Bachelor of Science in Data Science

Course Code: CS4033

Course Name: Data Warehousing

Semester-Fall, 2022

Practical Project

Building and Analysing a Near-Real-Time Data Warehouse Prototype for METRO Shopping Store in Pakistan

Total Marks: 100

Weight in grade: 15%

Delivery date: 15-Nov-2022

1. Assessment task

The student has to design, implement, and analyse a near-real-time Data Warehouse (DW) prototype for METRO shopping store in Pakistan.

2. Project overview

METRO is one of the biggest superstores chains in Pakistan. The stores has thousands of customers and therefore it is important for the store to online analyse the shopping behaviour of their customers. Based on that the store can optimise their selling strategies e.g. giving promotions on different products.

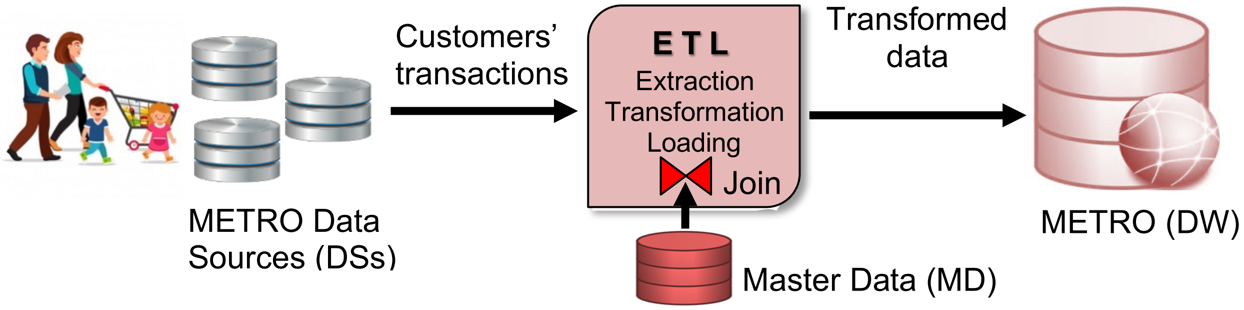

Figure 1: An overview of METRO DW

Now, to make this analysis of shopping behaviour 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. The overview of METRO DW is presented in Figure 1. 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 incomplete as it required by DW, it needs to complete in the transformation layer of ETL. For example enriching some information from Master Data (MD) as shown in Figure 2.

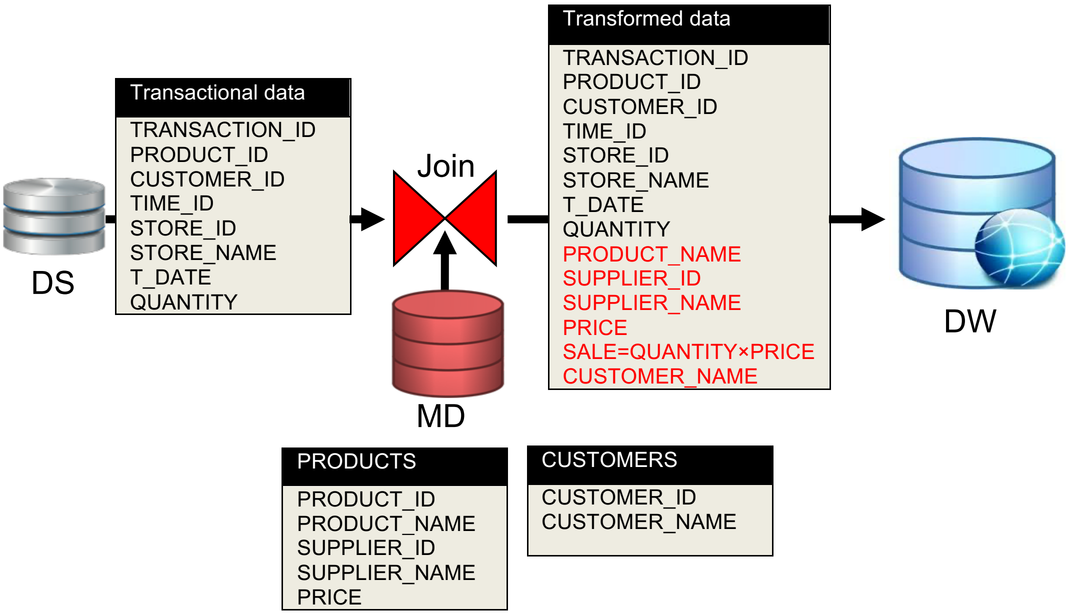

Figure 2: Enrichment example

To implement this enrichment feature in the transformation phase of ETL, we need a Stream-Relation join operator. There are a number of algorithms available to implement this join operation however, the seminal one is MESHJOIN (Mesh Join) which is explained in next section and you will implement its extended version in this project using **Java with Eclipse IDE**.

