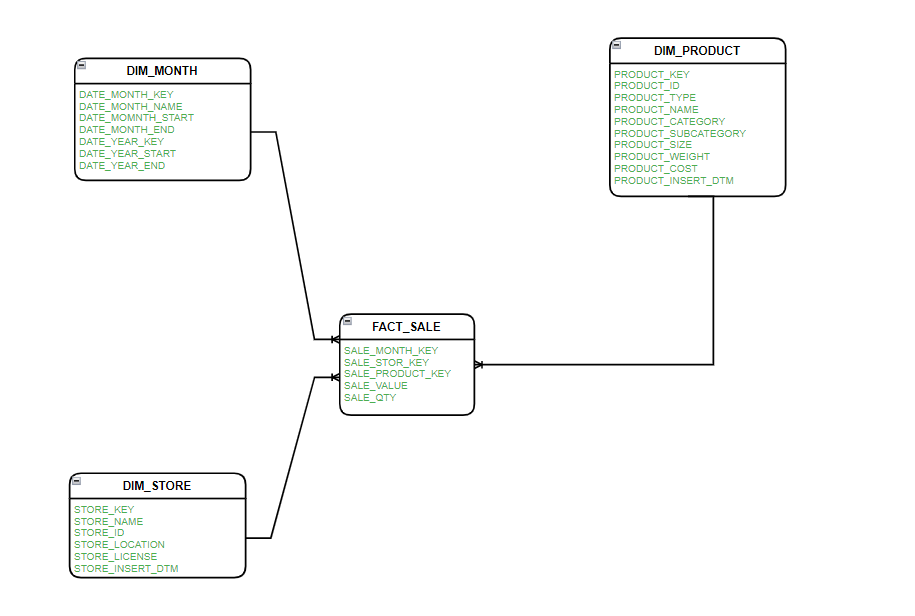
**ASSIGNMENT 4**

**6.Demonstrate fact and dimension table with an example and list their characteristics.** (<https://www.amazon.com/Data-Warehouse-Toolkit-Definitive-Dimensional/dp/1118530802> )

Fact and dimension tables are key components of a dimensional data model, commonly used in data warehousing and business intelligence. These tables are designed to efficiently store and retrieve data for analysis and reporting. Here, I'll provide an example of fact and dimension tables and list their characteristics.

**Example:** Consider a sales data scenario for a retail company. The company wants to analyze sales data, including revenue, products, customers, and regions. Here's how you can create fact and dimension tables:

**Fact Table - Sales Fact Table:** A fact table typically contains quantitative data (measures) and foreign keys to related dimension tables. In this case, the primary measure is "Sales Revenue," and it contains references to dimension tables.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OrderDate** | **ProductID** | **CustomerID** | **RegionID** | **SalesRevenue** |
| **2023-01-01** | **101** | **501** | **1** | **1000** |
| **2023-01-02** | **102** | **502** | **2** | **1500** |
| **2023-01-03** | **103** | **503** | **1** | **1200** |

**Dimension Tables:**

1. **Product Dimension Table:**
   * **Contains details about the products sold.**

| **ProductID** | **ProductName** | **Category** | **Price** |
| --- | --- | --- | --- |
| **101** | **Laptop** | **Electronics** | **800** |
| **102** | **Smartphone** | **Electronics** | **500** |
| **103** | **Headphones** | **Electronics** | **50** |

1. **Customer Dimension Table:**
   * **Contains details about customers.**

| **CustomerID** | **CustomerName** | **Email** | **Age** |
| --- | --- | --- | --- |
| **501** | **John Smith** | [**john@email.com**](mailto:john@email.com) | **35** |
| **502** | **Sarah Brown** | [**sarah@email.com**](mailto:sarah@email.com) | **28** |
| **503** | **Robert Lee** | [**robert@email.com**](mailto:robert@email.com) | **45** |

1. **Region Dimension Table:**
   * **Contains information about regions.**

| **RegionID** | **RegionName** | **Country** |
| --- | --- | --- |
| **1** | **North** | **USA** |
| **2** | **South** | **USA** |
| **3** | **East** | **Canada** |

**Characteristics:**

**Fact Table Characteristics:**

* Measures: Contains quantitative data (e.g., sales revenue).
* Foreign Keys: Contains foreign keys referencing dimension tables.
* Granularity: Typically has a fine level of granularity, representing individual transactions.
* Aggregations: Can be aggregated to various levels for reporting (e.g., daily, monthly).
* Time-variant: Contains a date or time column to track when the data was recorded.

