COURSE CODE: DS3003

DATAWAREHOUSE AND BUSINESS INTELLIGENCE

NAME: HAMZA KHAN  
ROLL NO: 20I-0583

BS(DS)-N

Table of Contents

[Project Overview 3](#_Toc120452758)

[Data Warehouse Schema 3](#_Toc120452759)

[Mesh-Join Algorithm 4](#_Toc120452760)

[3 Shortcomings of Mesh-Join 4](#_Toc120452761)

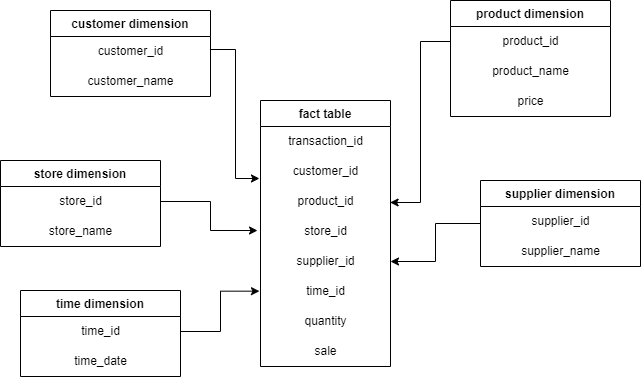
[What I learned 4](#_Toc120452762)

## Project Overview

The primary purpose of this project was transfer data (in near real time) from Database to warehouse. To do that, we were required to implement mesh join between transactional data and master data to produce an output which could then be sent to data warehouse. The data warehouse was also designed and implemented in SQL with accordance to the data present. Finally, SQL queries were also performed to do analysis.

Anomaly in Q9: The anomaly I found out was that Tomatoes occur with 2 different product ids and 2 different prices

## Data Warehouse Schema



## Mesh-Join Algorithm

Def queue ‘q’ which can hold a list of pointers

Def Hash map ‘h’

Loop until all data loaded onto Data Warehouse:

Read a chunk of data from database and enqueue it in q

Hash each tuple in queue onto h

Read next partition of master data

If master data maps to h (via keys):

Append master data values to hash map values

If head of q went through all partitions of master data:

Def temp <- Pop head of q

Send those hash map key and values to data warehouse for which pointers are present in temp

Delete those key, value pair of h for which pointers are present in temp

## 3 Shortcomings of Mesh-Join

1. Suppose we have created 5 partitions of master data and the join is completed on the chuck of data in 1st partition. The algorithm still waits till the chuck of data has gone through all partitions even though it wasn’t necessary. Furthermore, there can also be a case where a partition of master data will not join with chuck of transactional data. Hence, we can conclude that mesh-Join will not always produce the optimal performance.
2. Time taken to perform mesh join could greatly increase if size of Master data is huge. This is due to the fact that we need to find out cartesian product to get the keys and see which of the keys are present in hash table. We can try to reduce the amount of data in buffer by creating more partitions of data. However, creating too many partitions can lead to too many iterations which again, can reduce performance. Furthermore, extra time will also be spent to read the partitions again and again.ie, increase in I/O operations.
3. Frequency of each products sold differ. Some products are more frequently sold than others. Mesh Join algorithm doesn’t take that into account.

## What I learned

1. How to create Java- SQL connection.
2. How to write SQL queries in Java and then retrieve data from Workbench
3. How to write SQL queries in Java and then send data to Workbench
4. How to practically implement Mesh Join which includes:
   1. Partitioning of DB data as well as Master data and iteratively choosing next partition
   2. Hashing each tuple in HashMap and storing pointers in Queue
   3. Joining Master data with DB data and sending tuples to Data Warehouse, then deleting those tuples from HashMap and popping Queue.
5. Finding out new Data Warehouse specific SQL queries such as rollup and what they do.
6. How to perform analysis on Data Warehouse.