3. MESHJOIN (Mesh Join)

The MESHJOIN (Mesh Join) algorithm has been introduced by Polyzotis in 2008 with objective of implementing the Stream- Relation join operation in the transformation phase of ETL.

The main components of MESHJOIN are: **The disk-buffer** which will be an array and used to load the disk partitions in memory. Typically, MD is large, it has to be loaded in memory in partitions. Normally, the size of each partition in MD is equal to the size of the disk-buffer. Also MD is traversed cyclically in an endless loop. **The hash table** which stores the customers' transactions (tuples)**. The queue** is used to keep the record of all the customers' transactions in memory with respect to their arrival times. The queue has same number of partitions as MD to make sure that each tuple has joined with the whole MD before leaving the join operator. **The stream-buffer** will be an array and is used to hold the customer transaction meanwhile the algorithm completes one iteration.

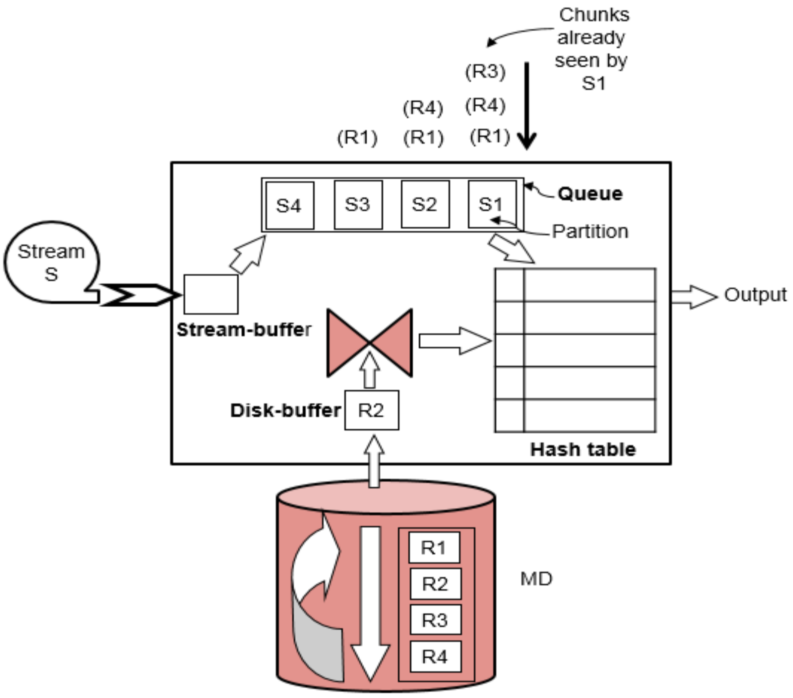

Figure 3: Working of MESHJOIN when R_2 is in memory but not yet processed

The crux of MESHJOIN is that with every loop step a new chunk of customers' transactions is read into main memory **(Hash table)** and MD partition in the disk-buffer is replaced by the new MD partition from the disk. Each of these chunks will remain in main memory for the time of one full MD cycle. The chunks therefore 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. In main memory, the incoming customers' data is organized in a queue, each chunk being one element of the queue. Figure 3 with four MD partitions shows a pictorial representation of the MESHJOIN operation: at each point in time, each chunk S_i in the queue has seen a larger number of partitions than the previous, and started at a later position in MD (except for the case that the traversal of MD resets to the start of MD). The figure shows the moment when partition R_2 of MD is read into the disk-buffer but is not yet processed.

After loading the disk partition into the disk buffer, the algorithm probes each tuple of the disk buffer in the hash table. If a matching tuple is found, the algorithm generates the join output. After each iteration the algorithm removes the oldest chunk of customers' transactions from the hash table along with their pointers from the queue. This chunk is found at the end of the queue; its tuples were joined with the whole of MD and are thus completely processed now.

4. Star-schema

The star schema (which you will use in this project) is a data modelling technique that is used to map multidimensional decision support data into a relational database. Star-schema yields an easily implemented model for multidimensional data analysis while still preserving the relational structures on which the operational database is built.

The star schema represents aggregated data for specific business activities. Using the schema, one can create multiple aggregated data sources that will represent different aspects of business operations. For example, the aggregation may involve total sales by selected time periods, by products, by stores, and so on. Aggregated totals can be total product sold, total sales values by products, etc. The basic star schema has three main components: *facts, dimensions, attributes,* and *classification levels.* Figure 2 can help you to determine the right components for your star schema.

5. Data specifications

The assessment provides a **MySQL** scripts file named "Transaction_and_MasterData_Generator.sql". By executing the script using MySQL database it will create two tables in your account. One is TRANSACTIONS table with 10,000 records populated in it. This data will be generated randomly based on 100 products, 50 customers, 10 stores, and one year time period as a date - from 01-Jan-17 to 31-Dec-17. The values for the quantity attribute will be random between 1 and 10. The other two tables named PRODUCTS and CUSTOMERS in MD with 100 and 50 records respectively..