**Dimension Table Characteristics:**

* Descriptive Attributes: Contains attributes that provide context to measures in fact tables (e.g., product details, customer information).
* Hierarchical: Often includes hierarchical attributes (e.g., product categories or geographic hierarchies).
* Low Cardinality: Has relatively low cardinality (few unique values) compared to fact tables.
* Static Data: Typically, the data in dimension tables changes less frequently, making them suitable for caching.
* Categorical Data: Contains categorical data that can be used for filtering and grouping.

**7.Demonstrate star and snow flake schema with a use case and illustration in detail. (** [**https://docs.oracle.com/en/database/oracle/oracle-database/19/dwhsg/data-warehousing-concepts.html**](https://docs.oracle.com/en/database/oracle/oracle-database/19/dwhsg/data-warehousing-concepts.html) **)**

A star schema and a snowflake schema are both database schema designs used in data warehousing and business intelligence to structure data for efficient querying and reporting. Let's demonstrate both schemas with a use case and illustrate them in detail.

**Use Case: Sales Data Warehouse**

Suppose we have a data warehousing project for a retail company, and we want to design a database schema for storing and analyzing sales data. We will consider a star schema and a snowflake schema for this use case.

**Star Schema:**

In a star schema, there is one central fact table surrounded by dimension tables, with each dimension table directly linked to the fact table. This design is straightforward and is ideal for simple and efficient querying.

**Illustration of Star Schema:**

In our sales data warehouse use case, we have the following tables:

Fact Table - "Sales":

SalesID (Primary Key)

DateKey (Foreign Key to Date Dimension)

ProductKey (Foreign Key to Product Dimension)

StoreKey (Foreign Key to Store Dimension)

SalesAmount

QuantitySold

Dimension Table - "Date Dimension":

DateKey (Primary Key)

Date

DayOfWeek

Month

Year

HolidayFlag

Dimension Table - "Product Dimension":

ProductKey (Primary Key)

ProductName

Category

Brand

Price

Dimension Table - "Store Dimension":

StoreKey (Primary Key)

StoreName

Location

Manager

Characteristics of the Star Schema:

Simple, easy to understand, and query.

Efficient for reporting and analysis.

Denormalized - no hierarchical relationships within dimension tables.

Limited to basic attributes in dimension tables.

Snowflake Schema:

In a snowflake schema, dimension tables are normalized into sub-dimensions, creating a hierarchy within the dimensions. This design reduces data redundancy but can make querying slightly more complex.

Illustration of Snowflake Schema:

In our sales data warehouse use case, the snowflake schema might look like this:

Fact Table - "Sales":

SalesID (Primary Key)

DateKey (Foreign Key to Date Dimension)

ProductKey (Foreign Key to Product Dimension)

StoreKey (Foreign Key to Store Dimension)

SalesAmount

QuantitySold

Dimension Table - "Date Dimension":

DateKey (Primary Key)

Date

DayOfWeek

Month

Year

HolidayFlag

Sub-Dimension Table - "Product Sub-Dimension":

ProductKey (Foreign Key to Product Dimension)

Category

Brand

Price

Sub-Dimension Table - "Store Sub-Dimension":

StoreKey (Foreign Key to Store Dimension)

