AUT UNIVERSITY COMPUTER + MATHEMATICAL SCIENCES

Master of Computer and Information Sciences

Assessment 2 – Version 01 – DW Project

Building and Analysing a DW for Countdown Stores in NZ

Paper: Data Warehousing and Big Data

Paper Code: COMP810

Semester-2, 2015

Weight in your final grade: 80%

Note: To pass the paper, student needs to attempt both assessments and obtain a C- grade overall.

1. Assessment task

Student has to design, implement and analyse a Data Warehouse (DW) for countdown, one of the biggest super market chains in NZ.

2. Project overview

Countdown is a one of the biggest superstores chains in NZ. The stores locate all over the country. Countdown has thousands of customers and therefore it is important for the organisation to analyse the shopping behaviour of their customers. Based on that the organisation can optimise their selling techniques e.g. giving of promotions on different products.

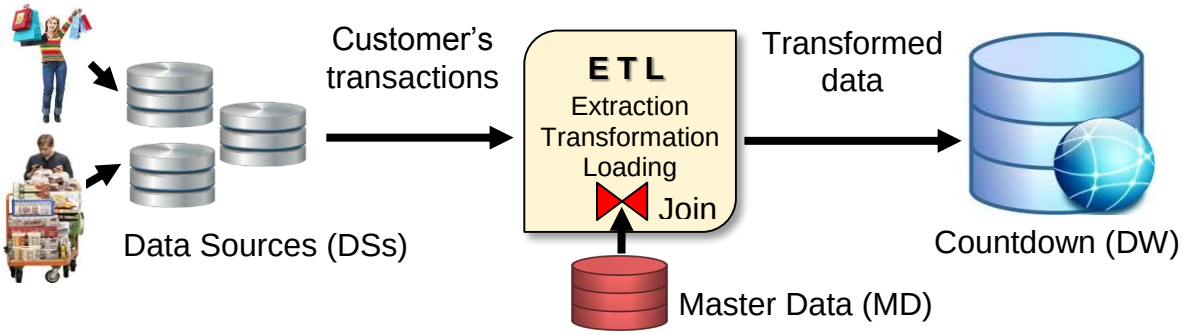

Figure 1: An overview of data integration in countdown scenario

Now, to make this analysis of shopping behaviour practical there is a need of building a DW and customers' transactions from Data Sources (DS) are required to reflect into DW on daily basis. This process of reflecting the customer data into DW is called Data Integration (DI) as shown in Figure 1. To implement DI we usually need 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 from Master Data (MD) as shown in Figure 2.

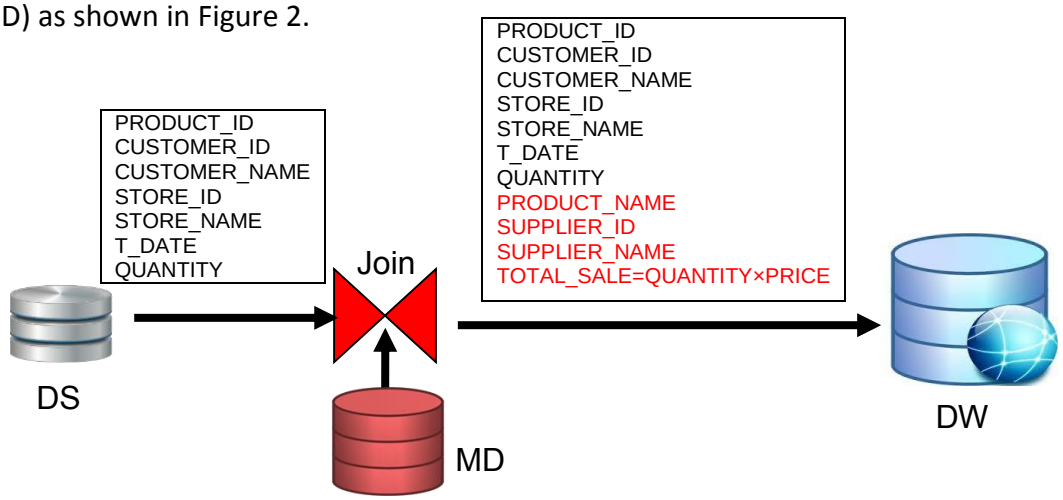

Figure 2: Enrichment example

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

3. MESHJOIN (Mesh Join)

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

The main components of MESHJOIN are: **The disk-buffer** which is 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** is used to hold the customer transaction meanwhile the algorithm completes one iteration. However, you don't need the stream buffer in this project as we are not considering the stream of customers' transactions.

