Dimensional Modeling for Data Warehouses

Vijaya Durga N · Follow

6 min read · Nov 2, 2023

Image by AcatXlo • So long, and thanks for all the likes! from Pixabay

Introduction

The data landscape has transformed dramatically, and organisations are dealing with an unprecedented volume of data. To navigate this data deluge efficiently, businesses have turned to data warehousing as a central element of their data strategy. In this blog, we will explore the design principles of data warehouses and the foundational concepts including dimensionality, fact tables, dimension tables, star schema, and snowflake schema.

Centralised Data Warehouse

In the era of big data, having data spread across different systems can lead to inefficiencies and hinder the ability to derive valuable insights. Centralising data in a data warehouse is a fundamental design principle. A centralised data warehouse:

- Aggregates Data: It consolidates data from various sources, making it accessible from a single location.
- Simplifies Access: Users can query and analyse data without having to access multiple data sources.
- Ensures Data Consistency: Data in a centralised warehouse is standardised and consistent.
- Supports Data Governance: Centralised storage simplifies data governance and ensures data quality.

Dimensional Analysis for a Data Warehouse

Dimensional modelling is a critical aspect of data warehouse design because it helps structure and organise data in a way that is optimised for query and reporting purposes. It is a method of designing the schema of a data warehouse to support efficient and intuitive data retrieval, analysis, and reporting.

Principles of Dimensionality

Understanding dimensionality is essential for effective data warehouse design. Here are some key concepts to know:

Facts

Facts are numeric metrics, the measurements or data you want to analyse. Key points about facts:

- Examples include sales revenue, quantities sold, and profit margins.
- Facts are typically stored in fact tables.

Fact Tables

Fact tables store facts and are central to a data warehouse. Best practices for fact tables include:

- Choosing the right granularity.
- Designing relationships with dimension tables.

- Ensuring that fact tables are narrow, with fewer columns.

Dimension

Dimensions are used to categorise, filter, and group facts. Key points about dimensions:

- Examples include time, location, product, and customer.
- Dimensions are stored in dimension tables.

Dimension Tables

Dimension tables contain attributes and hierarchies that help organise data. Best practices for dimension tables include:

- Creating dimension hierarchies to make data navigation easier.
- Including attributes that provide context.
- Avoiding repeated data by using surrogate keys.

We represent the fact tables and dimension table either in star schema or in snowflake schema

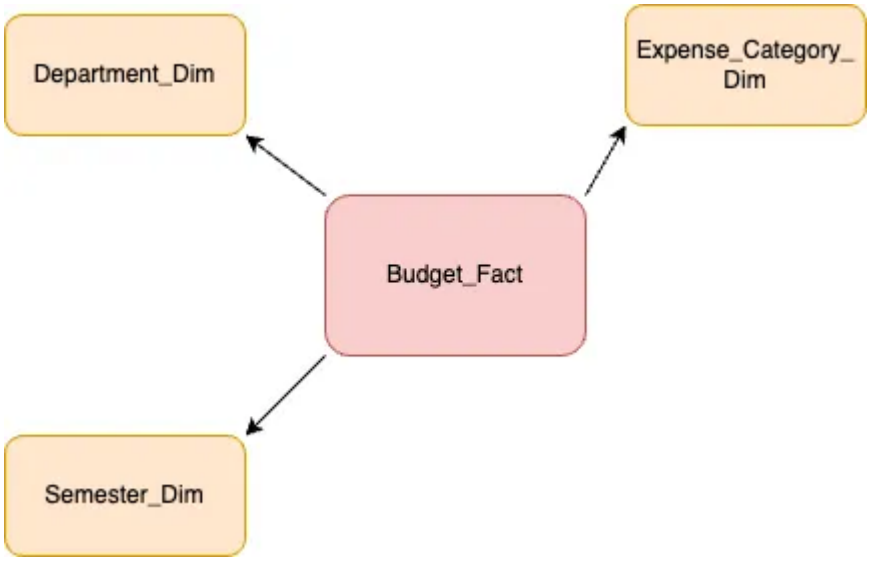
Star Schema

The star schema is a popular design approach for data warehousing. It features a centralised fact table and dimension tables connected to the fact table. Key characteristics of the star schema:

1. Simplicity: The star schema is a simpler and more denormalized structure. It consists of a central fact table connected to dimension tables. The fact table contains measures (numeric data) and foreign keys that link to the dimension tables.
2. Performance: Star schemas are generally easier to query and provide better query performance, especially for simple and common analytical queries. This simplicity makes them suitable for data warehousing environments where query performance is critical.

