**Documentation on Normalization in PowerBi:**

A Beginner's Guide to Data Modeling in Power BI

**1. Normalization: The Foundation**

First, let's talk about where your data comes from. Databases, like the ones that run websites or banking systems, are usually \*\*normalized\*\*.

\*\*Normalization\*\* is a process that organizes data to reduce repetition and ensure data accuracy. Think of it like a highly organized library:

- A book's title, author, and publisher are in one list.

- Information about the author's biography is in another list.

- The publisher's address is in a third list.

This setup is great for making sure every detail is correct and easy to update. However, it can be slow to pull all the information together for a report because you have to connect all those lists.

**2. Denormalization: Power BI's Preference**

For a tool like Power BI, speed is the most important thing. You want your reports to load instantly. To achieve this, we use a different approach called \*\*denormalization\*\*.

\*\*Denormalization\*\* is the opposite of normalization. It's the process of deliberately adding some controlled repetition to your data to make it faster to query and analyze. The main goal is to create a simple, easy-to-use structure called a \*\*star schema\*\*.

**3. The Star Schema: Power BI's Best Friend**

A star schema is the most common data model for Power BI. It gets its name because the diagram of the tables looks like a star, with a central table surrounded by other tables.

It has two main types of tables:

**A. Fact Tables 📈**

- These are the central tables.

- They store the \*\*numerical data\*\* or "facts" that you want to measure, like sales amounts, quantities sold, or order totals.

- Fact tables are typically very long (many rows) but narrow (only a few columns).

**B. Dimension Tables 📊**

- These tables surround the fact table.

- They provide the \*\*descriptive data\*\* or "dimensions" that give context to your facts, like product names, customer names, dates, or locations.

- Dimension tables are usually wide (many columns) but relatively short (fewer rows).

**Simple Example:** A Star Schema for Sales Data

Imagine you have a sales table. You would split it into these two types of tables:

| \*\*SalesFact Table (Fact)\*\* | \*\*ProductDim Table (Dimension)\*\* | \*\*DateDim Table (Dimension)\*\* |

| :------------------------- | :------------------------------- | :---------------------------- |

| `OrderID` | `ProductID` | `DateKey` |

| `ProductID` | `ProductName` | `FullDate` |

| `DateKey` | `Category` | `DayOfWeek` |

| `Quantity` | `Price` | `MonthName` |

| `SalesAmount` | | |

In Power BI, you would link the `SalesFact` table to the `ProductDim` and `DateDim` tables using the shared `ProductID` and `DateKey` columns. This structure makes it incredibly fast to do things like "show me the total sales amount for laptops in January."

**Why a Star Schema?**

- **\*\*Performance:\*\*** Power BI's engine is designed to work with star schemas, making your reports load very quickly.

- **\*\*Simplicity:\*\*** It's easy to build and understand, even for beginners.

- **\*\*Usability:\*\*** It makes it simple for report creators to find the right data to create visualizations.

**More Examples of Star Schema Data Models**

To really understand the star schema, let's look at how it applies to different business scenarios. In each example, we'll start with a common data problem (a single, flat table) and show how we can restructure it into a more efficient star schema for Power BI.

**Example 1: Human Resources (HR) Data**

**The Problem:** Your HR data is in a single, wide table. Every time an employee's title or department changes, it needs to be updated in many places.

The Initial, Normalized Data (not ideal for Power BI):

| EmployeeID | HireDate | EmployeeName | DepartmentName | DepartmentLocation | JobTitle | Salary |

| :----------- | :--------- | :------------- | :--------------- | :------------------- | :--------- | :------- |

| 101 | 2022-01-15 | Alice | Marketing | Floor 3 | Manager | $80,000 |

| 102 | 2022-02-20 | Bob | Sales | Floor 2 | Rep | $65,000 |

| 103 | 2022-03-10 | Alice | Marketing | Floor 3 | Analyst | $70,000 |

Notice how "Alice" has two rows because her job title changed, and "Marketing" and "Floor 3" are repeated.

**The Star Schema (better for Power BI):**

Here, we'll create a **Fact** table for salary events and a **Dim** table for the employee details.

|  |  |  |
| --- | --- | --- |
| **SalaryFact Table (Fact)** | **EmployeeDim Table (Dimension)** | **DateDim Table (Dimension)** |
| SalaryID | EmployeeID | DateKey |
| EmployeeID | EmployeeName | FullDate |
| DateKey | DepartmentName | MonthName |
| SalaryAmount | DepartmentLocation | Year |
| JobTitle | HireDate |  |

**What Changed?**

* The SalaryFact table now only contains key values (EmployeeID, DateKey) and the numerical SalaryAmount.
* The EmployeeDim table holds all the descriptive information about each employee. If Alice's job title or department changes, we update it in a single place.

**Example 2: Financial Transactions**

**The Problem:** You have a log of financial transactions, and you want to analyze them by account, transaction type, and date. A single table would be very slow to query.

The Initial, Normalized Data (not ideal for Power BI):

| TransactionID | Date | AccountID | AccountOwner | TransactionType | TransactionDescription | Amount |

| :-------------- | :----------- | :---------- | :------------- | :---------------- | :----------------------- | :------- |

| 2001 | 2023-05-01 | 9876 | Sam | Withdrawal | ATM Withdrawal | $50 |

| 2002 | 2023-05-02 | 5432 | Jane | Deposit | Check Deposit | $200 |

| 2003 | 2023-05-03 | 9876 | Sam | Transfer | Transfer to Savings | $100 |

**The Star Schema (better for Power BI):**

Here, the central fact is the transaction itself.

|  |  |  |  |
| --- | --- | --- | --- |
| **TransactionFact Table (Fact)** | **AccountDim Table (Dimension)** | **TransactionTypeDim Table (Dimension)** | **DateDim Table (Dimension)** |
| TransactionID | AccountID | TypeID | DateKey |
| AccountID | AccountOwner | TransactionType | FullDate |
| TypeID | Address | TransactionDescription | Quarter |
| DateKey |  |  | Year |
| Amount |  |  |  |

**What Changed?**

* The TransactionFact table only has numerical data (Amount) and foreign keys to link to the descriptive tables.
* The descriptive information has been pulled into separate dimension tables (AccountDim, TransactionTypeDim, DateDim).
* This makes it easy to ask questions like, "What was the total withdrawal amount for the account owned by Sam in the third quarter of 2023?"

**Key Takeaways for a Beginner:**

* **Separate the "What happened" from the "Who, What, Where, When."**
* The **"What happened"** (the numbers you want to analyze) goes into your **Fact Table**.
* The **"Who, What, Where, When"** (the descriptive details) go into your **Dimension Tables**.
* This separation makes your Power BI reports run faster and makes the data model easier to understand and use.