provide seamless integration with popular NoSQL databases such as MongoDB, Cassandra, HBase, and Apache CouchDB, allowing users to fetch and transform data easily. By integrating NoSQL databases with Power BI, organizations can unlock valuable insights, identify patterns, and discover trends from their unstructured and semi-structured data.

Contd..

- **Step-by-Step Guide to Integrating NoSQL Database into Power BI**

- Integrating a NoSQL database into Power BI involves several steps. Here is a comprehensive guide to the integration process:

- Identify the NoSQL database you want to integrate with Power BI. Evaluate the compatibility and available data connectors for the chosen database.

- Install and configure the Power BI connector specific to your NoSQL database. Follow the documentation provided by the database vendor or the Power BI community for detailed instructions.

- Establish a connection between Power BI and the NoSQL database using the configured connector.

- Define the data retrieval method – whether you want to import the data into Power BI or create a live connection to the NoSQL database. Consider factors such as data size, real-time requirements, and performance implications for your decision.

- Transform the data as per your analysis requirements using Power BI's built-in data shaping and modeling capabilities.

- Create interactive visualizations, reports, and dashboards using the transformed data.

- Publish and share the Power BI reports and dashboards with your stakeholders.

Contd..

- **Choosing the Right NoSQL Database for Power BI Integration**

- When integrating a NoSQL database with Power BI, it is crucial to select the right database that aligns with your organization's requirements. Consider factors such as the data model, scalability, performance, community support, and available connectors for Power BI. MongoDB, Cassandra, Apache CouchDB, and HBase are some popular NoSQL databases that offer seamless integration with Power BI. Evaluate these databases based on your specific needs to make an informed choice.

- **Exploring Different Types of NoSQL Databases Compatible with Power BI**

- There are various types of NoSQL databases compatible with Power BI, each offering unique features and capabilities. Let's take a closer look at a few popular ones:

- MongoDB: A document-oriented database that stores data in JSON-like documents. It is known for its flexibility and scalability, making it a popular choice for many applications.

- Cassandra: A highly scalable and distributed database that excels at handling large amounts of data across multiple nodes. It offers high availability and fault tolerance.

- HBase: A column-oriented database that is built on top of Hadoop and provides random access to large amounts of structured and semi-structured data. It is designed for high read and write throughput.

- Apache CouchDB: A database that uses a document-oriented model and provides seamless replication and synchronization capabilities. It is suitable for disconnected and distributed environments.

Contd..

- **Installing and Configuring the Power BI Connector for NoSQL Databases**

- To integrate a NoSQL database with Power BI, you need to install and configure the relevant Power BI connector. Here are the general steps to follow for the installation and configuration process:

- Download and install the Power BI connector specific to your chosen NoSQL database.

- Launch Power BI and navigate to the Get Data section.

- Select the installed connector from the list of available connectors.

- Enter the required connection details such as server address, username, password, and database name.

- Test the connection to ensure successful communication between Power BI and the NoSQL database.

- **Connecting Power BI to a MongoDB Database: A Comprehensive Tutorial**

- Connecting Power BI to a MongoDB database involves a specific set of steps. Here is a comprehensive tutorial to guide you through the process:

- Install the MongoDB Power BI connector from the official Microsoft website.

- Launch Power BI and go to Get Data > More.

- Search for the MongoDB connector and select it.

- Enter the required connection details, including the server address, database name, and credentials.

- Choose the appropriate data retrieval method – import or direct query. Importing data will create a copy in Power BI, while direct query will establish a live connection to the MongoDB database.

- Perform any necessary data transformations and modeling to suit your analysis requirements.

- Create interactive visualizations, reports, and dashboards using the MongoDB data.

Contd..

- **Optimizing Performance of NoSQL Databases in Power BI Dashboards**
- To optimize the performance of NoSQL databases in Power BI dashboards, consider the following best practices:
- Use appropriate indexing techniques provided by the NoSQL database to speed up data retrieval.
- Optimize your data modeling and structure to ensure efficient query execution.
- Implement caching mechanisms to reduce the number of database queries.
- Consider partitioning your data to distribute the workload across multiple nodes and improve query performance.
- Regularly monitor and tune the performance of your NoSQL cluster to meet the demands of Power BI dashboards.

Contd..

- **Troubleshooting Common Issues When Integrating NoSQL Databases Into Power BI**

- When integrating NoSQL databases into Power BI, you may encounter some common issues. Here are a few tips to troubleshoot and resolve them:

- Check the compatibility of your NoSQL database with the Power BI connector version you are using. Ensure you are using the latest version of both.

- Verify the connection details, such as server address, port, credentials, and database name, for accuracy.

- Confirm that the necessary drivers and prerequisites are installed on your system.

- Check for any network or firewall issues that may be blocking the connection between Power BI and the NoSQL database.

- Refer to the vendor documentation and community forums for specific troubleshooting steps related to your NoSQL database and Power BI connector.

- **Security Considerations When Integrating NoSQL Databases Into Power BI**

- Security is a crucial aspect to consider when integrating NoSQL databases into Power BI. Here are some important security considerations:

- Encrypt communication between Power BI and the NoSQL database to ensure data privacy and protection.

- Implement strong authentication mechanisms, such as username/password or key-based authentication, for secure access to the NoSQL database.

- Restrict user access and permissions based on the principle of least privilege to prevent unauthorized access or data leakage.

- Regularly update and patch the NoSQL database software to address any security vulnerabilities.

Data From Online Services

- Get Data from Online Sources into PowerI Desktop:
- Step1:Goto GetData
- Step2: select Web
- Step3:Give Link
- And connect
- https://en.wikipedia.org/wiki/UEFA_European_Football_Championship

Import Data into POwerBI Desktop.

Analyse and Visualize the data.

Create Dynamic Reports with parameters

1. Understand your data The first step is understanding which data and key points your organization want to present. ...

2. Join tables and create a parameter or a table with the main KPI's ...

3. Associate the new metrics with the data ...

4. Create all the visualisations to build the final report ...

Unit-2 Experiments

- Importing Data from excel File. Data Visualization and Data Analysis to be done with Chart and Report.

- Importing Data from CSV File. Data Visualization and Data Analysis to be done with Chart and Report.

- Importing Data from Online Sources. Data Visualization and Data Analysis to be done with Chart and Report.

- Importing Data from Relational Databases. Data Visualization and Data Analysis to be done with Chart and Report.

- Importing Data from NoSQL Database. Data Visualization and Data Analysis to be done with Chart and Report.

- Dynamic Report with Parameters.

Clean and transform the data

- To transform and clean data in Power BI, you need to

- Connect to the data source in Power BI.

- Use Power Query Editor to clean up and modify the data.

- Apply data cleaning operations like removing duplicates, filtering data, and handling missing values.

- Perform data transformation tasks such as splitting columns, merging tables, and creating calculated columns.

- Load the cleaned and transformed data into Power BI.

- Data cleaning involves several steps that include handling null or unexpected values, eliminating any duplicate values, transforming data fields into appropriate formats, using the correct data types in Power BI, and removing erroneous data to prepare the data for analysis and visualization