

Data Warehousing - PROJECT REPORT

**Name: Mohammad Nouman**

**Roll no: 17I-0235**

**Section: ‘A’**

**Building and Analyzing Data Warehouse Prototype for METRO Shopping Store**

# PROJECT OVERVIEW

### PROJECT SCOPE

* Building a data warehouse prototype for transactional data stream of Metro Cash n Carry
* Learn the implementation of ETL techniques for mapping data from database into data warehouse
* Perform OLAP queries for data analysis on the built data warehouse

### BREAKDOWN OF TASKS

#### Identifying the facts and dimensions

* Identify the facts and dimensions in the given data and list them down, it’ll help in drafting the structure of the Data Warehouse
* Define granularities for the chosen dimensions referring to the business questions asked
* Identify the measures for calculating the facts (like sales in this case)

The following Facts and Dimensions were identified in the project with granularities (if any)

##### Facts:

1. Sales
2. Quantity (sold in a single transaction)

##### Dimensions:

1. Customer Dimension
2. Store Dimension
3. Supplier Dimension
4. Product Dimension
5. Time Dimension
   1. Year
   2. Quarter
   3. Month
   4. Date of Month
   5. Date (full date, unique)
   6. Weekday Name

The Sales fact value is calculated as a product of Quantity sold in transaction and Unit price of the product

#### Defining Attributes for Facts and Dimension Tables

The fact and dimension tables are composed of following attributes

##### Dimensions:

1. Customer Dimension
   1. Customer id (Unique)
   2. Customer name
2. Store Dimension
   1. Store id (Unique)
   2. Store name
3. Supplier Dimension
   1. Supplier id (Unique)
   2. Supplier name
4. Product Dimension
   1. Product id (Unique)
   2. Product name
   3. Unit price
5. Time Dimension
   1. Year (Unique)
   2. Quarter
   3. Month
   4. Date of Month
   5. Date (full date, unique)
   6. Weekday Name

##### Fact:

1. Sales Fact (Table)
   1. Customer id
   2. Product id
   3. Supplier id
   4. Store id
   5. Date
   6. Quantity
   7. Sales

A materialized, view with following attributes, is also created as demanded in the problem statement

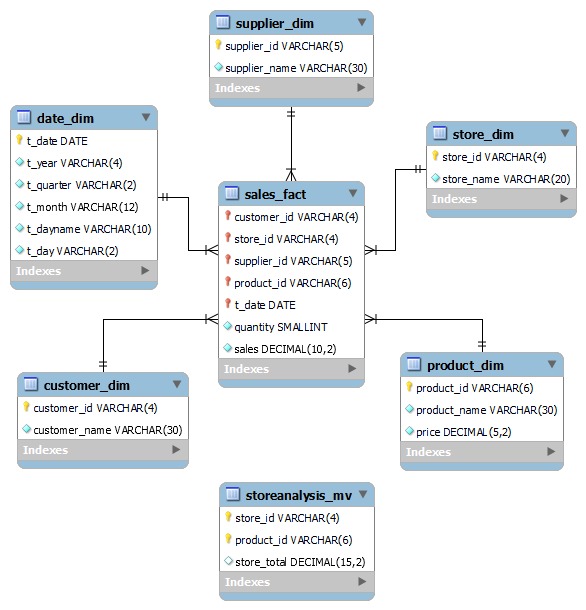
* Store Analysis Materialized View

1. Store id
2. Product id
3. Total sales

This view table contains the product wise total sales of each store

#### Designing Star Schema

The image below is the Star Schema formed for the Data Warehouse



#### Extraction, Transformation and Loading processes

The ETL process is carried out using Mesh Join algorithm in Java.

The following classes were made with their given attributes and methods

|  |  |
| --- | --- |
| Transactions  Contains the data of a transaction | |
| Attributes | Methods |
| transaction id product id customer id customer name store id  store name  date quantity | * Transactions () Default Constructor * Fill (…)   (takes the attribute values of the class object as parameter and assign them to respective attributes) |

|  |  |
| --- | --- |
| Mdata  Contains the data of a single tuple from master data | |
| Attributes | Methods |
| product id product name supplier id supplier name double price | * Mdata ()   Default Constructor   * Fill (…)   (takes the attribute values of the class object as parameter and assign them to respective attributes) |

|  |  |
| --- | --- |
| Fact  Contains the joint data of a transaction and its associated master data tuple | |
| Attributes | Methods |
| transaction id | * Fact () |
| customer id  customer name | Default Constructor   * Fill (…) |
| store id | Takes the attribute values of the class |
| store name | object as parameter and assign them to |
| product id  product name | respective attributes   * Calculate Sale () |
| supplier id | Calculate the sales value of the fact as a |
| supplier name | product of price and quantity |
| date |  |
| double price |  |
| quantity |  |

|  |  |
| --- | --- |
| MeshJoin  Contains main function implementing java meshjoin algorithm | |
| Attributes | Methods |
| none | * Public Static Void Main (…) Implements Mesh Join |