Location

Manager

**Characteristics of the Snowflake Schema:**

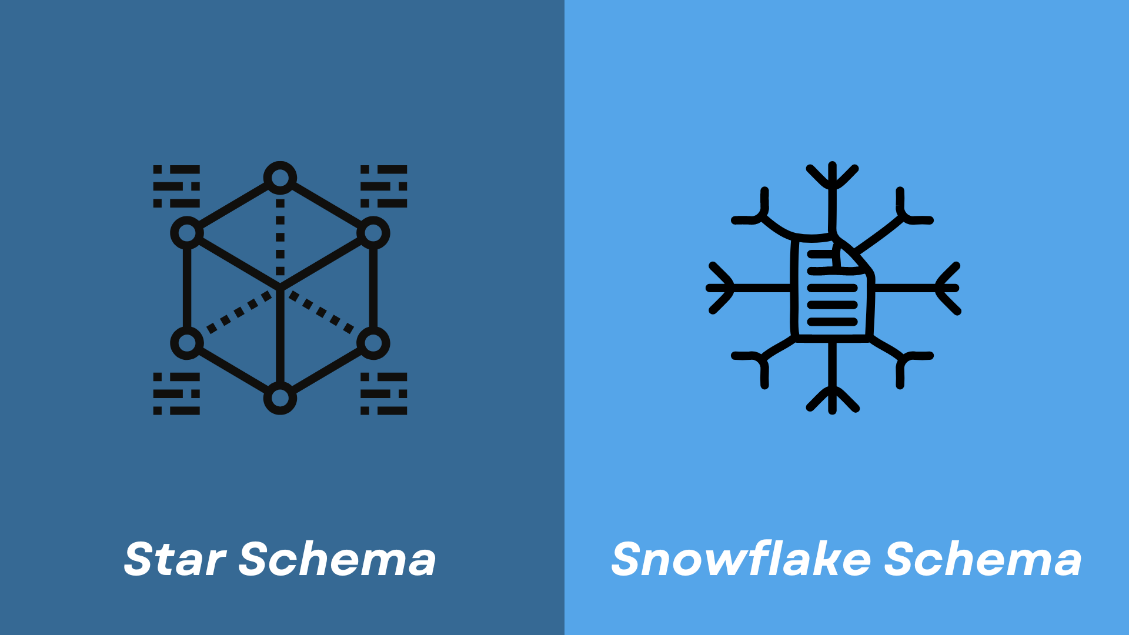
Normalized dimension tables, reducing data redundancy.

Hierarchy within dimensions, creating a more organized structure.

More complex queries due to the need for joins between dimension and sub-dimension tables.

Suitable for scenarios where data integrity and storage efficiency are critical.

In summary, the choice between a star schema and a snowflake schema depends on the specific requirements of your data warehousing project. Star schemas are simpler and efficient for reporting, while snowflake schemas are more normalized and suitable for scenarios where data integrity and storage efficiency are a priority.



**8.Demonstrate different types of relationship present between tables with an example** **(** [**https://www.w3schools.com/sql/sql\_foreignkey.asp**](https://www.w3schools.com/sql/sql_foreignkey.asp) **)**

In a relational database or data modeling context, different types of relationships describe how tables are connected or associated with each other. The common types of relationships include one-to-one, one-to-many, and many-to-many. Let's demonstrate these relationships with an example:

**Example: Online Retail Database**

Imagine you're designing a database for an online retail store, and you have three tables: Customers, Orders, and Products.

**1. One-to-One Relationship**:

In a one-to-one relationship, each record in one table is associated with exactly one record in another table. Let's say you have an additional table called CustomerDetails, which stores sensitive customer information that should be kept separately.

* **Customers Table**:
  + Fields: CustomerID (Primary Key), Name, Email
* **CustomerDetails Table**:
  + Fields: CustomerID (Primary Key), Address, Phone

In this scenario, each customer has a single corresponding record in the CustomerDetails table, and each record in the CustomerDetails table is linked to one customer in the Customers table. This is a one-to-one relationship.

