Data Warehouse

Project Report



Submitted By:

Syed Husnain Haider Bukhari

20i-0626

Bs DS-N

Contents

[Introduction 3](#_Toc120402606)

[Project Overview 3](#_Toc120402607)

[Schema for Data Warehouse 3](#_Toc120402608)

[Diagram: 3](#_Toc120402609)

[MESHJOIN Algorithm 5](#_Toc120402610)

[Shortcomings of Mesh Join 5](#_Toc120402611)

[Learning outcomes 5](#_Toc120402612)

# Introduction

This document reports the Data Warehouse project implemented and completed by Syed Husnain Haider Bukhari for the course Data Warehouse and Business Intelligence semester project 2022.

# Project Overview

This project was divided into three ETL steps:

1. We connect and extract the database data with Eclipse using Java.
2. We perform the necessary transformation steps on the loaded data from the database in Java.
3. Loading the final data after transformation into the created data warehouse.

All the above steps were performed using extended Mesh Join algorithm.

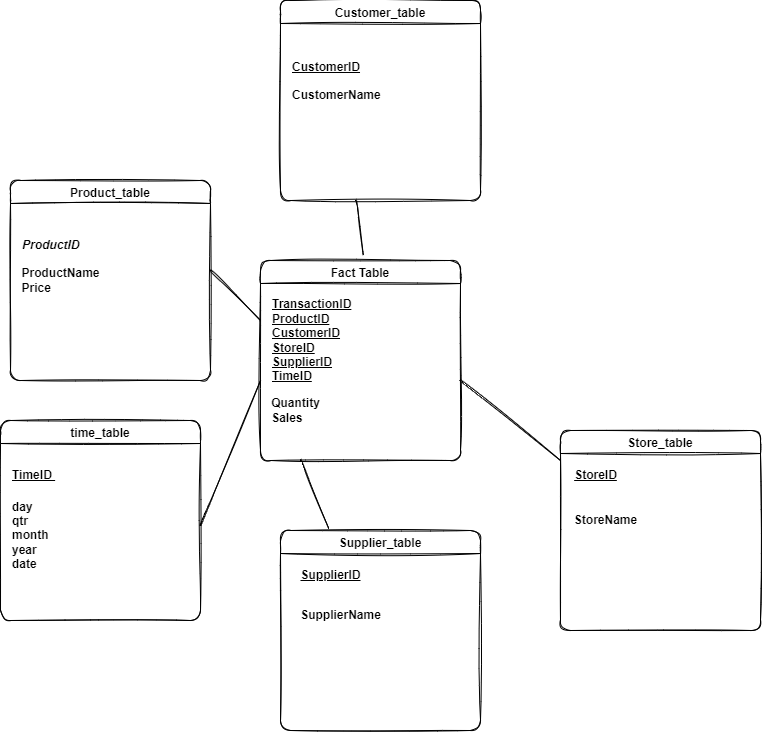
# Schema for Data Warehouse

After the analysis of the provided data and the analysis questions, I created the Snowflake schema for the data warehouse. Using other techniques as Star and Star schema, was not that relevant and beneficial for this situation. I made the individual dimension tables for "Customer," "Product," "Store,” Supplier," Time," and the Fact table having sales and quantity as a measure.

An issue in the data warehouse data was that master data had a product called "tomatoes," which had two different prices, which was an anomaly.

## Diagram:

**(The Star schema is attached below)**



# MESHJOIN Algorithm

The algorithm starts by reading and storing the data from the streaming data of the transaction with a block size of 50, adding it into an ArrayBlockingQueue by storing it into a hashmap using the ProductID and CustomerID, separately after that hashing it onto a hash table. Meanwhile, the data is read from master data divided into ten partitions for each table of Products and Customers with a size of 10 and 5, respectively. For joining the two tables, it repeatedly iterates over the current partition of master data (Product and Customer tables) and gets its "ProductID" and "CustomerID," respectively. It then checks for the "ProductID" on the hashtable once it is matched then it checks for the "CustomerID" after it, the process of data enrichment is done like data from the stream, product, and customer tables are inserted into the respective tables in the data warehouse like "Product ID" and "Product Name" into the "Product" dimension table as well as on the fact table relevant fields. In the fact table, the "Sales" attribute is calculated using "Quantity" from transactional data and "Price" from the master data Product table. The ArrayBlockingQueue maintains the order of the transactions and ensures that once the partition has been checking with all the blocks of master data until matched, in either case, once the queue gets full, it de-queues the head, which was the oldest partition (having 50 transactions).

# Shortcomings of Mesh Join

1. Stream records are waiting unnecessarily before being processed.
2. Streams are not processing in parallel thus it can be improved, and the process time can be reduced.
3. MESHJOIN cannot deal with a non-uniform stream and bursty stream effectively
4. Requires external dependency (Jar files).
5. It causes duplications when we use two product table ids and hash them together.

# Learning outcomes

* This was the first and hopefully last project implemented in JAVA.
* Learning JAVA and exploring the online available resources was a whole experience.
* Practically Implemented a data warehouse, like from the schema to ETL.
* Practically implemented the OLAP queries that not only helped in learning the implementation of techniques of drill down and roll up.
* I hope I can now Implement a DWH and resolve all the queries related to it.