**Table of Contents**

[ **Link to the CHAT GPT Prompt Page:** 2](#_Toc170566048)

[ **Link to DAX Learn Microsoft Page:** 2](#_Toc170566049)

[ **What’s the dataset all about?** 2](#_Toc170566050)

[ **What is a Fact Table and what information does it contain?** 2](#_Toc170566051)

[Fact Table 2](#_Toc170566052)

[Summary 2](#_Toc170566053)

[ **What is a bridge table and what is its use?** 3](#_Toc170566054)

[Bridge Table 3](#_Toc170566055)

[Example: dbo.FactInternetSalesReason 3](#_Toc170566056)

[How it Works 3](#_Toc170566057)

[Example Data 4](#_Toc170566058)

[Query Example 4](#_Toc170566059)

[Result: 4](#_Toc170566060)

[Summary 5](#_Toc170566061)

[ **What is a stare schema and what other kinds of schemas exists?** 5](#_Toc170566062)

[Star Schema 5](#_Toc170566063)

[Snowflake Schema 5](#_Toc170566064)

[Galaxy Schema (Fact Constellation) 6](#_Toc170566065)

[Summary of Schemas 7](#_Toc170566066)

[ **What is the ETL process and how to perform it?** 7](#_Toc170566067)

[ETL Process 7](#_Toc170566068)

[ETL Implementation Example 7](#_Toc170566069)

[Summary 8](#_Toc170566070)

# **Link to the CHAT GPT Prompt Page:**

https://chatgpt.com/share/391649fe-9f29-4dfc-9b68-f043f05130e4

Note that the model used to interact with CHAT GTP was the 4o

# **Link to DAX Learn Microsoft Page:**

[Αναφορά παραστάσεων ανάλυσης δεδομένων (DAX) - DAX | Microsoft Learn](https://learn.microsoft.com/el-gr/dax/)

# **What’s the dataset all about?**

**Origin of Data**

The data within AdventureWorksDW2022 is fictional but structured to mimic real-world business scenarios. The AdventureWorks company represented in the database is a fictional bicycle manufacturing company that sells products through multiple channels, including internet sales and resellers.

# **What is a Fact Table and what information does it contain?**

Fact Table

A fact table is a central table in a star schema of a data warehouse. It contains the quantitative data for analysis and typically consists of:

* **Measures (or Facts):** These are the numeric data points that you want to analyze. Examples include sales amount, order quantity, profit, etc.
* **Foreign Keys:** These keys link the fact table to dimension tables. Each foreign key corresponds to a primary key in a dimension table.

**Characteristics of Fact Tables:**

* They store large volumes of data.
* They are usually normalized to reduce redundancy.
* They often contain transactional data.

**Example:** In a sales data warehouse, a fact table might be **FactInternetSales** with columns like SalesAmount, OrderQuantity, and foreign keys like CustomerID, ProductID, and DateID.

Summary

* **Fact Tables** store quantitative data for analysis, such as sales figures or transaction counts.
* **Dimension Tables** provide descriptive context to the facts, such as customer details or product information.
* **Metric Tables** (if used) store pre-calculated metrics or aggregated data for quick access in reporting.

# **What is a bridge table and what is its use?**

A bridge table, also known as a helper table or a link table, is used in data warehousing to handle many-to-many relationships between dimension tables and fact tables. In scenarios where a single fact can be associated with multiple dimension records, a bridge table becomes essential.

### Bridge Table

A bridge table is used to resolve many-to-many relationships and often consists of foreign keys from the related tables along with any additional attributes necessary to describe the relationship.

### Example: dbo.FactInternetSalesReason

In the AdventureWorksDW2022 database, the dbo.FactInternetSalesReason table is an example of a bridge table. It handles the many-to-many relationship between sales transactions and sales reasons. Here’s a breakdown of its role and structure:

#### Context

In a sales data warehouse, each sales transaction (fact) can have multiple associated reasons (dimension). For instance, a customer might buy a product due to multiple promotional reasons, such as discounts and special offers.

#### Structure

The dbo.FactInternetSalesReason bridge table typically includes:

* **Fact Table Foreign Key:** This links to the fact table (FactInternetSales), representing a sales transaction.
* **Dimension Table Foreign Key:** This links to the dimension table (DimSalesReason), representing the reason for the sale.

#### Schema Example

The schema of FactInternetSalesReason might look like this:

* **SalesOrderID (FK):** Foreign key linking to the FactInternetSales table.
* **SalesReasonID (FK):** Foreign key linking to the DimSalesReason table.