**2. One-to-Many Relationship**:

In a one-to-many relationship, each record in one table can be associated with multiple records in another table. For example, in the context of online retail:

* **Customers Table**:
  + Fields: CustomerID (Primary Key), Name, Email
* **Orders Table**:
  + Fields: OrderID (Primary Key), CustomerID (Foreign Key), OrderDate

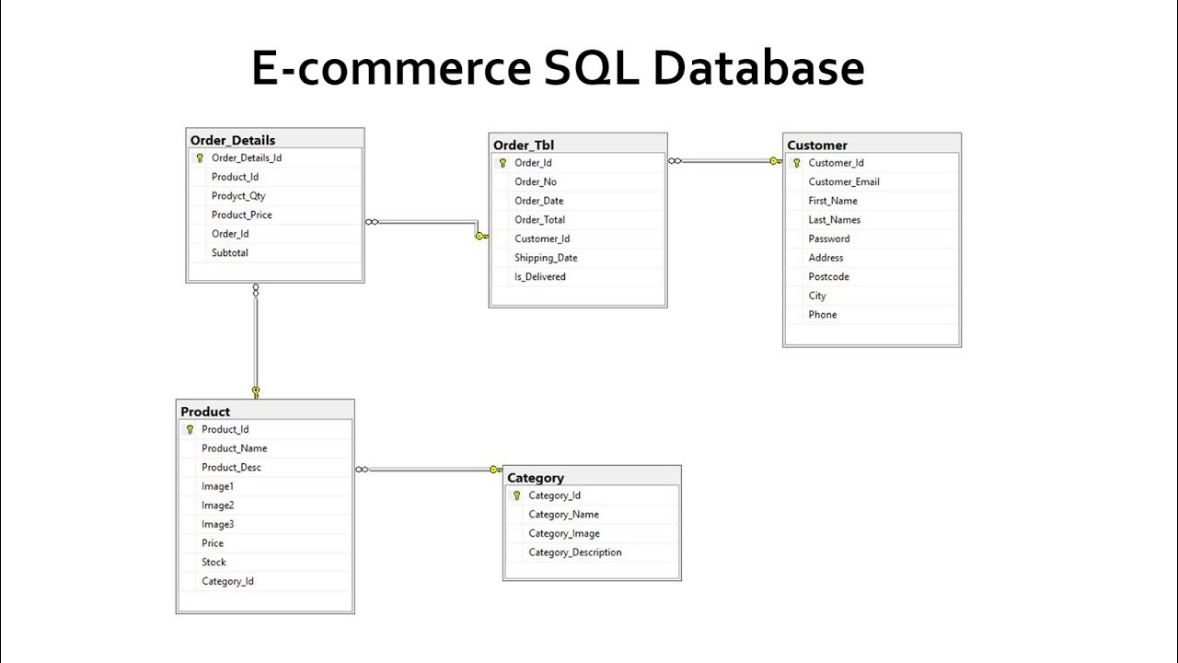
Here, each customer in the Customers table can have multiple orders in the Orders table, but each order is associated with one customer. This is a one-to-many relationship.

**3. Many-to-Many Relationship**:

In a many-to-many relationship, each record in one table can be associated with multiple records in another table, and vice versa. Consider the relationship between products and orders in the online retail database:

* **Products Table**:
  + Fields: ProductID (Primary Key), ProductName, Price
* **Orders Table**:
  + Fields: OrderID (Primary Key), OrderDate
* **OrderItems Table** (Intermediary table):
  + Fields: OrderItemID (Primary Key), OrderID (Foreign Key), ProductID (Foreign Key), Quantity

In this scenario, many products can be part of many orders, and many orders can contain many products. This is a many-to-many relationship. The intermediary table, OrderItems, is used to link products to orders.



