Lesson - 5

Topic: Data Modeling Basics

1. **What is a primary key in a table?**

A **primary key** is a **unique identifier** for each row in a table.

- Example: CustomerID in a Customers table.

2. **Name the two types of table relationships in Power BI.**

   - ■ **One-to-many (1:*)** – Most common
   - ■ **Many-to-many (:)** – Less common, more complex

3. **How do you create a relationship between two tables in Power BI?**

   - ■ Go to **Model View**
   - ■ Drag the key from one table to the matching column in the other (e.g., drag CustomerID in Customers to CustomerID in Sales)
   - ■ Choose relationship type (usually one-to-many)

4. **What is a "star schema"?**

A **star schema** is a modeling design with:

- **Fact table** at the center (e.g., Sales)
- Surrounded by **dimension tables** (e.g., Products, Customers, Calendar)

It looks like a **star** and improves query performance and clarity.

5. **Which table is typically the fact table in a sales dataset?**

✅ **Sales** is the **fact table**, because it stores:

- Quantities
- Revenue
- Foreign keys to Products, Customers, etc.

6. **Link Sales.csv to Customers.csv using CustomerID (one-to-many).**

☐ Ensure CustomerID is unique in **Customers**

☐ In **Model View**, drag CustomerID from **Customers** to **Sales**

☐ Set relationship as **One (Customers) to Many (Sales)**

7. **Why is ProductID in Sales.csv a foreign key?**

Because it **points to** the ProductID in the **Products** table.
It is **not unique** in Sales, but it helps relate to **product details** like name and price.

8. **Fix a relationship error where ProductID has mismatched data types.**

☐ Go to **Power Query Editor**

☐ Ensure both columns (ProductID) are of the **same data type** (e.g., whole number or text)

☐ Apply changes → Recreate the relationship

9. **Explain why a star schema improves performance.**

- Keeps **tables flat and clean**
- **Reduces redundant data**
- Allows **faster DAX queries** by simplifying joins
- Avoids complex relationships like many-to-many

10. **Add a new column TotalSales in Sales (Quantity * Price from Products).**

☐ Create a relationship between Sales and Products on ProductID

☐ Use **DAX** in Sales table:

```
TotalSales = Sales[Quantity] * RELATED(Products[Price])
```

11. **Optimize a model with circular relationships—how would you resolve it?**

☐ **Remove unnecessary relationships**

- Replace with **DAX functions** like LOOKUPVALUE or CALCULATE instead of physical relationships

- Consider using **bridge tables** to break the loop

12.Create a role-playing dimension for OrderDate and ShipDate.

- ■ Duplicate your Date table (e.g., Date, Date (Ship))
- ■ Create two relationships:

- OrderDate → Date[Date]
- ShipDate → Date (Ship)[Date]

- ■ Use USERELATIONSHIP in DAX to switch between them:

TotalSalesByShipDate =

CALCULATE([TotalSales], USERELATIONSHIP(Sales[ShipDate], 'Date (Ship)'[Date]))

13.Handle a many-to-many relationship between Customers and Products.

- Use a **bridge table** (e.g., CustomerProduct) that holds unique combinations

- Create 1:* relationships from both Customers and Products to the bridge

14.Use bidirectional filtering sparingly—when is it appropriate?

- Only use when **report filters must flow both ways** (e.g., slicers on both ends)

- Best when dealing with **lookup tables** and **many-to-many** scenarios
⚠️ Avoid in large models — can cause ambiguity and performance issues

15.Write DAX to enforce referential integrity if a CustomerID is deleted.

To return only Sales with valid CustomerID:

ValidSales =

CALCULATETABLE(

```
    Sales,
    NOT(ISBLANK(RELATED(Customers[CustomerID])))
)
```

Or, create a measure:

```
ValidSalesAmount =
CALCULATE(
    SUM(Sales[TotalSales]),
    NOT(ISBLANK(RELATED(Customers[CustomerID])))
)
```