

Project Report

BS DS (M)

DWH & BI

Abdul Haque 21I-1769

Submitted to: Muhammad Asif Naeem

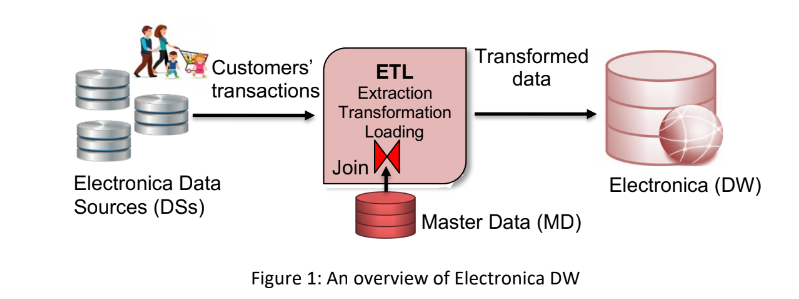
# Project Overview:

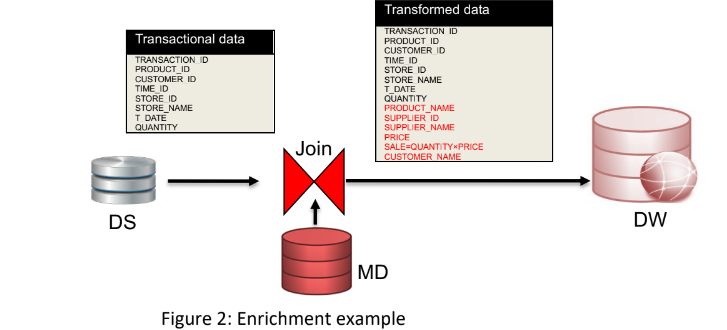
The project is assigned to us students and it is to design, implement, and analyze a Data Warehouse prototype for an Electronica Business Chain in Pakistan.

Electronica is one of the biggest Electronics Business chains in Pakistan and worldwide. The stores have thousands of customers and therefore it is important for the business to online analyze the shopping behavior of their customers. Based on that the business can optimize their selling techniques e.g. giving promotions on different products.

Now, to make this analysis of shopping behavior practical there is a need of building a near-real-time DW and customers’ transactions from Data Sources (DSs) are required to reflect into DW as soon as they appear in DSs. To build a near-real-time DW we need to implement a near-real-time ETL (Extraction, Transformation, and Loading) tools. Since the data generated by customers is not in the format required by DW therefore, it needs to process in the transformation layer of ETL. For example enriching of some information e.g.

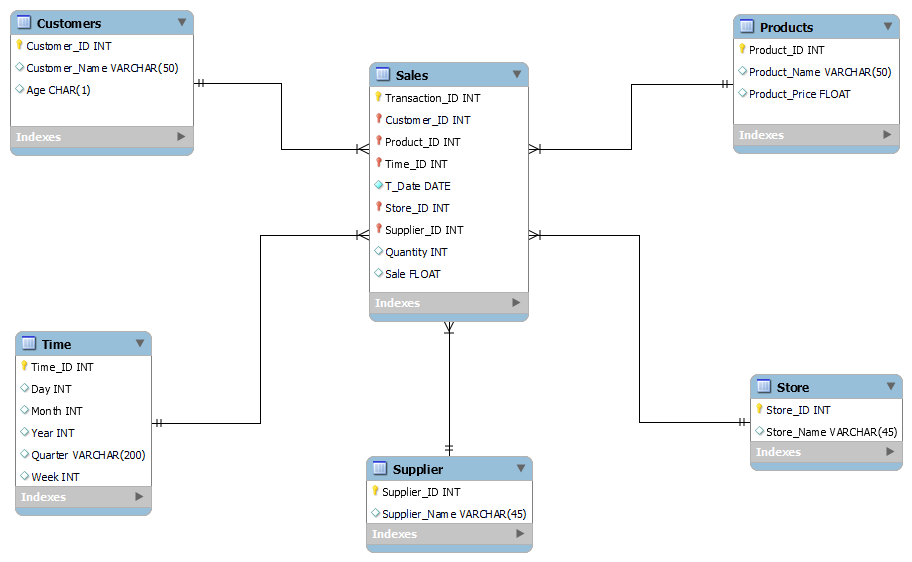
attributes in color red from disk-based Master Data (MD).





To implement this enrichment feature in the transformation phase of ETL we need a join operator. There are a number of algorithms available to implement this join operation however, the most popular one is HYBRID JOIN (Hybrid Join) which is explained in the next section and I will implement it in this project using Java Eclipse.