3. Redundancy: The star schema may involve some redundancy in the dimension tables, but this redundancy can help improve query performance by reducing the number of joins required.

4. Maintenance: It is generally easier to maintain and update a star schema because of its simpler structure.



Only one level away from fact table along each hierarchy

Snowflake Schema

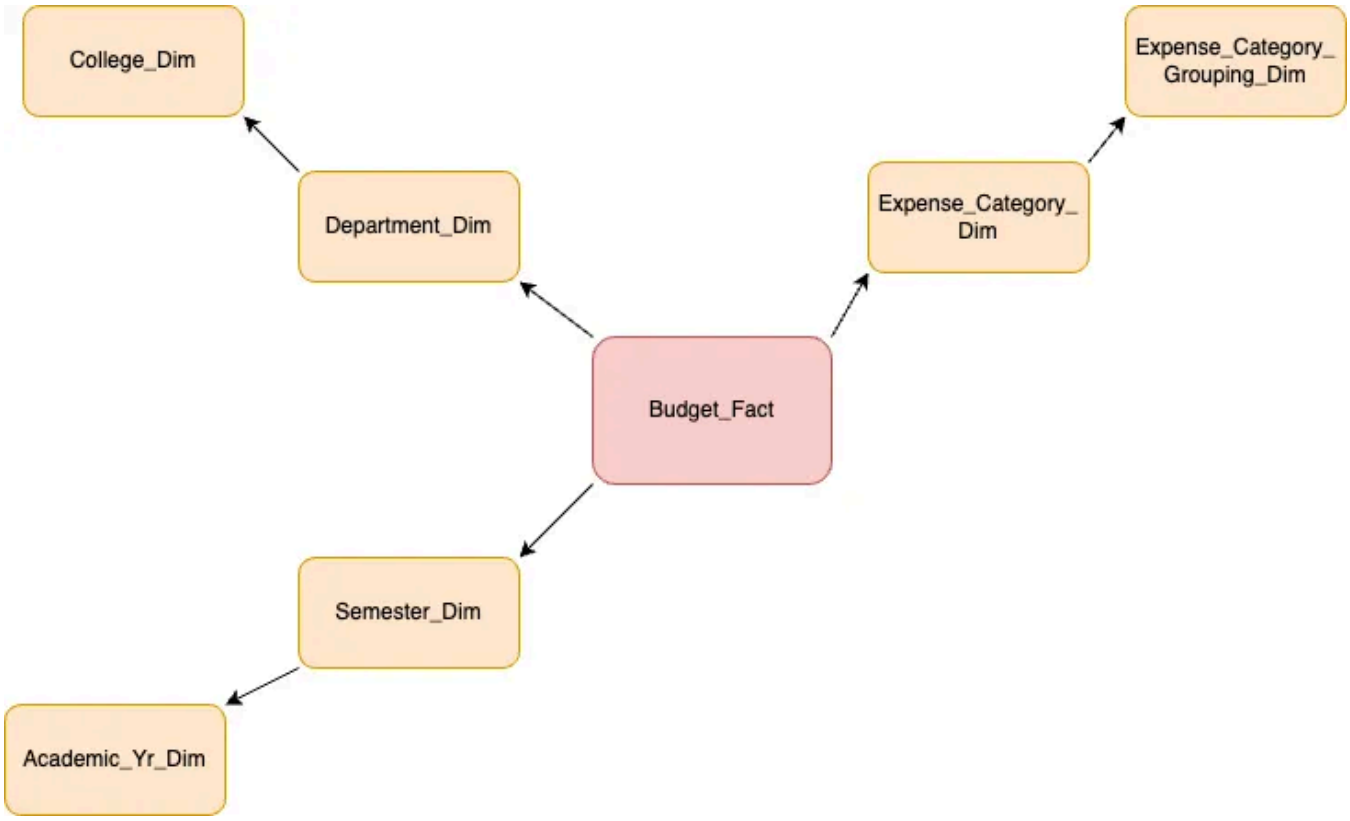
The snowflake schema is an extension of the star schema that normalises dimension tables. Key points about the snowflake schema:

1. Normalisation: The snowflake schema is a more normalized structure compared to the star schema. In a snowflake schema, dimension tables are further divided into sub-dimensions or related tables, which reduces redundancy.

2. Storage Efficiency: Snowflake schemas are more storage-efficient due to their normalisation. However, this may come at the cost of increased complexity in query construction and potentially slower query performance.

3. Data Integrity: Snowflake schemas can maintain data integrity more rigorously because of the normalisation. Updates and changes to data are easier to manage and maintain in a snowflake schema.

4. Complexity: Querying a snowflake schema can be more complex and may require more joins, which can impact query performance for certain types of queries.



One or more levels away from fact table along each hierarchy

Retail Sales analysis Use case:

Imagine you are building a data warehouse for a retail company. You want to analyse and report on various aspects of your sales data. Here's how you would apply the concepts of facts and dimensions in this context:

Fact: Sales Amount, Profit, Quantity Sold, and Discount Amount

We represent the measurement in fact table and context in dimension table and design relationship using surrogate keys in the fact tables and primary keys in dimension tables

Fact Table ("Sales" Fact Table):

Columns:

> *Date_ID (foreign key to the "Time" dimension)*
>
> *Product_ID (foreign key to the "Product" dimension)*
>
> *Store_ID (foreign key to the "Store" dimension)*

> *Sales_Amount*
>
> *Profit*
>
> *Quantity_Sold*
>
> *Discount_Amount*

Dimension Tables:

Time Dimension:

- Attributes: Date, Month, Quarter, Year, etc.

Product Dimension:

- Attributes: Product ID, Product Name, Category, Manufacturer, etc.

Store Dimension:

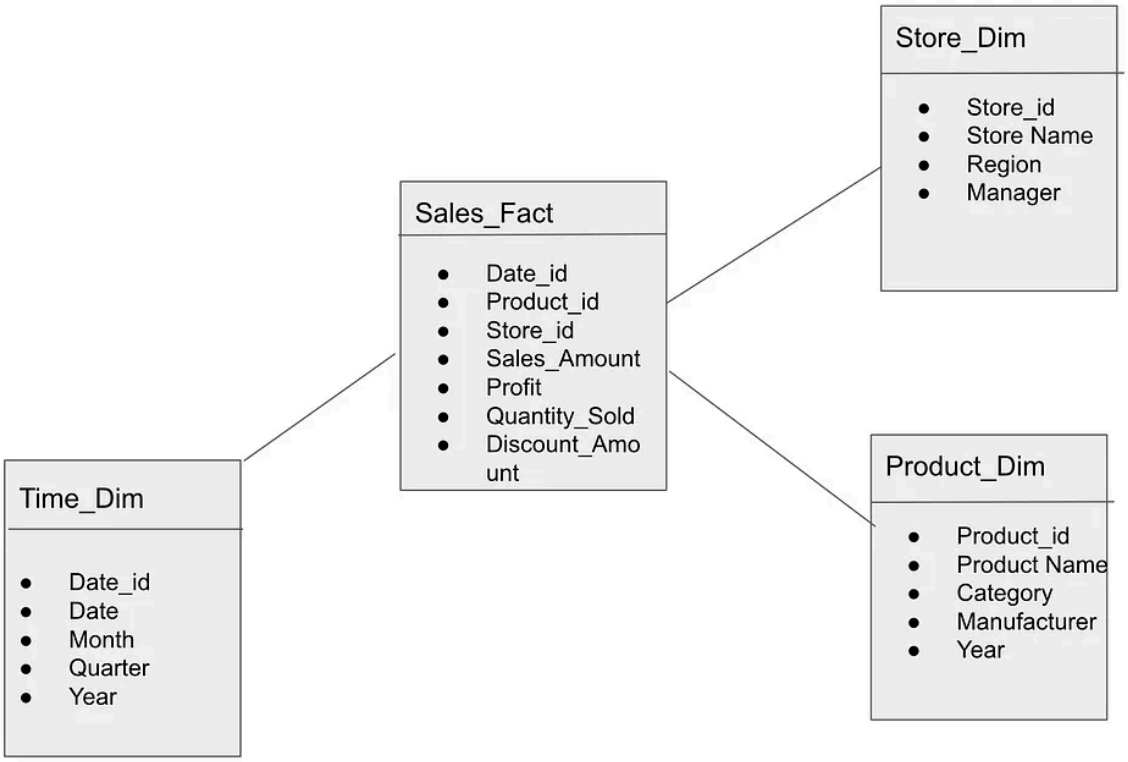
- Attributes: Store ID, Store Name, Region, Manager, etc.

