****

**National University of Computer and Emerging Sciences**

**FAST School of Computing**

**Islamabad, Pakistan**

**CS4033 Project Report**

**Semester 7**

***Data Warehousing***

Building and Analysing a Near-Real-Time Data Warehouse

Prototype for METRO Shopping Store in Pakistan

**Submitted by:** 19i-0454 Aamna Kamran

**Submitted to:** Dr. Muhammad Asif Naeem

**Project Overview**

The goal of this project was to implement the Mesh Join Algorithm to make a near real time data warehouse prototype in Java.

The customers’ transactions were loaded into 2 separate hash tables: one for *Customers* (**hashTableC**) and one for *Products* (**hashTableP**); the keys used were *customer\_id* and *product\_id* respectively. A chunk of the transactions was also loaded into a queue from the **stream buffer**. The Master Data was loaded into the **disk buffer** and from there each partition is read one by one. From *Customers* table the *customer\_id* is searched for in **hashTableC** and from the *Products* table the *product\_id*  is searched in **hashTableP**. If a match is found the record is searched for in the queue and joined with the master data and then is saved in an ArrayList for **transformed data**. After all the master data partitions have been read for each queue element; the queue elements are removed in the same order as they arrived in the queue and a new chunk of customers’ transactions are loaded into the queue and hastables.

After we get the transformed data, all the records are distributed into the star schema; the OLAP queries can now be applied to the data.

The following dimensions were identified for the star schema of the data warehouse:

1. Customer
2. Product
3. Store
4. Supplier
5. Time

**DW Analysis**

**Query 2:** We can use Data Mining techniques to make predictions for future trends using the data of current trends.

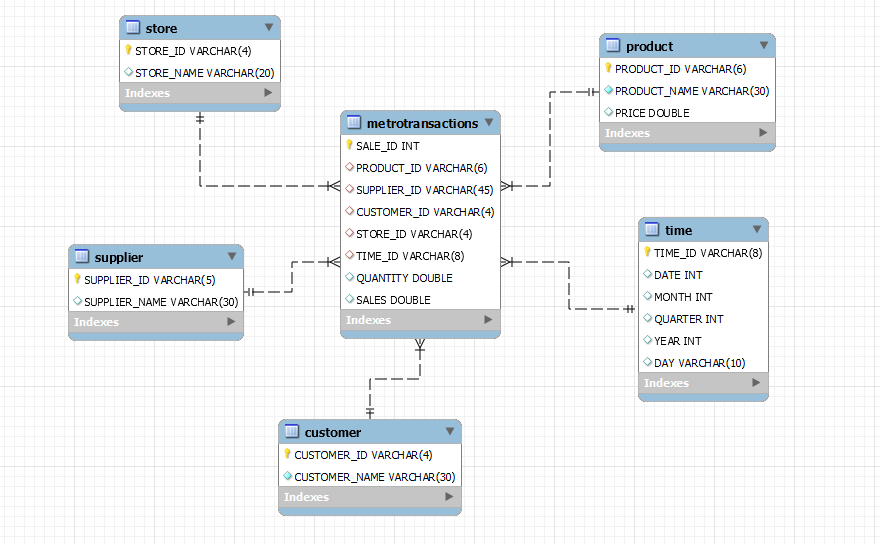
**Query 7:** These query results will show the maximum, minimum and average sales with respect to store, supplier and product.

**Query 9:** Any value which is outside (Average-3\*Standard Deviation) and (Average+3\*Standard Deviation) is strongly anomalous.

**Query 10:** How Materialized Views help in OLAP query optimization?

Materialized views reduce the execution times for complex queries that require joins and aggregate function; the effect of this optimization is realized when the OLAP query’s computation cost is high but the resulting data set is small.

**Data Warehouse Schema: Star Schema**

****

**Mesh Join Algorithm**

For 0 to Stream Buffer size / Number of Partitions

Read next Customer Transactions’ Number of Partitions size chunk to Queue and Hash Table

For each Customer Transaction element in queue

For 0 to Number of Partitions

Read Master Data partition

If Hash Table *contains* Master Data’s *customer\_id* OR *product\_id*

Write to Transformed Data element

EndIf

Read next Master Data Partition

If Master Data Partition is the last partition

Master Data Partition = first Master Data Partition

EndIf

Add Transformed Data element to Transformed Data List

Read next chunk of customers’ transactions

Remove all Queue elements

Remove all Hash Table elements

**Shortcomings of the Mesh Join Algorithm**

The dependency between the size of the partitions in the queue for the stream buffer and the number of iterations required to bring the master data into the disk buffer does not allow for an optimal distribution of memory among the join components.

The chunks of customers’ transactions leave main memory in the order that they enter main memory and their time of residence in main memory is overlapping. This leads to the staggered processing pattern of MESHJOIN.

**Learning**

1. Practical experience of writing and running OLAP queries which provide insightful data for analysis.
2. Designing a Data Warehouse: Implementation of a Star Schema in MySQL.
3. How to connect and access a Data Warehouse from a Java application (use of JDBC).
4. Materialized views cannot be created on MySQL