### How it Works

Let’s consider the relationship between FactInternetSales and DimSalesReason:

1. **Fact Table (FactInternetSales):**
   * Contains individual sales transactions.
   * Columns might include SalesOrderID, SalesAmount, CustomerID, ProductID, DateID.
2. **Dimension Table (DimSalesReason):**
   * Contains reasons for sales.
   * Columns might include SalesReasonID, ReasonDescription (e.g., "Discount", "Advertisement", "Word of Mouth").
3. **Bridge Table (FactInternetSalesReason):**
   * Links FactInternetSales to DimSalesReason.
   * Columns include SalesOrderID (from FactInternetSales) and SalesReasonID (from DimSalesReason).

### Example Data

Here’s a simplified example of how the tables might be populated:

**FactInternetSales:**

| **SalesOrderID** | **SalesAmount** | **CustomerID** | **ProductID** | **DateID** |
| --- | --- | --- | --- | --- |
| 1 | 100.00 | 101 | 201 | 202301 |
| 2 | 150.00 | 102 | 202 | 202301 |

**DimSalesReason:**

| **SalesReasonID** | **ReasonDescription** |
| --- | --- |
| 1 | Discount |
| 2 | Advertisement |
| 3 | Word of Mouth |

**FactInternetSalesReason:**

| **SalesOrderID** | **SalesReasonID** |
| --- | --- |
| 1 | 1 |
| 1 | 2 |
| 2 | 3 |

### Query Example

To find out all the reasons associated with a specific sales order, you can join these tables together:

SELECT

f.SalesOrderID,

f.SalesAmount,

r.ReasonDescription

FROM

FactInternetSales f

JOIN

FactInternetSalesReason br ON f.SalesOrderID = br.SalesOrderID

JOIN

DimSalesReason r ON br.SalesReasonID = r.SalesReasonID

WHERE

f.SalesOrderID = 1;

### Result:

| **SalesOrderID** | **SalesAmount** | **ReasonDescription** |
| --- | --- | --- |
| 1 | 100.00 | Discount |
| 1 | 100.00 | Advertisement |

### Summary

* **Bridge Table (FactInternetSalesReason):** Manages the many-to-many relationship between sales orders and sales reasons.
* **Purpose:** Ensures that each sales order can be linked to multiple reasons and each reason can be associated with multiple sales orders.
* **Structure:** Contains foreign keys from both related tables (FactInternetSales and DimSalesReason).

Bridge tables are crucial in complex data models where relationships between entities are not straightforwardly one-to-one or many-to-one, allowing for a more flexible and normalized design.

# **What is a stare schema and what other kinds of schemas exists?**

In data warehousing, schemas are blueprints for how data is organized and related. The most common types of schemas are the star schema, snowflake schema, and galaxy schema (or fact constellation). Here’s an explanation of each:

### Star Schema

**Star schema** is the simplest type of data warehouse schema. It consists of a central fact table surrounded by dimension tables. The layout resembles a star, hence the name.

**Characteristics:**

* **Fact Table:** Contains quantitative data (measures) for analysis and foreign keys to dimension tables.
* **Dimension Tables:** Contain descriptive attributes related to the facts, connected to the fact table via foreign keys.
* **Denormalization:** Dimension tables are typically denormalized to reduce the complexity of joins and improve query performance.

**Advantages:**

* Simple and easy to understand.
* Fast query performance due to fewer joins.

**Disadvantages:**

* Redundancy in dimension tables can lead to increased storage requirements.

**Example:** For a sales data warehouse:

* **Fact Table:** FactSales (SalesAmount, OrderQuantity, CustomerID, ProductID, DateID)
* **Dimension Tables:** DimCustomer (CustomerID, Name, Address), DimProduct (ProductID, ProductName, Category), DimDate (DateID, Date, Month, Year)

### Snowflake Schema

**Snowflake schema** is a more complex type of schema that normalizes the dimension tables into multiple related tables.

**Characteristics:**

* **Fact Table:** Same as in the star schema, containing measures and foreign keys.
* **Normalized Dimension Tables:** Dimension tables are split into related tables, reducing redundancy but increasing the number of joins needed.

**Advantages:**

* Reduces data redundancy and saves storage space.
* Can lead to a more organized and flexible design.

**Disadvantages:**

* More complex queries with additional joins.
* Slightly slower query performance compared to star schema due to the normalization.

**Example:** Continuing the sales data warehouse example:

* **Fact Table:** FactSales (SalesAmount, OrderQuantity, CustomerID, ProductID, DateID)
* **Normalized Dimension Tables:**
  + DimCustomer (CustomerID, Name, AddressID)
  + DimAddress (AddressID, Address, City, Country)
  + DimProduct (ProductID, ProductName, CategoryID)
  + DimCategory (CategoryID, Category)
  + DimDate (DateID, Date, Month, Year)

### Galaxy Schema (Fact Constellation)

**Galaxy schema** (also known as a fact constellation) is a more complex schema that contains multiple fact tables sharing dimension tables. It’s used to represent more complex data relationships and support multiple business processes.

