Dimension Modelling of Invoice Data

1. Data Profiling and Inference Process

- Overview The dataset provided contains transactional sales data with attributes related to customers, products, stores, timestamps, and dates. Based on this, a dimensional model was designed to facilitate efficient querying and analytical processing.
- Identified key attributes: Customer name, product name, store ID, invoice date, quantity, unit price, total sales, and date components.
- Examined data types and checked for inconsistencies.
- Conducted exploratory analysis to identify missing values and duplicate records.

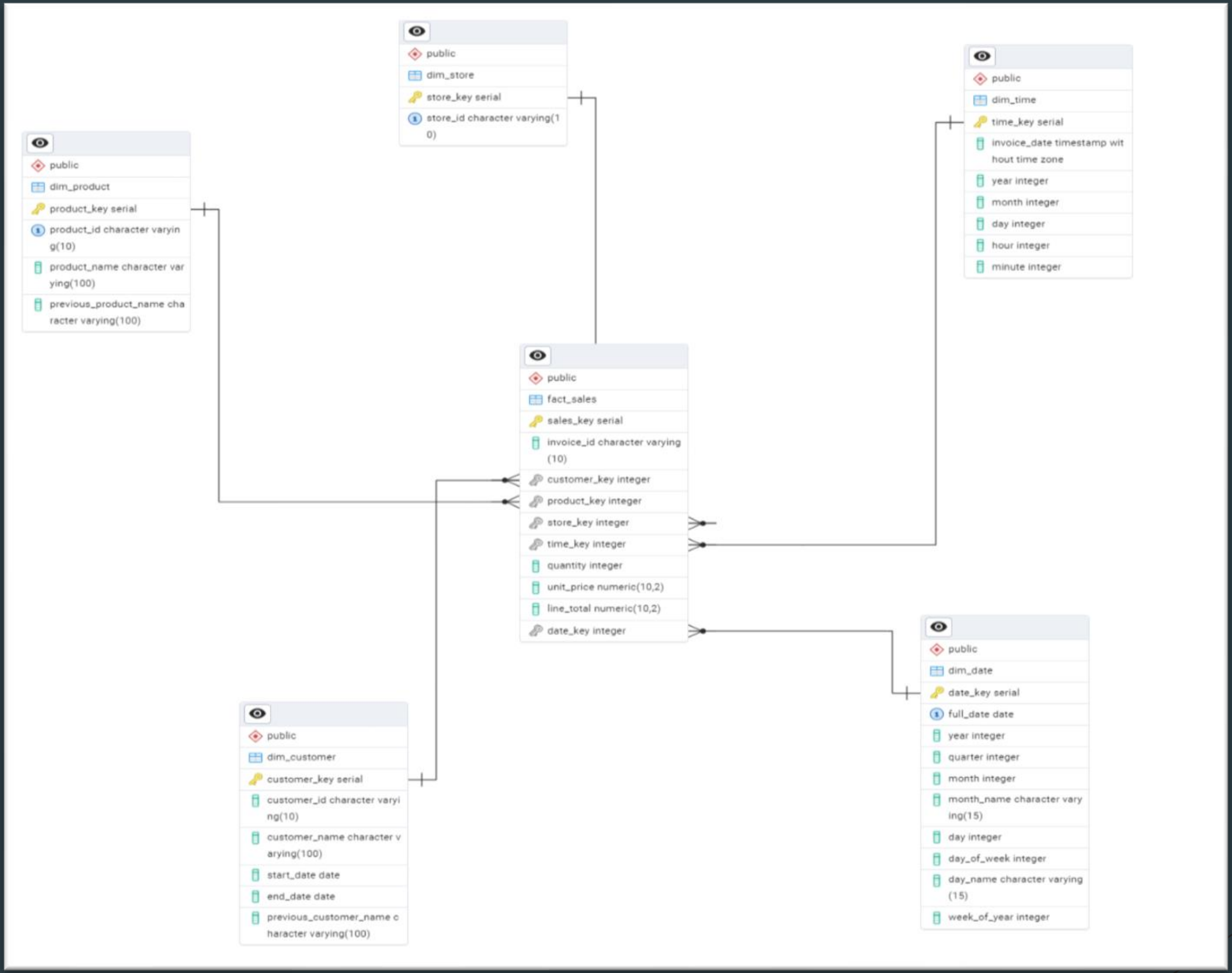
Sample Data set

Invoice_ID	Invoice_Date	Customer_ID	Customer_Name	Product_ID	Product_Name	Quantity	Unit_Price	Line_Total	Store_ID
INV001	2025-03-15 09:15:00	C001	John Smith	P1001	Wireless Mouse	2	25.00	50.00	S01
INV001	2025-03-15 09:15:00	C001	John Smith	P1002	Mechanical Keyboard	1	85.00	85.00	S01
INV002	2025-03-15 10:05:00	C002	Jane Doe	P1003	HD Monitor	1	200.00	200.00	S02
INV003	2025-03-15 11:30:00	C003	Bob Johnson	P1001	Wireless Mouse	1	25.00	25.00	S01
INV003	2025-03-15 11:30:00	C003	Bob Johnson	P1004	USB-C Hub	2	30.00	60.00	S01

Inferring Dimensions

- Customer Dimension (dim_customer): Captures customer details.
- Product Dimension (dim_product): Stores product-related attributes.
- Store Dimension (dim_store): Contains store-related identifiers.
- Time Dimension (dim_time): Extracts hierarchical time attributes for efficient time based analysis.
- Date Dimension (dim_date): Stores additional date attributes for better time-based aggregations.
- Fact Table (fact_sales): Captures transactional details linking dimensions via foreign keys.

Dimensional Model Design



public
dim_store
store_key serial
store_id character varying(10)

public
dim_product
product_key serial
product_id character varying(10)
product_name character varying(100)
previous_product_name character varying(100)

public
dim_time
time_key serial
invoice_date timestamp without time zone
year integer
month integer
day integer
hour integer
minute integer

public
fact_sales
sales_key serial
invoice_id character varying(10)
customer_key integer
product_key integer
store_key integer
time_key integer
quantity integer
unit_price numeric(10,2)
line_total numeric(10,2)
date_key integer

public
dim_customer
customer_key serial
