Submitted By Ammad Umar i170092

Project Overview

METRO is one of the biggest superstores chains in Pakistan having thousands of customers We intend to optimise their selling techniques e.g. giving of promotions on different products.

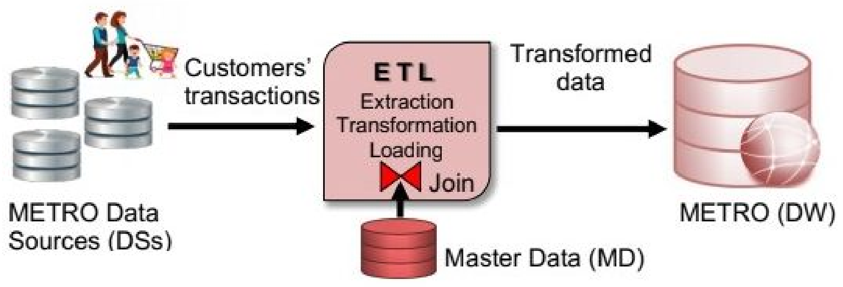


Figure 1: An overview of METRO DW

We are building a near-real-time DW therefore we need to implement a near-real-time ETL (Extraction, Transformation, and Loading) tool to process in the transformation layer of ETL.

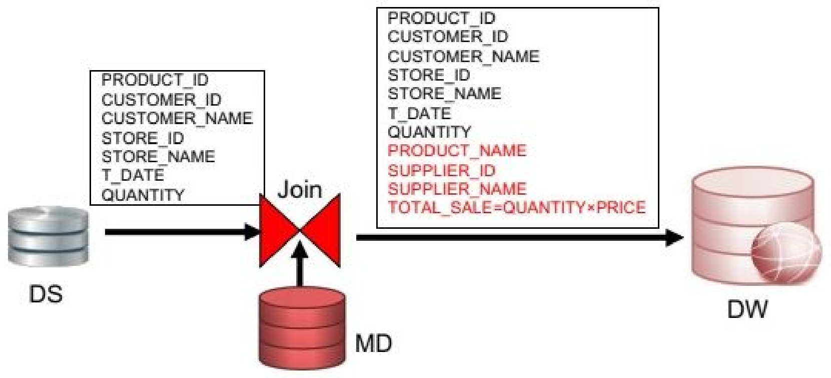
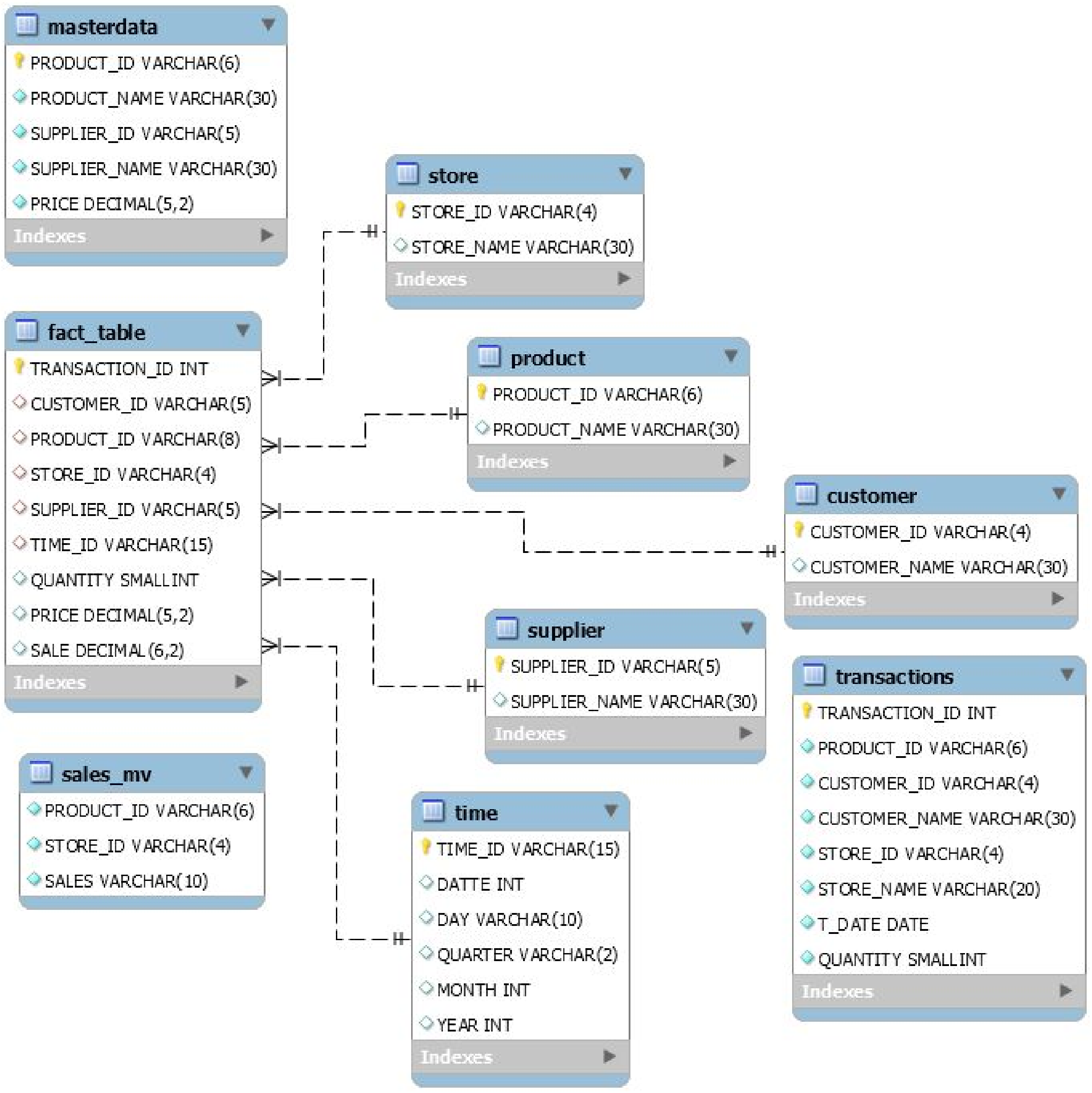


