CS4033 DATA WAREHOUSING PROJECT

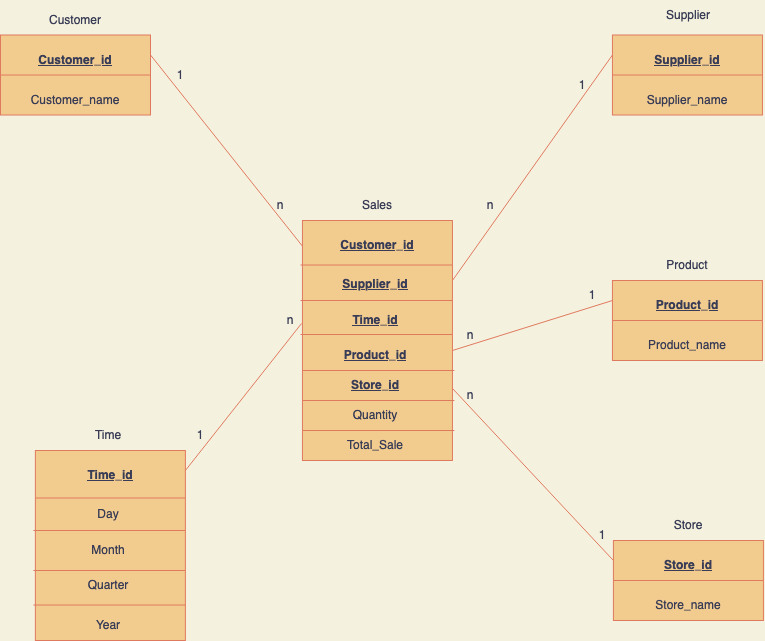
Building and Analyzing Data Warehouse Prototype for METRO Shopping Store

TALAL ZAFAR 18I-0497 (A)

Project Overview

The aim of the project was to design, implement, and analyze a Data Warehouse (DW) prototype for METRO shopping store in Pakistan so that we make analysis of shopping behavior, optimize selling techniques etc. We had to build the warehouse with the data source which we were given. For this we implemented a real time ETL (Extraction, Transformation, and Loading) because the warehouse schema is different from database. We implemented the Mesh Join algorithm for integrating the transactional data with master data before loading into warehouse. After building the warehouse I analyzed the DW by applying OLAP queries.

Schema for Data Warehouse



Mesh join Algorithm:

Start

While (reading\_data or Queue\_not\_empty)

stream\_buffer = load partition from Transactional table

Queue = insert(50) , hash\_table = insert (key,tuple)

disk\_buffer = load next partition from Master table

for i in disk\_buffer

if (hash\_table contains)

DW\_tuple = join(i,hash\_table(key))

load\_into\_DW(DW\_tuple)

if (Queue.size == partition.size)

Queue.poll()

hash\_table.remove()

end

OLAP QUERIES:

**-- 1ST QUERY**

SELECT SUPPLIER\_ID, QUARTER, MONTH, ROUND(SUM(TOTAL\_SALE),2) AS TOTAL\_SALES

FROM SALES

INNER JOIN TIME

USING (TIME\_ID)

GROUP BY SUPPLIER\_ID, QUARTER, MONTH

ORDER BY SUPPLIER\_ID;



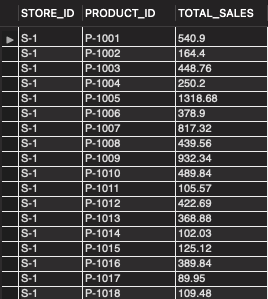
**-- 2ND QUERY**

SELECT STORE\_ID, PRODUCT\_ID, ROUND(SUM(TOTAL\_SALE),2) AS TOTAL\_SALES

FROM SALES

GROUP BY STORE\_ID, PRODUCT\_ID

ORDER BY STORE\_ID;



**-- 3RD QUERY**

SELECT PRODUCT\_ID, SUM(QUANTITY) AS QUANTITY

FROM SALES

INNER JOIN TIME

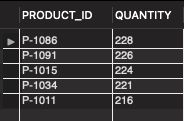
USING (TIME\_ID)