**9.Define DAX. Discuss the features of Power BI, how DAX formulas are written. (** [**https://docs.microsoft.com/en-us/power-bi/transform-model/desktop-data-analysis-expressions**](https://docs.microsoft.com/en-us/power-bi/transform-model/desktop-data-analysis-expressions) **)**

DAX (Data Analysis Expressions) is a formula language and a collection of functions used in Microsoft Power BI, as well as other Microsoft tools like Excel Power Pivot and SQL Server Analysis Services Tabular models. DAX is designed for creating custom calculations and aggregations in data models. It is often used for creating calculated columns, calculated tables, and measures in Power BI to perform data analysis and generate insights.

Features of Power BI:

Power BI is a powerful business intelligence tool developed by Microsoft. It has several key features that make it a popular choice for data analysis and reporting:

Data Visualization: Power BI provides a wide range of data visualization options, including charts, graphs, tables, and maps. Users can create interactive and compelling reports and dashboards.

Data Transformation: Power BI includes Power Query, a data transformation tool that allows users to connect to various data sources, clean, shape, and transform data before loading it into a data model.

Data Modeling: Users can create data models by defining relationships between tables, creating calculated columns, and writing DAX measures to perform calculations on the data.

Real-Time Data: Power BI supports real-time data streaming, enabling users to work with live data and see updates as they occur.

Integration: It integrates seamlessly with other Microsoft products like Excel, Azure, and SQL Server. Additionally, it supports connections to a wide range of data sources, including cloud-based services.

Mobile Access: Power BI offers mobile apps for various platforms, allowing users to access and interact with reports and dashboards on their mobile devices.

Collaboration: Power BI Service facilitates collaboration by allowing users to share reports and dashboards with colleagues and stakeholders. It provides options for data sharing and access control.



Writing DAX Formulas in Power BI:

DAX formulas in Power BI are used for creating calculated columns, calculated tables, and measures. Here's how DAX formulas are written:

Calculated Columns: These are columns added to a table based on a DAX formula. To create a calculated column in Power BI:

In the Power Query Editor or Data View, select the table.

Go to the "Modeling" tab and click on "New Column."

Write your DAX formula in the formula bar.

**Example DAX formula for a calculated column:**

# Total Sales = Sales[Quantity] \* Sales[Price]

**Calculated Tables:** These are tables created by defining a DAX formula. To create a calculated table in Power BI:

In the "Data" view, go to the "Modeling" tab.

Click on "New Table" and write your DAX formula in the formula bar.

**Example DAX formula for a calculated table:**

# Top Products = TOPN(10, Product, [Total Sales])

Measures: Measures are used to perform calculations on data, such as sums, averages, or ratios. To create a measure in Power BI:

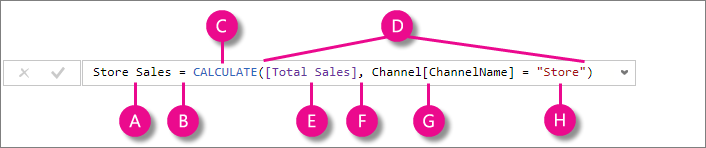
In the "Data" view, go to the "Modeling" tab.

Click on "New Measure" and write your DAX formula in the formula bar.

Example DAX formula for a measure:

# Total Sales = SUM(Sales[Total Sales])

DAX formulas use functions and operators to manipulate data, perform calculations, and aggregate values. They provide a powerful way to define custom calculations and metrics for data analysis in Power BI. DAX is essential for creating meaningful insights and reports from your data model.



**10.Demonstrate the following DAX formulas with a context: COUNT, DISTINCTCOUNT, SUM, AVERAGE, MIN, MAX,SUMMARISE, CALCULATE, IF, IFERROR, ISBLANK, EOMONTH, DATEDIFF. (** [**https://docs.microsoft.com/en-us/dax/dax-quick-reference**](https://docs.microsoft.com/en-us/dax/dax-quick-reference) **)**

let's demonstrate the usage of various DAX formulas with examples and context:

1. COUNT:

The COUNT function counts the number of rows in a table or column. It's often used to count the number of records in a dataset.

Example: Count the number of orders in a Sales table.