6. Implementation of Extended MESHJOIN

To implement MESHJOIN algorithm you will implement the following steps using Java Eclipse.

1. Read a segment of stream from TRANSACTIONS table as an input data into the hash table with their join attribute values in the queue.
2. Load next MD partitions form the both MD tables into the disk buffers. After the last partition the next partition to read will be the first partition in each table.
3. Perform join operation between the MD tuples and stream tuples.
4. If the join is successful, enrich the required information to the transaction tuple. It is important to note that the attribute TOTAL_SALE does not exist in both TRANSACTION and MD.
5. The transaction tuples will then be loaded into DW. Make sure the dimension tables will not have the duplication records.
6. Repeat steps 1 to 5 until you load all the data from TRANSACTIONS table to DW.

7. DW analysis

Once the entire data has been loaded into DW, you will be required to analyse your DW by applying following OLAP queries.

Q1. Determine the top 3 store names who generated highest sales in September, 2017.

Q2. Find Top 10 suppliers that generated most revenue over the weekends. Just Explain how can we forecast the top suppliers for the next weekend?

Q3. Present total sales of all products supplied by each supplier with respect to quarter and month.

Q4. Present total sales of each product sold by each store. The output should be organised store wise and then product wise under each store.

Q5. Present the quarterly sales analysis for all stores using drill down query concepts.

Q6. Find the 5 most popular products sold over the weekends.

Q7. Preform ROLLUP operation to store, supplier, and product. Explain your query results in a few lines.

Q8. Extract total sales of each product for the first and second half of year 2017 along with its total yearly sales.

Q9. Find an anomaly in the data warehouse dataset. Write a query to show that anomaly and explain the anomaly in your project report.

Q10. Create a materialised view with name "STORE_PRODUCT_ANALYSIS" that presents store and product wise sales. The results should be ordered by store name and then product name. How the materialized view helps in OLAP query optimisation?

8. Tasks break-up

Following is list of tasks that you need to complete in this project.

1. Identifying appropriate dimension tables, fact table, and their attributes for the sales scenario presented in Figure 2. Based on that create a star-schema for DW with appropriate primary and foreign keys.
2. Implementing the MESHJOIN algorithm (using steps described in Section 6) in Java for successfully loading transactional data into DW after joining it with MD.
3. Applying of different analysis (described in Section 7) on DW using slicing, dicing, drill down, and materialising view concepts.
4. Writing a project report that should include project overview, schema for DW, MESHJOIN algorithm, any three shortcomings in Mesh Join, and what did you learn from the project?

9. What to submit

Each student has to submit the following files:

1. *createDW* –SQL script file to create star-schema for DW.
   Note: your script should remove the pre-existed schema.
2. *meshJoin* – an Eclipse based Java project that implements MESHJOIN algorithm.
   Note: You program should take the database credentials from the user at execution time.
3. *projectReport* – a doc file containing all contents described in point 4 under the tasks break-up section.
4. *readMe* – a text file describing the step-by-step instructions to operate your project.

Note: all above files need to submit in a zipped folder named by your family name, student ID, and assessment version e.g. Bilal-12345v1.

10. When to submit

Due date: **1st December 2022**

Late penalty: maximum late submissions time is 24 hours after the due date. In this case 5% late penalty will be applied.

11. Who to submit

The project should be submitted trough Google Class Room.

NOTE: Every student has to complete the project individually. Each student's project source and report materials should be unique and done by his/her own. All assessments will be assessed through turnitin system and in case of finding any duplication or identical material **0 marks will be marked**.

Marking guide

Project Component	Marks
createDW – SQL script file to create star-schema for DW	/15
The script should create all dimensions' and fact tables table in DW and if any table with same name exists already the script should drop that. It should also apply all primary and foreign keys on the right attributes.	
meshJoin – an Eclipse based Java project that implements MESHJOIN algorithm	/30
A Java file *meshJoin* that should implement all three phases of ETL – it should extract records from TRANSACTIONS table, transform these with MD and then load these records to DW successfully.	
queriesDW – SQL script file containing of all your OLAP queries	/30
The file should include OLAP queries for all tasks presented in Section 7.	
projectReport – a doc file containing all contents described in point 4 under the task break-up section.	/20
Report must contain project overview, schema for DW, MESHJOIN algorithm, any three shortcomings in Mesh Join, and what did you learn from the project?	
readMe – a text file describing the step-by-step instructions to operate your project	/5
readMe file should contain a step-by-step guide to operate the project. In case of missing this file, 5 marks will be deducted.	
Late submission penalty	-/5
TOTAL MARKS	/100

---------------------------E N D-------------------------