WHERE dayname(TIME\_ID) like 'Saturday' or dayname(TIME\_ID) like 'Sunday'

GROUP BY PRODUCT\_ID

ORDER BY QUANTITY DESC

LIMIT 5;



**-- 4TH QUERY**

SELECT PRODUCT\_ID,

ROUND (SUM (CASE WHEN QUARTER =1 THEN TOTAL\_SALE END),2) AS 1st\_QUARTER,

ROUND (SUM (CASE WHEN QUARTER =2 THEN TOTAL\_SALE END),2) AS 2nd\_QUARTER,

ROUND (SUM (CASE WHEN QUARTER =3 THEN TOTAL\_SALE END),2) AS 3rd\_QUARTER,

ROUND (SUM (CASE WHEN QUARTER =4 THEN TOTAL\_SALE END),2) AS 4th\_QUARTER,

ROUND (SUM (CASE WHEN QUARTER <=4 THEN TOTAL\_SALE END),2) AS TOTAL\_SALES

FROM SALES

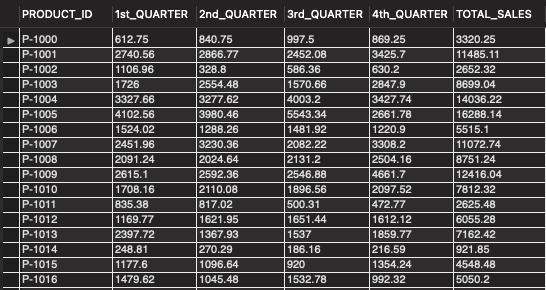
INNER JOIN TIME

USING (TIME\_ID)

WHERE YEAR =2016

GROUP BY PRODUCT\_ID

ORDER BY PRODUCT\_ID;



**-- 5TH QUERY**

SELECT PRODUCT\_ID, ROUND (SUM (CASE WHEN QUARTER <=2 THEN TOTAL\_SALE END),2) AS FIRST\_HALF\_SALE,

ROUND (SUM (CASE WHEN QUARTER >2 THEN TOTAL\_SALE END),2) AS SECOND\_HALF\_SALE,

ROUND (SUM (CASE WHEN QUARTER <=4 THEN TOTAL\_SALE END),2) AS YEARLY\_SALE

FROM SALES

INNER JOIN TIME

USING (TIME\_ID)

WHERE YEAR =2016

GROUP BY PRODUCT\_ID

ORDER BY PRODUCT\_ID;



**-- 6TH QUERY**

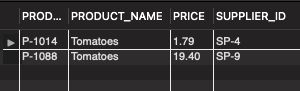
SELECT DISTINCT(PRODUCT\_ID), PRODUCT\_NAME, PRICE, SUPPLIER\_ID

FROM TRANSACTIONS

inner join MASTERDATA

using (PRODUCT\_ID)

where PRODUCT\_NAME= 'Tomatoes';



Anomaly:

The product “Tomatoes” has two different suppliers with different price so we have two different product ids for the same product which makes “Tomatoes” difficult to analyze.

**-- 7TH QUERY**

CREATE OR REPLACE VIEW STOREANALYSIS\_MV AS

SELECT STORE\_ID, PRODUCT\_ID, ROUND(SUM(TOTAL\_SALE),2) AS STORE\_TOTAL

FROM SALES

GROUP BY STORE\_ID, PRODUCT\_ID

ORDER BY STORE\_ID;



Shortcomings in Mesh Join:

1. Stream buffer can create a bottleneck as its size is fixed. It can overflow if we consider a stream of data coming.
2. When multiple partition of queue has same join id in hash table it becomes a challenge to differentiate between there tuples in the hash table when we are removing the tuple.
3. The no of partitions of master data should be equal to the partitions of the queue which has to be predefined which creates an overhead.
4. The time for one partition of transactional data to join with all of the master data is large.

What I learned:

I learned the following things through this project:

1. How to perform ETL process
2. Understanding and implementing Mesh Join algorithm
3. Building warehouse schema and loading data into warehouse
4. Analyzing data warehouse through OLAP queries