**Characteristics:**

* **Multiple Fact Tables:** Represent different business processes or events.
* **Shared Dimension Tables:** Common dimension tables are shared across multiple fact tables.
* **Flexibility:** Can accommodate complex business scenarios.

**Advantages:**

* Can handle complex business requirements.
* Reduces redundancy by sharing dimensions.

**Disadvantages:**

* More complex design and maintenance.
* Potentially slower query performance due to multiple fact tables.

**Example:** Combining sales and inventory data:

* **Fact Tables:**
  + FactSales (SalesAmount, OrderQuantity, CustomerID, ProductID, DateID)
  + FactInventory (InventoryQuantity, ProductID, DateID, WarehouseID)
* **Shared Dimension Tables:** DimCustomer, DimProduct, DimDate
* **Additional Dimension Tables:** DimWarehouse (WarehouseID, WarehouseName, Location)

### Summary of Schemas

1. **Star Schema:**
   * Simple and intuitive.
   * Fact table surrounded by denormalized dimension tables.
   * Optimized for query performance but with redundancy.
2. **Snowflake Schema:**
   * More normalized version of the star schema.
   * Dimension tables are split into related tables.
   * Reduces redundancy but requires more complex joins.
3. **Galaxy Schema (Fact Constellation):**
   * Supports multiple fact tables.
   * Shared dimension tables among facts.
   * Suitable for complex business scenarios but more complex to design and maintain.

Understanding these schema types helps in designing efficient data models that suit specific business requirements and optimize data retrieval for analysis and reporting.

# **What is the ETL process and how to perform it?**

ETL (Extract, Transform, Load) is a process used in data warehousing to collect data from various sources, transform it into a format suitable for analysis, and load it into a target data warehouse. Here's a detailed explanation of each step and an example of how to build an ETL procedure.

### ETL Process

1. **Extract:**
   * The extraction phase involves collecting data from various source systems, which can include databases, flat files, APIs, or other data streams.
   * The goal is to retrieve all the necessary data without affecting the performance of the source systems.
2. **Transform:**
   * The transformation phase involves cleaning, validating, and converting the extracted data into a suitable format for analysis.
   * This may include operations such as filtering, aggregating, joining, sorting, and applying business rules.
   * Data is also transformed to ensure consistency, handle missing values, and enforce data quality.
3. **Load:**
   * The loading phase involves inserting the transformed data into the target data warehouse or data mart.
   * This step can include inserting new data, updating existing data, and handling data deletions.
   * The load process must ensure data integrity and consistency in the target system.

### ETL Implementation Example

Using a hypothetical ETL tool, here's how the process might look:

1. **Extract:**

-- Extract from SQL database

SELECT \* FROM store1\_sales;

-- Extract from CSV file (using a script or ETL tool function)

LOAD DATA INFILE 'store2\_sales.csv' INTO TABLE temp\_store2\_sales;

-- Extract from REST API (using a script)

CALL extract\_from\_api('https://api.store3.com/sales', 'temp\_store3\_sales');

1. **Transform:**

-- Standardize date format and clean data

INSERT INTO transformed\_sales

SELECT StoreID,

STR\_TO\_DATE(SaleDate, '%Y-%m-%d') AS SaleDate,

ProductID,

SUM(SaleAmount) AS TotalSales,

AVG(SaleAmount) AS AverageSaleAmount

FROM temp\_sales

WHERE SaleAmount IS NOT NULL

GROUP BY StoreID, SaleDate, ProductID;

1. **Load:**

-- Load data into the central data warehouse

INSERT INTO consolidated\_sales (StoreID, SaleDate, ProductID, TotalSales, AverageSaleAmount)

SELECT StoreID, SaleDate, ProductID, TotalSales, AverageSaleAmount

FROM transformed\_sales;

### Summary

* **ETL (Extract, Transform, Load):** A process to collect data from various sources, transform it, and load it into a target system.
* **Steps:** Extract data from sources, transform it to meet business requirements, and load it into a data warehouse.
* **Tools:** Use ETL tools (e.g., SSIS, Talend) or custom scripts to automate the process.
* **Example:** Integrating sales data from multiple stores into a central data warehouse involves extracting data from databases, files, and APIs, transforming it for consistency and accuracy, and loading it into a consolidated table for analysis.