**2nd Case Name :** Data Modelling

**Case Objective :** Design a data warehouse for e-commerce platform to track its sales and operations performance

**Task and Constraints:**

* Design star schema within the data warehouse
* Handle slowly changing dimensions tables
* Performace optimization (indexing and partitioning strategy, handle large scale data)
* Provide ERD or schema design within the warehouse (relationships between tables, key attributes and constraints)

**My approach to solve this case:**

1. **Understanding the business process of e-commerce platform**

Before designing a data warehouse, it's crucial to first understand the business process clearly. This understanding allows me to convert business requirements into specific design requirements. The image below illustrates a typical business process in which a buyer purchases a product on an e-commerce platform

Figure 1 - Common e-commerce business process

1. **Determine data design pattern for the Data Warehouse**

I propose adopting **medallion architecture** that organizes data into layers to progressively enhance data quality and structure. The layers within the warehouse include:

* Bronze layer 🡪 Contains raw data directly ingested from the original sources without transformations.
* Silver Layer 🡪 Holds data that has been filtered, cleaned, and enriched.
* And Gold layers 🡪 Provides business-ready aggregated data structured in a star schema for analytics purposes.

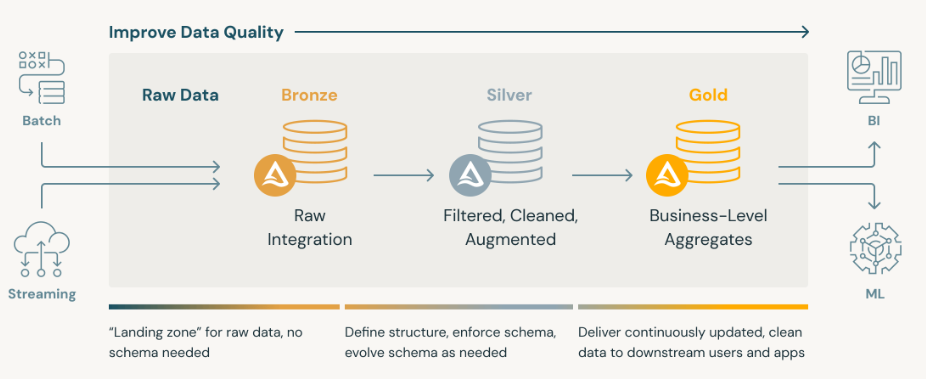


Figure 2 - Medallion Architecture

1. **Design a star schema within the gold layer in data warehouse**