#### MeshJoin Algorithm

ResultSet RS1 //Result set for obtaining transactional data from database ResultSet RS2 //Result set for obtaining masterdata from database

int transOffset = 0 //offset to start reading from transaction table from

//database

int mdataOffset = 0 //offset to start reading from masterdata table from

//database

Queue <int> queue; //to keep transactions id of currently retrieved transaction hashtable <int, Fact> hashmap //hashtable with transaction IDs as key and

//Fact object as value Transactions t // Transactions object to store a transaction Mdata m // Masterdata object to store a masterdata entry

for (all 10000 transactions)

RS1 = select \* from transactions limit transoffset, 50 transOffset += 50

// 50 tupples read from the transactions data Queue.enqueue (RS1.transaction\_IDs)

// Retrieved all Transactions IDs enqueued

for (every transaction in RS1) t = RS1.next()

for (all 100 masterdata)

RS2 = select \* from transactions limit mdataoffset, 25 mdataOffset += 50

for(every masterdata in RS2) m = RS2.next();

if (m.productID = t.productID) //join condition Fact f = (m+t)

//fact object with joint data of m and t hashmap.put(t.transactionID, f)

break

break->if the join has already occured

Enumeration <Fact> e = hashmap.enumerate() //gets all the data values in hash

//table

while (e has more elements)

Fact f2 = e.next(element) //copies next element in hashtable into f2

//inserts into star schema tables before checking for duplicates executeQuery ("insert ignore supplier details into supplier dimension") executeQuery ("insert ignore customer details into supplier dimension") executeQuery ("insert ignore product details into supplier dimension") executeQuery ("insert ignore store details into supplier dimension") executeQuery ("insert ignore date details into supplier dimension") executeQuery ("insert ignore fact details into supplier dimension")

hashmap.clear() //clears hashtable for next chunk

//Now all of the data is loaded into star schema tables

//Popullate Materialized view

executeQuery ("insert into StoreAnalysisMV (select from store, product and fact tables)")

#### Analysis Queries’ Results

The snaps below are the piece of results of the respective queries

##### Query 1: Present total sales of all products supplied by each supplier with respect to quarter and Month.

##### Quarter Wise Sales:

##### A screenshot of a cell phone Description automatically generated

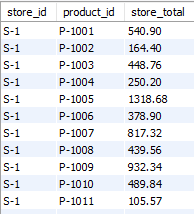
##### Month Wise Sales:

##### A screenshot of a cell phone Description automatically generated

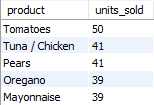
##### Joint sales of month and quarter wise:

##### A screenshot of a cell phone Description automatically generated

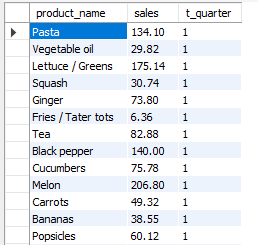
* **Query 2: Present total sales of each product sold by each store. The output should be organized store wise and then product wise under each store.**



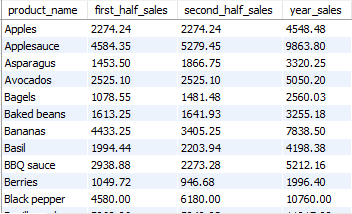
* **Query 3: Find the 5 most popular products sold over the weekends.**



* **Query 4: Present the quarterly sales of each product for year 2016 using drill down query concept.**



* **Query 5: Extract total sales of each product for the first and second half of year 2016 along with its total yearly sales.**



* **Query 6: Find an anomaly in the data warehouse dataset. Write a query to show the anomaly and explain the anomaly in your project report.**

The total number of tuples in sales fact table are less than the total transactions of data stream.

* **Query 7: Create a materialised view with name “STOREANALYSIS\_MV” that presents the product-wise sales analysis for each store.**

Materialized view was created in DDL.sql file and Populated in MeshJoin.java file.

The code for creating and populating table is given in Queries.sql file as well, to evaluate

Shortcoming of meshjoin and learnings from project

Shortcomings

* The provided algorithm apparently works for data from two sources only, it has to be improved to work on multiple data sources.
* High processing time as the joining of all the data causes excessive delay.
* Space inefficiency. It requires much space to process large data in using queues and hash tables and result sets.

Learnings

* How to identify a warehouse’s components from the database data
* How to design the structure of a data warehouse (the star schema)
* How to perform ETL operations on database and connect java code to MySQL database and run SQL queries in java
* How to use an ETL technique and how usually ETL techniques perform
* How to use a data warehouse for data analysis through OLAP queries
* How to manage a lot of work in little time