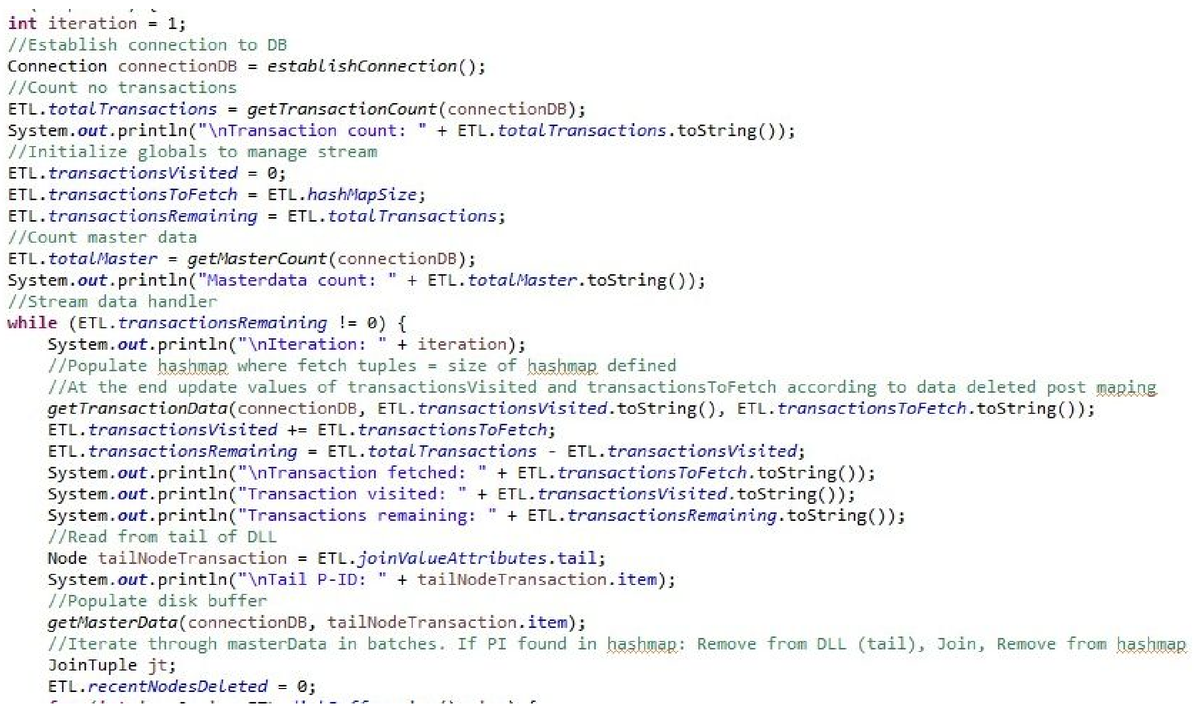
Figure 2: Enrichment example

We are implementing HYBRIDJOIN using Java Eclipse 8 and JDK 8..

