

Data Warehousing and business analytics

DS3003



November 26, 2024

HAMZA MAHMOOD BURNEY (22i-2058)

SUBMITTED TO: Dr. Asif Naeem

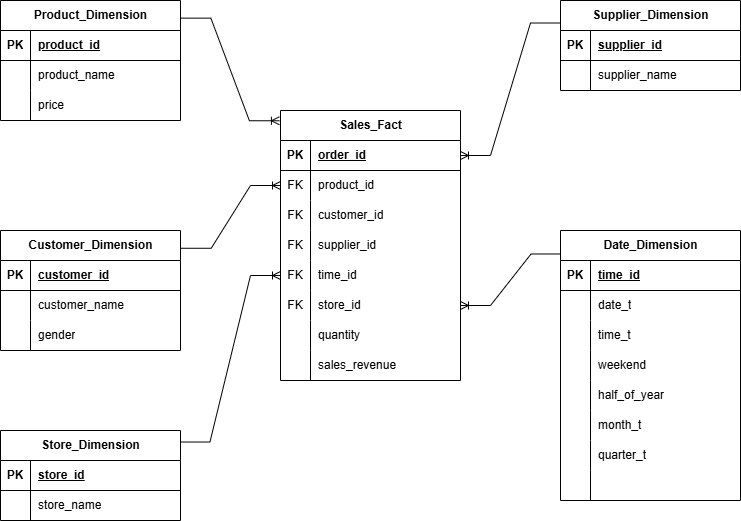


**Project**

**Project Overview:**

The project involves the creation of a Data Warehouse (DW) to facilitate advanced reporting and data analysis for METRO Cash & Carry Pakistan. This DW incorporates data from various dimensions such as products, customers, suppliers, stores, and time. The main aim is to provide comprehensive insights into sales patterns and customer behavior, enhancing strategic decision-making processes.

**Schema:**



**Project Architecture**

1. Dimensional Modeling:
   * Fact Table: The central table is the Sales\_Fact table, which records transactional data like quantity and sales revenue. It references all the dimension tables.
   * Dimension Tables: Includes Product\_Dimension, Customer\_Dimension, Supplier\_Dimension, Store\_Dimension, and Date\_Dimension. These tables store attributes related to their respective entities.
2. Database Schema:
   * Primary and foreign keys are established to ensure referential integrity and efficient data retrieval.
   * Dimension tables are linked to the fact table through foreign keys.
3. Data Integration and ETL Process:
   * Data Extraction: Data is sourced from relational databases and CSV files.
   * Data Transformation: Includes cleaning, filtering, and partitioning of data to fit the DW schema.
   * Data Loading: Using Java for backend processing, the data is loaded into the DW. The provided Java code handles the transformation and loading phases, including mesh joins of product and customer data based on transactions.
4. Technology Stack:
   * Database Management: MySQL for managing relational databases.
   * Programming: Java is used for the ETL process, handling file reading, data transformations, and database interactions.
5. Implementation Details:

Mesh Join Algorithm: It is an optimized approach for performing joins on large volumes of data, typically involving multiple dimensions. The primary goal is to reduce the computational overhead and improve the throughput of data processing tasks, especially when dealing with Big Data scenarios in a Data Warehouse environment.

* + Data Partitioning:
    1. The algorithm starts by partitioning reference data (products and customers) into multiple segments. This is based on a pre-defined number of partitions (NUMBER\_OF\_PARTITIONS).
    2. Each record from the product and customer datasets is assigned to a partition based on a hashing or round-robin mechanism, ensuring an even distribution.
  + In-memory Storage:
    1. Partitioned data is stored in memory to facilitate rapid access during the join process. This is crucial for performance, as accessing disk-based data would significantly slow down the operations.
    2. productPartitions and customerPartitions maps hold the data, where each key represents a partition number and the value is a list of product or customer records respectively.
  + Concurrent Processing:
    1. Transactions are read from a CSV file and loaded into chunks. Each chunk is pushed into a blocking queue (transactionsQueue), which acts as a buffer and supports concurrent processing by multiple threads.
    2. A dedicated thread (transactionReaderThread) reads transaction data and populates the queue.
  + Join Operation:
    1. Multiple threads consume the transaction chunks from the queue and perform the join operation.
    2. Each thread calculates which partition of product and customer data it needs to access based on the transaction’s product and customer IDs. This minimizes the dataset each thread has to scan, thereby speeding up the join process.
    3. Once the relevant records are identified, a new JoinedRecord is created by merging fields from the transaction, product, and customer records. This record contains comprehensive details necessary for analysis, such as product name, customer name, sales revenue, etc.
  + Batch Insertion into Data Warehouse:

After processing, the joined records are collected and batch inserted into the DW. This approach minimizes the number of write operations on the database, further enhancing performance.

1. Security and Compliance:
   * Ensures data integrity and security through controlled access and secure database connections.
   * Compliance with data protection regulations by implementing appropriate data handling and storage protocols.

**Shortcomings of Mesh Join:**

**1. Memory Consumption**

* **High Memory Usage**: Mesh Join requires significant memory allocation, as it relies on loading data partitions into RAM for fast access. This can be problematic in environments where memory resources are constrained or when dealing with extremely large datasets that exceed the available memory.

**2. Load Balancing Challenges**

* **Uneven Data Distribution**: The efficiency of the Mesh Join depends heavily on the even distribution of data across partitions. If data is skewed, some partitions may end up significantly larger than others, leading to load imbalance among processing threads.

**3. Complexity in Implementation and Maintenance**

* **Complex Setup**: Implementing a Mesh Join algorithm requires a sophisticated setup that involves partitioning data, managing concurrency, and handling data in memory. This complexity increases the development time and the need for specialized knowledge.
* **Maintenance Overhead**: The dynamic nature of data—where the volume and distribution can change over time—necessitates continuous monitoring and potential reconfiguration of the partitioning logic to maintain optimal performance. The maintenance of such a system can be resource-intensive, requiring regular adjustments and tuning.

**Learning Objectives:**

* Learn the basic principles of Data Warehousing, including its architecture, the purpose of dimensional modeling, and the role of fact and dimension tables. Be able to explain the significance of a data warehouse in business intelligence and describe how it differs from traditional databases.
* Understand and apply the concepts of dimensional modeling such as star schema, and learn how to design dimension and fact tables.
* Acquire basic Java programming skills focused on data manipulation tasks common in ETL processes, such as reading from and writing to files, executing SQL queries, and handling data structures.
* Learn to implement ETL processes using Java, focusing on extracting data from source systems, transforming this data to fit the dimensional model, and loading it into the data warehouse.

**Conclusion**

The project effectively sets up a scalable DW architecture tailored to support complex querying and reporting needs. By leveraging a structured ETL process and robust dimensional modeling, the company is well-positioned to gain deep insights into its operational data, driving better business outcomes.