Now, let's compare the Star Schema and Snowflake Schema for this use case:

Star Schema:

In a Star Schema, you have a central fact table directly connected to dimension tables. It's a denormalised structure, and it simplifies querying and reporting.

- Fact Table ("Sales") is directly linked to Time, Product, and Store dimensions.



- The relationships are straightforward, and queries are simple to write and understand.
- Example Query:

> *"What was the total sales amount in the 'Electronics' category for 'Q3 2023' in 'Store A'?"*

Snowflake Schema

In Snowflake Schema, dimension tables may be normalised by creating sub-dimensions, resulting in more complex relationships.

- Time Dimension could be split into separate tables like "Date," "Month," "Quarter," and "Year."
- Product Dimension might include additional tables for "Category" and "Manufacturer."
- Store Dimension might be normalised into sub-dimensions like "Region" and "Manager."

Example Query (Snowflake Schema):

> *"What was the total sales amount in the 'Electronics' category for 'Q3 2023' in 'Store A'?"*

To execute this query in a Snowflake Schema, you would need to join more tables compared to the Star Schema. While Snowflake Schema can save storage space and ensure data integrity due to normalization, it usually requires more complex queries and may result in slightly slower performance for certain queries compared to a Star Schema.

In summary:

- Use a star schema when query performance and simplicity are essential, and data redundancy is acceptable.
- Use a snowflake schema when data integrity, storage efficiency, and data update/insert/delete operations are more critical, and you can manage the increased complexity in query construction.

The choice between these two schema types should be based on your specific business requirements, data characteristics, and the performance demands of your data warehousing or database environment.

Best Practices

Implementing a data warehouse is a significant undertaking. To ensure success, consider these best practices:

- Data Quality: Ensure data quality and accuracy to derive meaningful insights.
- Performance Tuning: Optimise database performance through indexing and query optimisation.
- Data Governance: Establish data governance policies and procedures.
- Scalability: Design the data warehouse to handle increasing data volumes.
- Security: Implement strong security measures to protect sensitive data.
- Documentation: Maintain comprehensive documentation for data definitions, transformations, and business rules.

Conclusion

Whether you choose the star schema, snowflake schema, or a combination of both, it’s important to align your data warehouse design with your specific business

needs. Building and maintaining a data warehouse that adheres to best practices is a significant step toward data-driven success.

The design principles of data warehouses and best practices outlined here serve as a foundation for building a robust, efficient, and effective data warehousing solution. By centralising data, understanding dimensionality, and employing appropriate schema design, organisations can unlock valuable insights from their data.

Data Warehouse | Database Design | Data Engineering | Data | Data Warehousing

Follow

Written by Vijaya Durga N

14 Followers · 3 Following

Data Engineer

Responses (1)

What are your thoughts?

Respond

T

Thomas

Feb 22, 2024

Appreciating the article, I also want to spotlight an insightful guide dedicated to data warehouse best practices. It delves into effective strategies and tips crucial for enhancing data warehouse functionality and overall performance, providing a... more