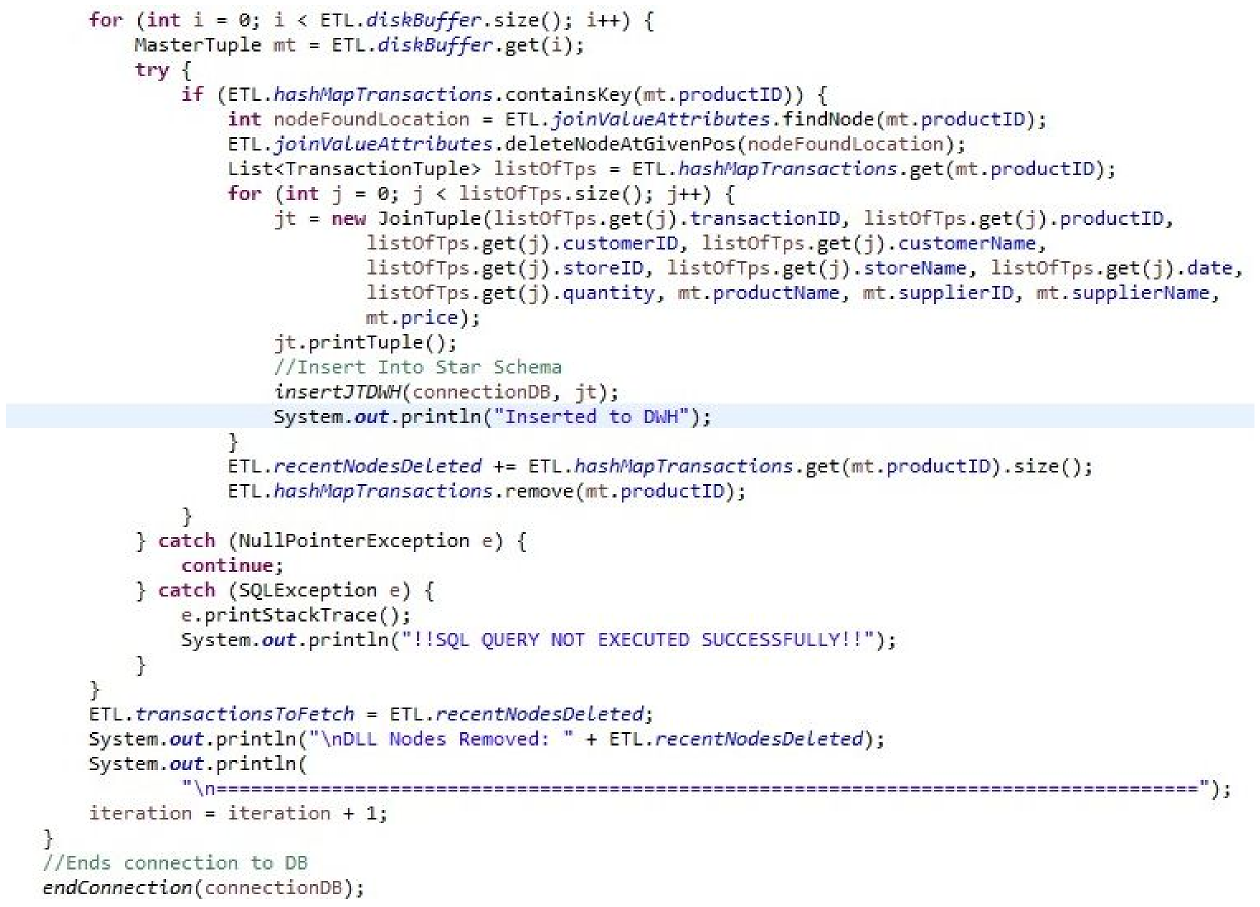
Metro Schema



HybridJoin Algorithm



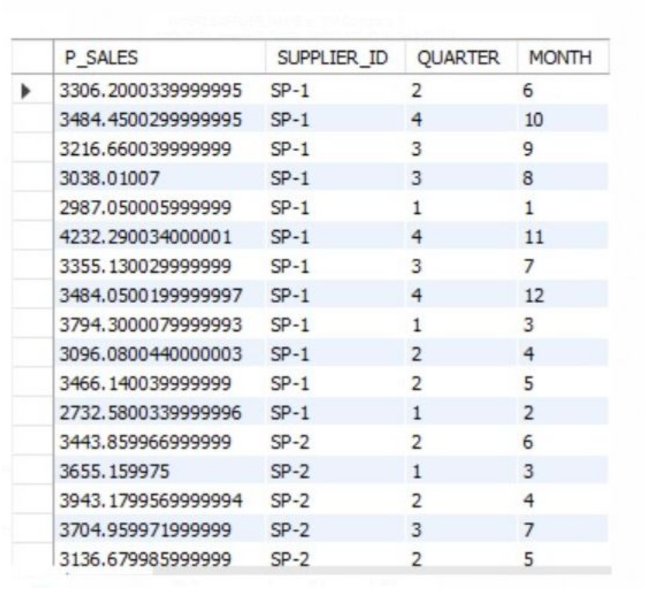
```java
int iteration = 1;
//Establish connection to DB
Connection connectionDB = establishConnection();
//Count no transactions
ETL.totalTransactions = getTransactionCount(connectionDB);
System.out.println("\nTransaction count: " + ETL.totalTransactions.toString());
//Initialize globals to manage stream
ETL.transactionsVisited = 0;
ETL.transactionsToFetch = ETL.hashMapSize;
ETL.transactionsRemaining = ETL.totalTransactions;
//Count master data
ETL.totalMaster = getMasterCount(connectionDB);
System.out.println("Masterdata count: " + ETL.totalMaster.toString());
//Stream data handler
while (ETL.transactionsRemaining != 0) {
    System.out.println("\nIteration: " + iteration);
    //Populate hashmap where fetch tuples = size of hashmap defined
    //At the end update values of transactionsVisited and transactionsToFetch according to data deleted post maping
    getTransactionData(connectionDB, ETL.transactionsVisited.toString(), ETL.transactionsToFetch.toString());
    ETL.transactionsVisited += ETL.transactionsToFetch;
    ETL.transactionsRemaining = ETL.totalTransactions - ETL.transactionsVisited;
    System.out.println("\nTransaction fetched: " + ETL.transactionsToFetch.toString());
    System.out.println("Transaction visited: " + ETL.transactionsVisited.toString());
    System.out.println("Transactions remaining: " + ETL.transactionsRemaining.toString());
    //Read from tail of DLL
    Node tailNodeTransaction = ETL.joinValueAttributes.tail;
    System.out.println("\nTail P-ID: " + tailNodeTransaction.item);
    //Populate disk buffer
    getMasterData(connectionDB, tailNodeTransaction.item);
    //Iterate through masterData in batches. If PI found in hashmap: Remove from DLL (tail), Join, Remove from hashmap
    JoinTuple jt;
    ETL.recentNodesDeleted = 0;
