Data Warehouse

PROJECT

Muhammad Haziq Ijaz   I21-2692

Section: DS-A

METRO Shopping Store

1. Project Overview

This project implements a **near-real-time Data Warehouse** for METRO Shopping Store in Pakistan, designed to analyze customer shopping behavior effectively. Leveraging the **MESHJOIN algorithm**, the system facilitates efficient streaming ETL processes, enabling rapid analysis of customer transactions.

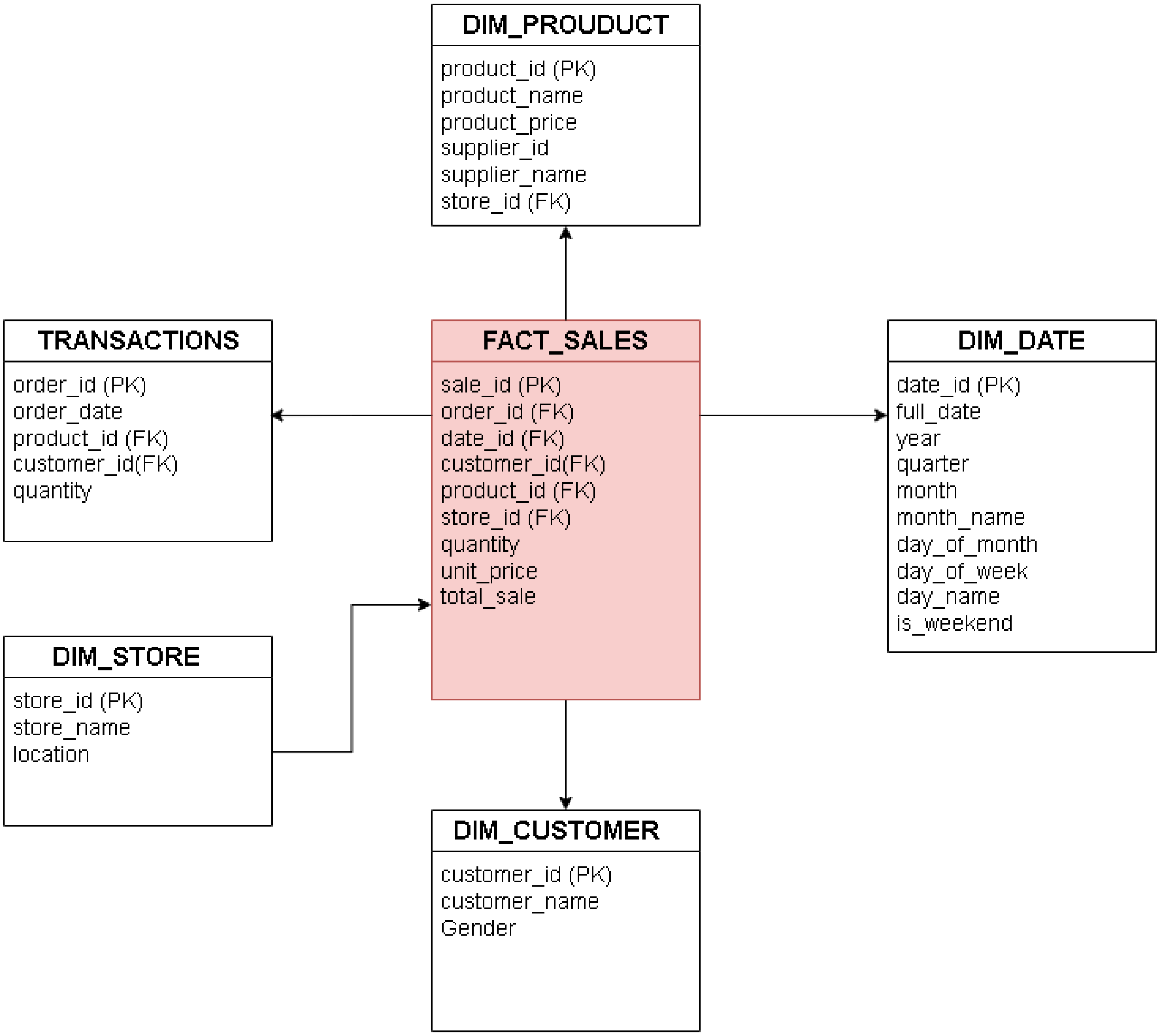
Key Components:

- **Star Schema Data Warehouse:** Optimized for analytical queries.
- **MESHJOIN Algorithm:** Efficient streaming data processing.
- **ETL Processes:** Seamless data integration and loading.
- **OLAP Queries:** Advanced business analytics for decision-making.

Objectives:

1. Build a near-real-time data warehouse for retail data.
2. Process streaming transaction data efficiently using cutting-edge algorithms.
3. Enable complex business analytics for actionable insights.
4. Optimize query performance for rapid data-driven decisions.

2. Data Warehouse Schema (Star Schema)



DIM_PROUDUCT

product_id (PK)
product_name
product_price
supplier_id
supplier_name
store_id (FK)

TRANSACTIONS

order_id (PK)
order_date
product_id (FK)
customer_id(FK)
quantity

FACT_SALES

sale_id (PK)
order_id (FK)
date_id (FK)
customer_id(FK)
product_id (FK)
store_id (FK)
quantity
unit_price
total_sale

DIM_DATE

date_id (PK)
full_date
year
quarter
month
month_name
day_of_month
day_of_week
day_name
is_weekend

DIM_STORE

store_id (PK)
store_name
location

DIM_CUSTOMER

customer_id (PK)
customer_name
Gender

3. MESHJOIN Algorithm

Overview: The **MESHJOIN algorithm** optimizes near-real-time ETL by efficiently joining streaming data with master data.

Key Components:

1. **Stream Buffer:**
    - Manages incoming transaction data in fixed-sized chunks for memory efficiency.
2. **Disk Buffer:**
    - Stores master data partitions; utilizes cyclic loading for efficient access.
3. **Hash Table:**
    - Enables quick lookups for join operations; resides in memory.

Implementation Process:

1. Load a stream chunk of transaction data.
2. Fetch a partition of master data from disk.
3. Perform join operations using hash tables.
4. Process and commit results to the data warehouse.
5. Clean up processed chunks to optimize resources.

4. **MESHJOIN Shortcomings**

1. **Memory Constraints:**
    - Performance limited by available RAM.
    - Buffer size requires careful configuration to handle large data streams.
2. **Join Latency:**
    - Requires complete master data cycle before processing new transactions.
    - Trade-off between throughput and latency.
3. **Data Skew Sensitivity:**
    - Uneven data distribution can create performance bottlenecks.
    - Leads to inefficient memory utilization and processing delays.

5. Learning Outcomes

Through this project, I have developed and honed several technical and business skills:

Technical Skills:

- **Data Warehouse Design:**
  I gained expertise in star schema modeling and designing dimension tables, along with optimizing queries for improved performance.

- **ETL Processing:**
  I learned about stream processing concepts, integrating real-time data, and implementing effective recovery mechanisms to handle errors.
- **Programming Proficiency:**
  I worked extensively with SQL and Java, mastering the creation of complex OLAP queries and implementing stream processing workflows.

Business Insights:

- **Retail Analytics:**
  I developed the ability to analyze customer behavior patterns, evaluate product performance, and assess store metrics for actionable insights.
- **Performance Optimization:**
  I improved my skills in query tuning, memory management, and efficient data loading techniques to enhance system performance.