# Total Orders = COUNT(Sales[OrderID])

2. DISTINCTCOUNT:

The DISTINCTCOUNT function counts the number of distinct values in a column. It's useful for finding the count of unique items.

Example: Count the number of unique products sold.

# Unique Products Sold = DISTINCTCOUNT(Sales[ProductID])

3. SUM:

The SUM function calculates the sum of values in a column. It's commonly used to find the total of numeric values.

Example: Calculate the total sales amount.

# Total Sales Amount = SUM(Sales[SalesAmount])

4. AVERAGE:

The AVERAGE function calculates the average of values in a column. It's used to find the mean value of a set of numbers.

Example: Calculate the average product price.

# Average Product Price = AVERAGE(Products[Price])

5. MIN:

The MIN function returns the smallest value in a column.

Example: Find the minimum sales amount.

# Minimum Sales Amount = MIN(Sales[SalesAmount])

6. MAX:

The MAX function returns the largest value in a column.

Example: Find the maximum sales amount.

# Maximum Sales Amount = MAX(Sales[SalesAmount])

7. SUMMARIZE:

The SUMMARIZE function is used to create a summary table that aggregates data based on specified columns. It's often used for creating summary reports.

Example: Create a summary table showing total sales amount by product.

# Product Sales Summary = SUMMARIZE(Sales, Products[ProductName], "Total Sales", SUM(Sales[SalesAmount]))

8. CALCULATE:

The CALCULATE function is used to modify the filter context for a calculation. It allows you to apply filters or conditions to existing measures.

Example: Calculate total sales for a specific product category.

# Total Sales in Electronics = CALCULATE([Total Sales Amount], Products[Category] = "Electronics")

9. IF:

The IF function is used for conditional logic. It returns one value if a condition is true and another value if the condition is false.

Example: Categorize products as "High Value" or "Low Value" based on their price.

# Value Category = IF(Products[Price] > 100, "High Value", "Low Value")

10. IFERROR:

The IFERROR function returns a value if a calculation results in an error; otherwise, it returns the result of the calculation.

Example: Handle errors when calculating profit margin.

# Profit Margin = IFERROR([Total Profit] / [Total Cost], 0)

11. ISBLANK:

The ISBLANK function checks if a value or expression is blank and returns a Boolean result.

Example: Check if the "Notes" column in a table is empty.

# Has Notes = NOT(ISBLANK(Table[Notes]))

12. EOMONTH:

The EOMONTH function returns the last day of the month, a specified number of months before or after a given date.

Example: Find the last day of the current month.

# Last Day of Current Month = EOMONTH(TODAY(), 0)

13. DATEDIFF:

The DATEDIFF function calculates the difference between two dates in terms of a specified interval (days, months, years, etc.).

Example: Calculate the number of days between the order date and the ship date.

# Days to Ship = DATEDIFF(Sales[OrderDate], Sales[ShipDate], DAY)

**11.   Describe the terminology used in Power BI with a example/use case:**

1. **Table**
2. **Fact**
3. **Dimension**
4. **Calendar**
5. **Relationship**
6. **One to many, one to one, many to many**
7. **Keys - Primary & Foreign keys**
8. **Star schema**
9. **Snowflake schema**
10. **Measures**
11. **Values, Aggregation**
12. **Data Modelling**
13. **Slicer**
14. **Filter**
15. **Query**
16. **ETL**
17. **Transformations**
18. **Batch**
19. **Data Pipeline**
20. **Source**
21. **Refresh**

**(** [**https://docs.microsoft.com/en-us/power-bi/**](https://docs.microsoft.com/en-us/power-bi/) **)**

**Table:**

Definition: A table is a collection of data organized in rows and columns.

Example: In Power BI, a "Sales" table can contain data on sales transactions, including columns like "OrderID," "ProductID," and "SalesAmount."

**Fact:**

Definition: A fact table is a table that contains quantitative, numerical data representing specific business events or transactions.

Use Case: In a sales analysis, the "Sales" table can be a fact table, recording each sale's details.