```



```java
    for (int i = 0; i < ETL.diskBuffer.size(); i++) {
        MasterTuple mt = ETL.diskBuffer.get(i);
        try {
            if (ETL.hashMapTransactions.containsKey(mt.productID)) {
                int nodeFoundLocation = ETL.joinValueAttributes.findNode(mt.productID);
                ETL.joinValueAttributes.deleteNodeAtGivenPos(nodeFoundLocation);
                List<TransactionTuple> listOfTps = ETL.hashMapTransactions.get(mt.productID);
                for (int j = 0; j < listOfTps.size(); j++) {
                    jt = new JoinTuple(listOfTps.get(j).transactionID, listOfTps.get(j).productID,
                            listOfTps.get(j).customerID, listOfTps.get(j).customerName,
                            listOfTps.get(j).storeID, listOfTps.get(j).storeName, listOfTps.get(j).date,
                            listOfTps.get(j).quantity, mt.productName, mt.supplierID, mt.supplierName,
                            mt.price);
                    jt.printTuple();
                    //Insert Into Star Schema
                    insertJTDWH(connectionDB, jt);
                    System.out.println("Inserted to DWH");
                }
                ETL.recentNodesDeleted += ETL.hashMapTransactions.get(mt.productID).size();
                ETL.hashMapTransactions.remove(mt.productID);
            }
        } catch (NullPointerException e) {
            continue;
        } catch (SQLException e) {
            e.printStackTrace();
            System.out.println("!!SQL QUERY NOT EXECUTED SUCCESSFULLY!!");
        }
    }
    ETL.transactionsToFetch = ETL.recentNodesDeleted;
    System.out.println("\nDLL Nodes Removed: " + ETL.recentNodesDeleted);
    System.out.println(
            "\n===================================================================================");
    iteration = iteration + 1;
}
//Ends connection to DB
endConnection(connectionDB);
```

