Database Design Choice

Project Title: Analyzing Business Opportunities in Global Superstore Data

Zhile Wu, DBA Team

Objective

The objective of this project was to analyze business performance across various operational dimensions using structured data extracted from the Global Superstore dataset. Specifically, I aimed to:

- Identify high-performing product categories and customer segments
- Understand how discounting affects profitability
- Highlight trends in regional sales and shipping performance
- Enable flexible querying for different business questions

To accomplish this, I needed to convert a flat CSV file into a relational database that supports efficient, scalable analysis through SQL. This required careful normalization and schema design to reduce redundancy, preserve relationships between data entities, and prepare the dataset for advanced querying, including aggregations, groupings, and joins.

Dataset Overview

- Source: Global Superstore Dataset (Kaggle)
- Link of the dataset: https://drive.google.com/file/d/1J0ZNE96Mx_qcbnZcgzPP9PZHdlWNsHy1/view?usp=sharing
- Size: around 500 rows sampled
- Attributes Covered: Orders, Products, Subcategories, Regions, Ship Modes, Customers, Discounts, Sales, Profit, and Dates

Normalization Goals

To reduce redundancy, improve data integrity, and support efficient joins, the dataset was normalized to 3rd Normal Form (3NF) across multiple tables. Key goals included:

- Eliminating repeating data (e.g., subcategory names, regions)

- Separating facts (e.g., orders, profit) from reference data (e.g., product names)
- Creating primary/foreign key relationships for structured queries

Final Tables Introductions

Primary Table: OrderDetails

The primary table in the database is orderdetails. It stores all transaction-level facts, including:

- Quantity of products sold
- Sales revenue and discounts applied
- Profit generated for each item

OrderDetails acts as the central fact table in the schema. All analysis ultimately stems from it, as it connects to the:

1. orders table (for order metadata, like date and region)
2. products table (for product and subcategory info)
3. customers table (for customer identity and segment)

Because it brings together multiple business dimensions (product, customer, location, discount), it is the anchor point for querying and analysis. Nearly every insight in this project was generated using orderdetails as the starting point.

Table Relationships

Here's an overview of key table relationships:

Table	Connected To	Relationship
orderdetails	orders	Many-to-one (many orderdetails per order)
orderdetails	products	Many-to-one (each detail line links to one product)
products	subcategories	Many-to-one (each product has one subcategory)
orders	customers	Many-to-one (each order belongs to a customer)
orders	regions	Many-to-one (region of sale)

orders	shipmodes	Many-to-one (shipping method used)

These relationships were implemented via foreign keys, with meaningful, normalized keys (product_id, order_id, etc.) to enforce referential integrity and enable complex joins across dimensions.

Tables and Their Roles

Table Name	Purpose
orderdetails	Stores line-item sales transactions with quantity, discount, sales, and profit
orders	Stores general order data, including order ID, order date, region, customer, and shipping info
products	Stores product name and associated subcategory
subcategories	Stores subcategory name and optional parent category
customers	Stores customer name and segment (e.g., Consumer, Corporate)
regions	Stores region names used for filtering geographic trends
shipmodes	Stores shipping types used in each order (e.g., Standard Class)

Design Decisions: What Was Omitted and Why

I made deliberate decisions to omit or condense certain fields from the original dataset:

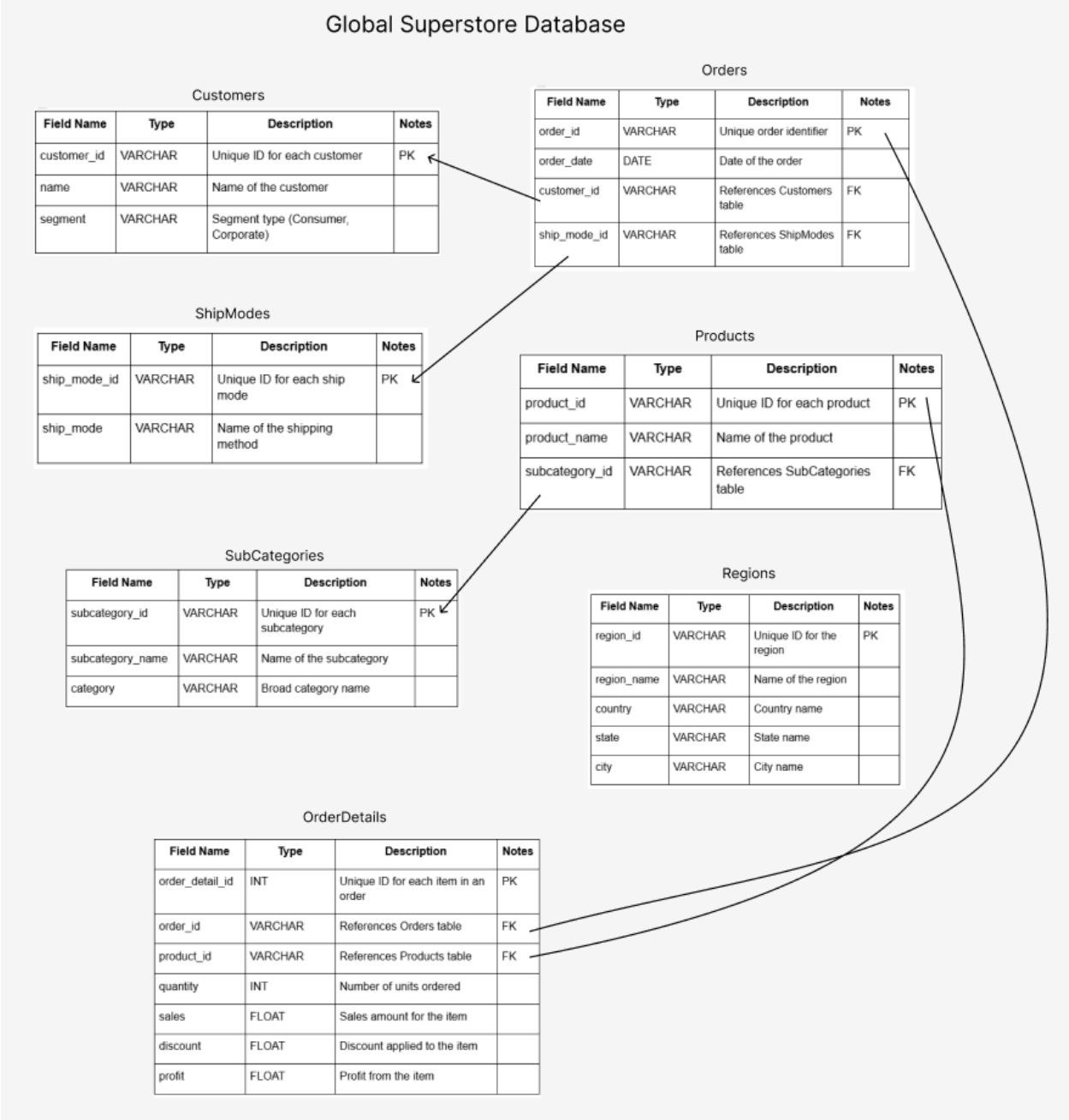
Omitted Fields:

- Full address fields (City, State, Postal Code):
  → These were excluded due to limited relevance in profitability analysis and a lack of sufficient regional granularity in the sample size.
- Row ID and Unused Codes:
  → Internal row identifiers and legacy tracking fields were excluded since they did not contribute to any key relationships or insights.

- Ship Date:
  → Only order_date was retained to keep the schema concise. In this project, the timing of the order, not the shipment, was relevant to the trend analysis.

These omissions helped me focus on fields that were analytically meaningful while maintaining a streamlined and performance-optimized schema.

ERD Diagram



Design Highlights

Here are the key highlights of the database design, aligned with best practices for analytical use cases:

1. Third Normal Form (3NF) Structure

- The schema was normalized to avoid duplication and ensure that each data point is stored in exactly one place.
- For example, subcategory names were extracted into their own table and linked via foreign keys to the products table.

2. Clear Separation of Facts and Dimensions

- Transaction-level data (e.g., quantity, discount, profit) is stored in the orderdetails table.
- Reference data (e.g., product names, customer segments, ship modes) is stored in dimension tables (products, customers, etc.).
- This design enables star-schema-like analytical queries, common in business intelligence environments.

3. Efficient Join Paths

- Every table has clearly defined primary and foreign keys to allow fast and reliable joins.
- Analysts can easily join orders to products, customers, shipping methods, and regions with minimal complexity.

4. Support for Diverse Business Questions

- The schema supports a wide range of analytical queries, such as:
    - “Which subcategories bring the most profit?”
    - “Which customer segments drive the most value?”
    - “At what discount levels do we start losing money?”
- These questions would be cumbersome or error-prone to answer using a flat table.

5. Scalability & Extensibility

- The design can accommodate future business needs, such as supplier info, returns, or new product categories — by adding tables without breaking existing queries.
- This makes the database a strong foundation for long-term analytics.

Conclusion

My database design choices reflect a structured, scalable approach to transforming raw sales data into a powerful decision-support tool. It aligns with best practices in relational modeling and enables data-driven insights that directly support business recommendations.