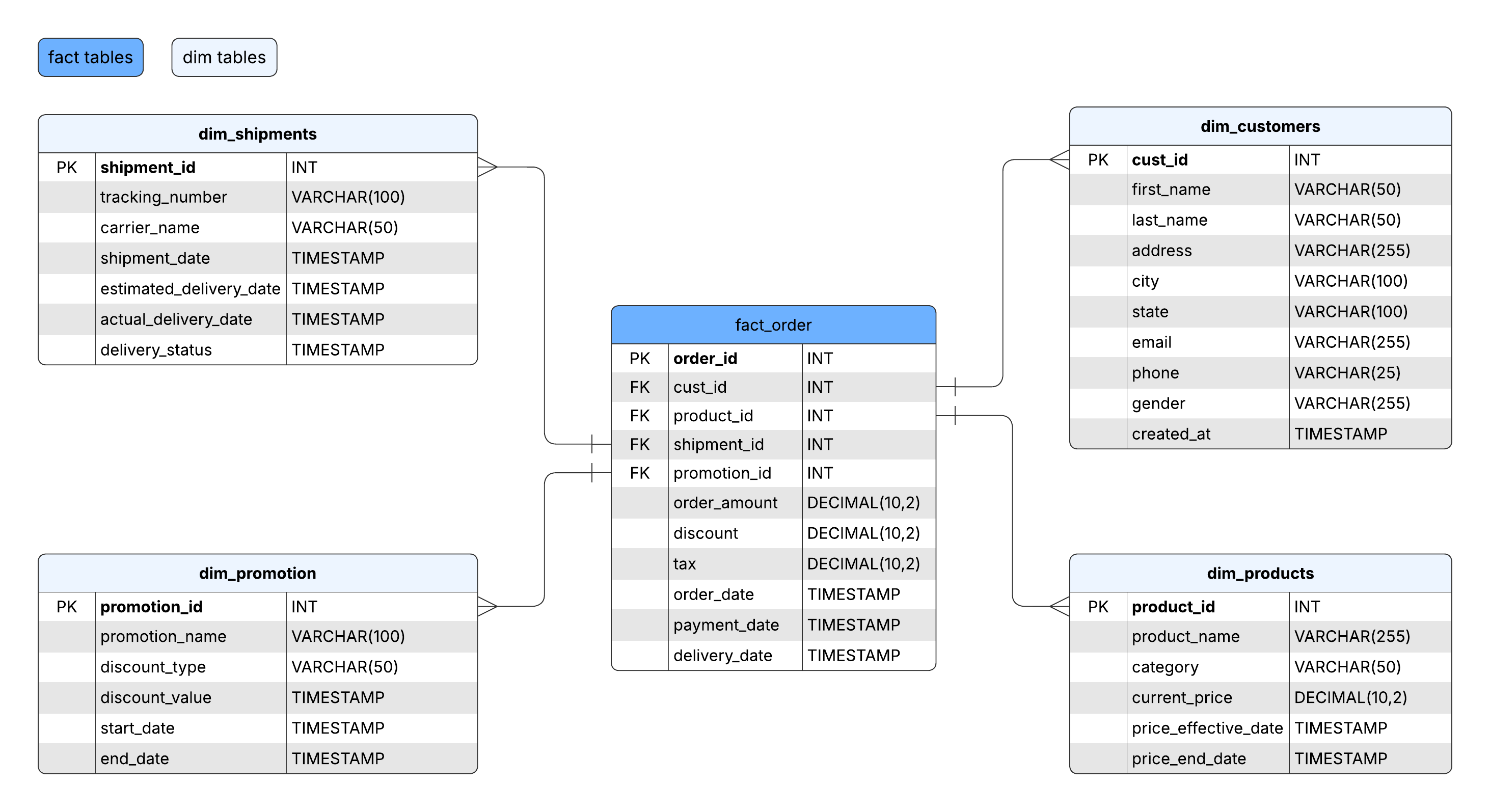


Figure 3 - Star Schema Diagram

1. **Handling slowly changing dimension tables** 
   1. **Product price in dim\_products (SCD Type 2)**
      * **Rationale:** Product price may change overtime, it’s important to track historical changes for analysis
      * **Implementation:**
        + Add price\_effective\_date and price\_end\_date to dim\_products
        + When a price changes, inserts a new row with the **price\_effective\_date** becomes the **price\_ends\_date** in the previous row
   2. **Customer details in dim\_customers (SCD Type 1)**
      * **Rationale:** Changes in customer data may change overtime but it may not important to track historical changes for analysis
      * **Implementation:** Update the existing row in dim\_customers with the new details
2. **Proposed partitioning strategy and indexing**

I propose to use partitioning and indexing for all fact and dim tables within the data warehouse for faster data access, and better organization and management of large datasets. The strategy:

1. **Indexing**
   * Fact table: create indexes based on foreign key (cust\_id, product\_id, etc)
   * Dim table: create indexes based on frequent queried attributes (product\_name, etc)
2. **Partitioning**
   * Fact table: partition data based on order\_date column
   * Dim table: partition data based on category (for dim\_product) or gender in dim\_customer

1. **Optimization strategy for performance**

I propose to implement **Data Archiving** to archivine old data every month changes to a separate table to keep the fact table manageable.