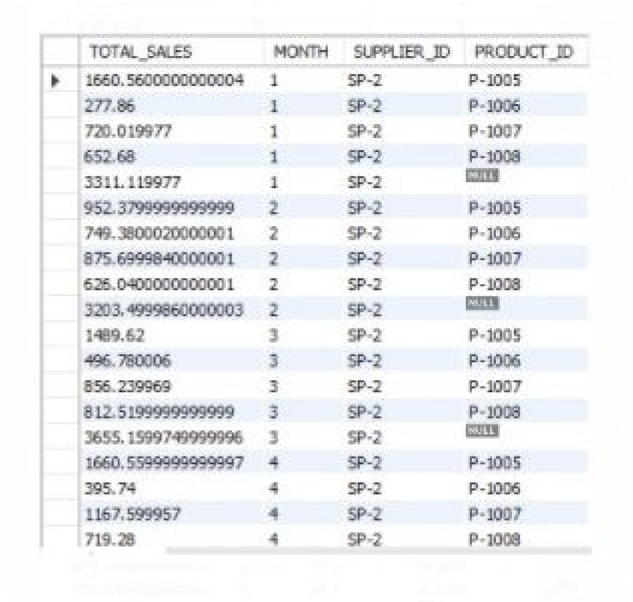
Output of Queries

Query 1



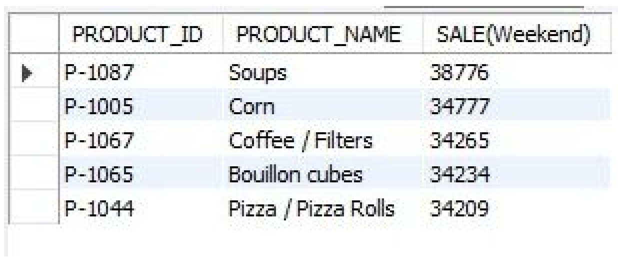
P_SALES	SUPPLIER_ID	QUARTER	MONTH
3306.2000339999995	SP-1	2	6
3484.4500299999995	SP-1	4	10
3216.660039999999	SP-1	3	9
3038.01007	SP-1	3	8
2987.050005999999	SP-1	1	1
4232.290034000001	SP-1	4	11
3355.130029999999	SP-1	3	7
3484.0500199999997	SP-1	4	12
3794.3000079999993	SP-1	1	3
3096.0800440000003	SP-1	2	4
3466.140039999999	SP-1	2	5
2732.5800339999996	SP-1	1	2
3443.859966999999	SP-2	2	6
3655.159975	SP-2	1	3
3943.1799569999994	SP-2	2	4
3704.959971999999	SP-2	3	7
3136.679985999999	SP-2	2	5