customer_id character varying(10)
customer_name character varying(100)
start_date date
end_date date
previous_customer_name character varying(100)

public
dim_date
date_key serial
full_date date
year integer
quarter integer
month integer
month_name character varying(15)
day integer
day_of_week integer
day_name character varying(15)
week_of_year integer

Handling Slowly Changing Dimensions (SCD)

- In this schema, I implemented SCD Type 2 in the dim_customer and dim_product tables.
- This method ensures we keep track of historical changes while maintaining data integrity.
- SCD Type 2 Implementation: When a customer’s name or product name changes, a new row is added with a new customer_key or product_key.
- The old record is marked with an end_date, while the new one has a start_date.
- This allows us to track changes over time and analyze past trends accurately. Example: If a customer changes their name due to marriage, we retain both old and new names to maintain accurate historical reporting.

ETL/ELT Process Flow

- To populate this schema, I followed a structured ETL process:
- Extract:We pull raw sales data, including store, customer, product, and time details.
- Transform:We standardize data formats to maintain consistency.We assign surrogate keys to ensure unique identifiers for records.Date and time are split into separate dimensions for better time-based analysis.Data is normalized into dimensions to remove redundancy and improve performance.
- Load:We insert the cleaned and structured data into fact and dimension tables.fact_sales stores the sales transactions with foreign keys linking to the dimension tables.

Performance Optimization

- Indexing: Primary and foreign keys indexed for fast joins.
- Partitioning: Considered for fact_sales based on time_key or date_key (not implemented due to simplicity).
- Query Optimization: Reduced sorting costs by using efficient join methods and indexing critical columns.

How This Schema Supports Business Objectives

- Track customer purchases over time.
- Analyze customer retention and churn by observing changes in dim_customer.
- Identify high-revenue products using dim_product.Compare sales performance across stores with dim_store.
- Evaluate time-based trends with dim_date and dim_time.
- Detect seasonal sales patterns.Use dim_time and dim_date for year-over-year and month-over-month comparisons.