**Dimension:**

Definition: A dimension table provides descriptive attributes and context for the data in a fact table.

Use Case: The "Product" and "Store" tables, containing information about products and stores, are dimension tables linked to sales data.

**Calendar:**

Definition: A calendar table is a dimension table that contains date-related attributes such as days, months, and years.

Use Case: A calendar table can be used to analyze sales trends over time or to compare performance across different years.

**Relationship:**

Definition: A relationship defines how tables are connected to each other based on shared columns.

Use Case: The relationship between the "Sales" table and the "Product" table is established using the common "ProductID" column.

**One to Many, One to One, Many to Many:**

Definition: These describe the cardinality of relationships. One to Many means one record can have multiple related records, One to One means one record is related to exactly one other record, and Many to Many means multiple records can be related to multiple other records.

Use Case: In a sales database, each customer can have multiple orders (One to Many), each employee has one manager (One to One), and students can enroll in multiple courses, and each course can have multiple students (Many to Many).

**Keys - Primary & Foreign Keys:**

Definition: A primary key uniquely identifies each record in a table, while a foreign key links a record in one table to a record in another table.

Use Case: In a "Customers" table, "CustomerID" is the primary key, and in an "Orders" table, "CustomerID" is a foreign key linking to the "Customers" table.

**Star Schema:**

Definition: A star schema is a database design with a central fact table connected to dimension tables in a star-like structure.

Use Case: In a retail sales analysis, the "Sales" table is the fact table, and "Product," "Store," and "Date" tables are dimension tables, creating a star schema.

**Snowflake Schema:**

Definition: A snowflake schema is a normalized version of a star schema, where dimension tables are further broken into sub-dimensions.

Use Case: In a snowflake schema, the "Product" dimension table may be split into "Product" and "Brand" sub-dimensions.

**Measures:**

Definition: Measures are calculations applied to fact data. They provide aggregate values for analysis.

Use Case: A "Total Sales" measure sums the sales amount in the "Sales" table.

**Values, Aggregation:**

Definition: Values are individual data points, and aggregation refers to summarizing or performing calculations on these values.

Use Case: Aggregation functions like SUM, COUNT, and AVERAGE are used to aggregate values in the "Sales" table.

**Data Modeling:**

Definition: Data modeling involves defining tables, relationships, and measures to create a structured data model for analysis.

Use Case: Data modeling in Power BI organizes data for creating insightful reports and dashboards.

**Slicer:**

Definition: A slicer is a visual control that allows users to filter and interact with data by selecting values from a list.

Use Case: Users can use a slicer to filter sales data by product category.

**Filter:**

Definition: Filters restrict the data displayed in a report to a specific subset, based on user-defined criteria.

Use Case: Applying a filter to show only sales data for a particular store location.

**Query:**

Definition: A query is a request for data retrieval from a database or data source.

Use Case: Writing a query to retrieve monthly sales data.

**ETL (Extract, Transform, Load):**

Definition: ETL is the process of extracting data from source systems, transforming it to fit the target database's structure, and loading it into the destination.

Use Case: Preparing and loading data from an external source into Power BI.

**Transformations:**

Definition: Data transformations involve cleaning, shaping, and enriching data during the ETL process.

Use Case: Transformations may include removing duplicates, formatting dates, and adding calculated columns.

**Batch:**

Definition: A batch is a group of data records processed together, typically during data import or refresh.

Use Case: Loading sales data in batches from a data source into Power BI.

**Data Pipeline:**

Definition: A data pipeline is a series of processes that move and transform data from source to destination.

Use Case: Creating a data pipeline that extracts, transforms, and loads data into a Power BI data model.

**Source:**

Definition: The source is the origin of data, such as a database, file, or web service.

Use Case: A SQL database, an Excel file, or a web API can be data sources for Power BI.

**Refresh:**

Definition: Data refresh is the process of updating the data in a Power BI dataset to reflect the latest changes in the source data.

Use Case: Scheduling regular data refreshes to keep reports up to date with the source data.