Query 2



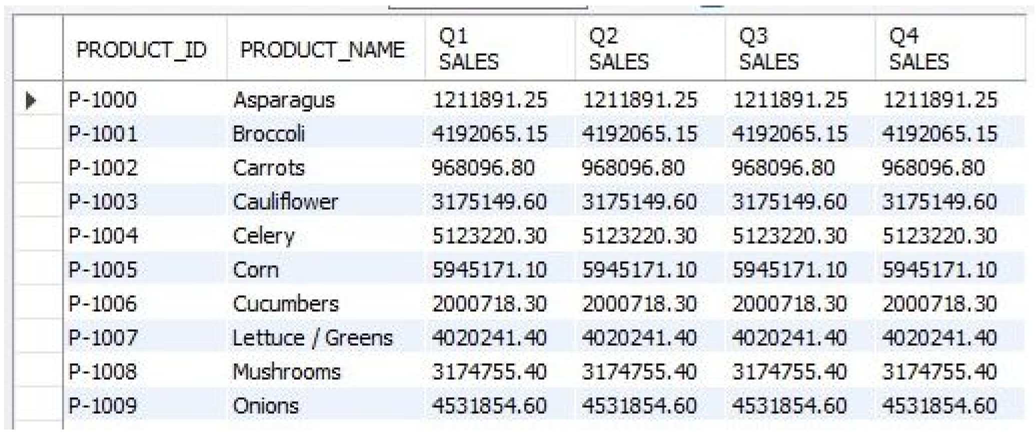
TOTAL_SALES	MONTH	SUPPLIER_ID	PRODUCT_ID
1660.5600000000004	1	SP-2	P-1005
277.86	1	SP-2	P-1006
720.019977	1	SP-2	P-1007
652.68	1	SP-2	P-1008
3311.119977	1	SP-2	NULL
952.3799999999999	2	SP-2	P-1005
749.3800020000001	2	SP-2	P-1006
875.6999840000001	2	SP-2	P-1007
626.0400000000001	2	SP-2	P-1008
3203.4999860000003	2	SP-2	NULL
1489.62	3	SP-2	P-1005
496.780006	3	SP-2	P-1006
856.239969	3	SP-2	P-1007
812.5199999999999	3	SP-2	P-1008
3655.1599749999996	3	SP-2	NULL
1660.5599999999997	4	SP-2	P-1005
395.74	4	SP-2	P-1006
1167.599957	4	SP-2	P-1007
719.28	4	SP-2	P-1008

Query 3



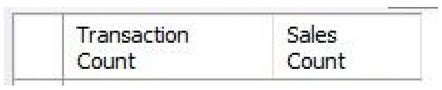
PRODUCT_ID	PRODUCT_NAME	SALE(Weekend)
P-1087	Soups	38776
P-1005	Corn	34777
P-1067	Coffee / Filters	34265
P-1065	Bouillon cubes	34234
P-1044	Pizza / Pizza Rolls	34209

Query 4



PRODUCT_ID	PRODUCT_NAME	Q1 SALES	Q2 SALES	Q3 SALES	Q4 SALES
P-1000	Asparagus	1211891.25	1211891.25	1211891.25	1211891.25
P-1001	Broccoli	4192065.15	4192065.15	4192065.15	4192065.15
P-1002	Carrots	968096.80	968096.80	968096.80	968096.80
P-1003	Cauliflower	3175149.60	3175149.60	3175149.60	3175149.60
P-1004	Celery	5123220.30	5123220.30	5123220.30	5123220.30
P-1005	Corn	5945171.10	5945171.10	5945171.10	5945171.10
P-1006	Cucumbers	2000718.30	2000718.30	2000718.30	2000718.30
P-1007	Lettuce / Greens	4020241.40	4020241.40	4020241.40	4020241.40
P-1008	Mushrooms	3174755.40	3174755.40	3174755.40	3174755.40
P-1009	Onions	4531854.60	4531854.60	4531854.60	4531854.60

Query 5



Transaction Count	Sales Count
10000	9990

Since Transaction count (sales data orignal) > Sale(transactions joined with masterdata), duplicates exist in original data which cause an anomaly.

Query 6

Shortcomings of HybridJoin

- Hybrid join sequentially loads from disk buffer, Searches hashtable and Doubly link list which takes too much time

Learning from Project

Since I had not taken Advanced Programming elective, I learned Java language as a challenge and implemented Hybrid Join using collections, data buffers and doubly linked list data structure. I learned about implementing a Data Warehouse from scratch