# Schema For DW:



The schema I designed for the data warehouse of ELECTRONICA.

# Hybrid Algorithm:

The pseudocode of Hybrid Join:

**Declare a Blocking Queue**

**BlockingQueue<ArrayList<String>> queue = new LinkedBlockingQueue<>();**

**Start StreamThread with Blocking Queue passed;**

**Start HybridJoinThread with Blocking Queue passed;**

**Inside StreamThread {**

**getConnection to transactions DB**

**Connection connect = DriverManager.getConnection(url, usr, pass);**

**create statement to run query**

**create ResultSet for storing query values and name it rs**

**while (rs.next())**

**{**

**fetch rows of data and store as tuple in private ArrayList<String> result = new ArrayList<>();**

**add result into our blocking queue**

**redeclare result to get new rows**

**BlockingQueue.put(result);**

**}**

**Close the Connection Connect.close();**

**}**

**Inside HybridJoinThread {**

**fetch 1000 rows to start with in queue**

**int i=0;**

**while(i < 1000)**

**{**

**Get the blockingQueue values by Recieve.take();**

**enqueue in Doubly Queue;**

**add the key and tuple in hash map**

**i++;**

**}**

**while(!Queue.isEmpty())**

**{**

**form a connection to masterData**

**get 10 rows from master data using Dequeue key**

**10 rows from master data will be Product\_ID >= DeQueue**

**Store those 10 rows in private ArrayList<ArrayList<String>> RESS = new ArrayList<>();**

**Iterate through values Iterator<ArrayList<String>> iter = RESS.iterator();**

**while(iter.hasNext())**

**{**

**get value of next iter value**

**get hash map tuple by getting first element of iter value**

**values = (List<ArrayList<String>>) map.remove(value.get(0));**

**start loop**

**for(int j=0; j<values.size(); j++)**

**{**

**getVals = values.get(j);**

**if (getVals.size() != 0)**

**{**

**get values from getvals list for Product\_ID, Store\_ID, etc**

**Connect with Electronica and insert values using queries**

**}**

**remove values from hashMap**

**remove values from iter**

**}**

**After 1000 rows start controller thread**

**if(Size of hash map is less than 1000)**

**{**

**increase speed of incoming stream**

**}**

**}**

**}**

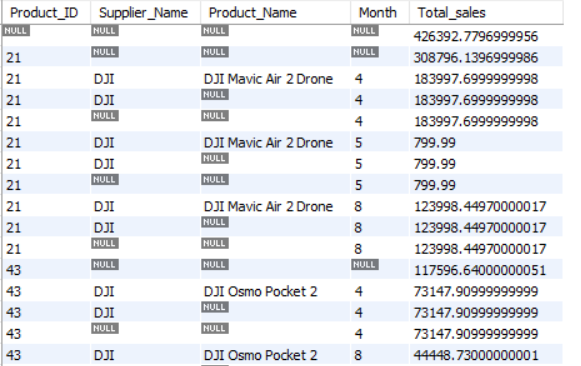
**}**

# OLAP Outputs:

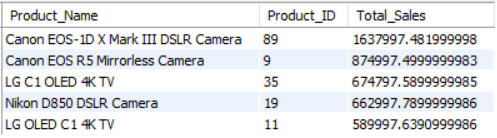
## Q1:



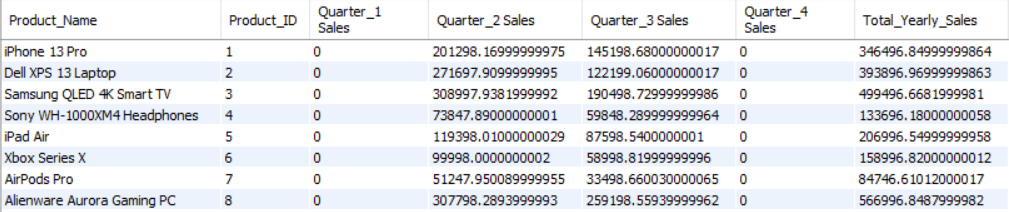
## Q2:



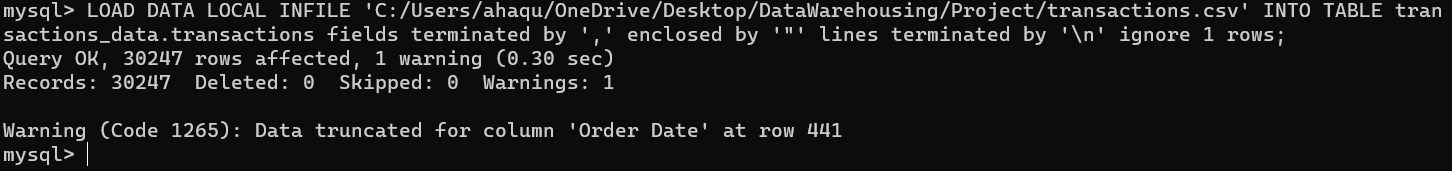
## Q3:



## Q4:



## Q5:

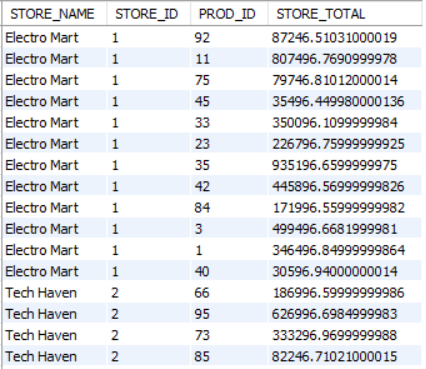


I have written query to fetch this anomaly but as my hybrid join wasn’t complete it wasn’t inserted inside my DW although the anomaly was 1819 date as in this DW we have to have dates till 2019 not before the query however is:

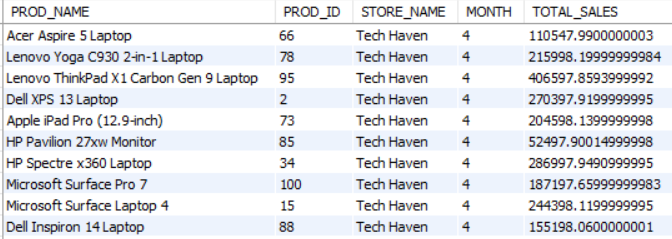
-- The anomaly wasn't inserted by my hybrid join but it is present when I was loading the csvs

select \* from `electronica-dw`.`Time` where `Time`.year != 2019;

## Q6:



## Q7:



## 

## 

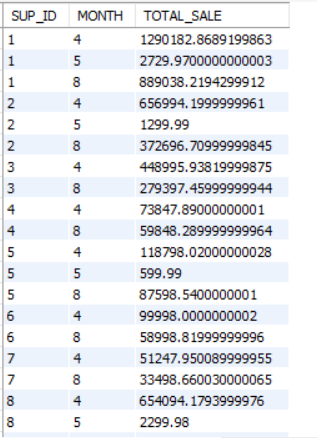
## 

## 

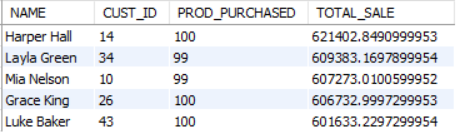
## 

## 

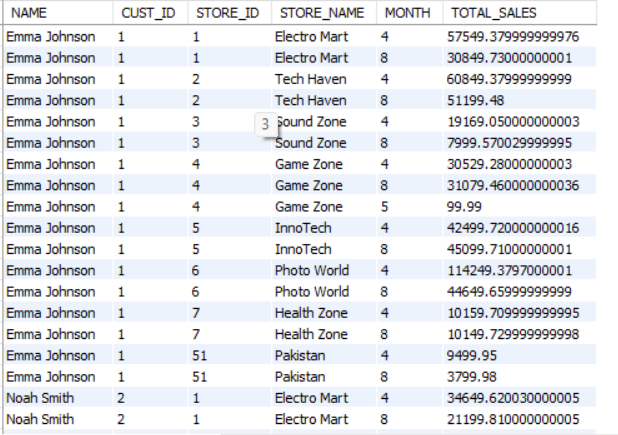
## Q8:



## Q9:



## Q10:



# Two ShortComings of Hybrid JOIN:

## 1: I/O Cost higher which can raise performance issues, the absolute performance cannot be driven out by this algorithm as there alot of DBS connected and taking input output in similar times which increase costs.

## 2: The Hybrid join is mostly used for limited resources and to be lightweight if we increase the dataset and have larger values then this algorithm will not achieve the goal effectively. When we talk about memory hash joining algorithms we need to utilize the memory and CPU cores effectively, however here it would be challenging as the characteristics of our data here is very different and we have to process it more.

# PROJECT LEARNING:

The things I learnt from this project is how to design and implement a DW and also how to do multithreading in java. In java it was a bit difficult for carrying inputs from one thread to another but I found a better way to do that using Blocking Queue which helped. Other than that joining the master with the ODS was challenging but was done, as for the controller I did make a code for that but I couldn’t run it in constant time and had some issues related to that. I also increased the amount of connections possible by MySql so that if I connect more DBs there is no problem in storing. The whole learning curve was steep and challenging.