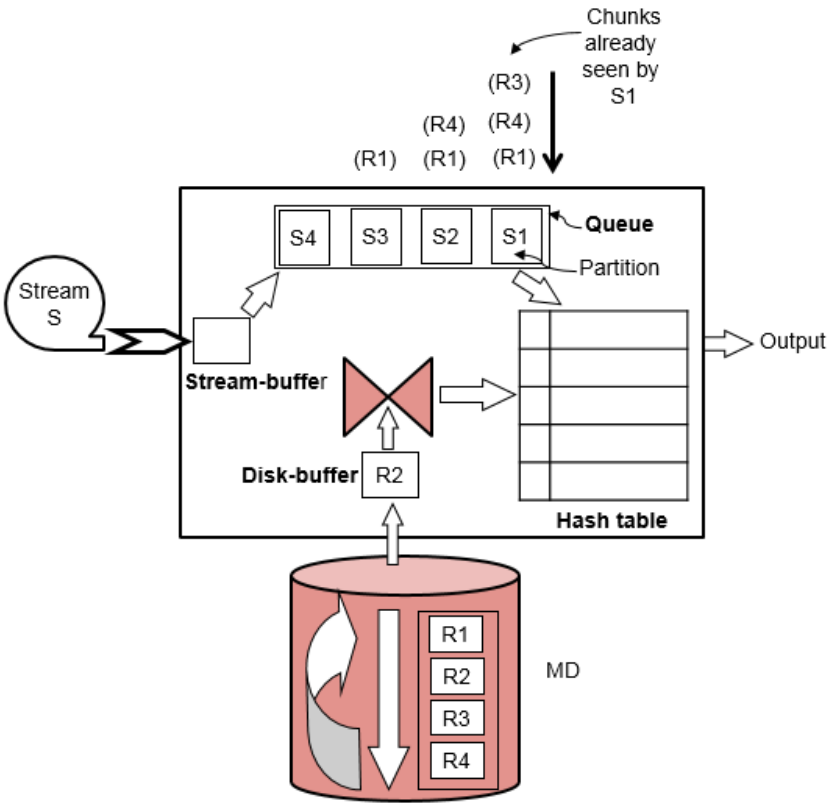

Figure 3: Working of MESHJOIN when R_2 is in memory but not yet processed

The crux of MESHJOIN is that with every loop step a new chunk of customers' transactions is read into main memory 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.* Usually in case of star-schema for sales the dimension tables are: *product*, *date*, *store,* and *supplier* while the fact table is *sales*. However, to determine the right attributes you will consider Figure 2.

5. Data specifications

The assessment provides a scripts file named "Transaction_and_MasterData_Generator.sql". By executing the script 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-14 to 31-Dec-14. The values for the quantity attribute will be random between 1 and 10. The other is MASTERDATA table with 100 records in it. The structure of both tables with their attributes name and data types is given below in Figure 4. The attributes TRANSACTION_ID and PRODUCT_ID are primary keys in TRANSACTIONS and MASTERDATA tables respectively.

TRANSACTIONS								
Attributes	TRANSACTION_ID	PRODUCT_ID	CUSTOMER_ID	CUSTOMER_NAME	STORE_ID	STORE_NAME	T_DATE	QUANTITY
Data type and size	VARCHAR2(8)	VARCHAR2(6)	VARCHAR2(4)	VARCHAR2(30)	VARCHAR2(3)	VARCHAR2(20)	DATE	NUMBER(3,0)

MASTERDATA					
Attributes	PRODUCT_ID	PRODUCT_NAME	SUPPLIER_ID	SUPPLIER_NAME	PRICE
Data type and size	VARCHAR2(6)	VARCHAR2(30)	VARCHAR2(5)	VARCHAR2(30)	NUMBER(5,2) DEFAULT 0.0

Figure 4: Structures for TRANSACTION and MASTERDATA tables

You will divide MD into 10 equal size partitions (10 tuples in each partition) and therefore, the total number of partitions in the queue (in MESHJOIN) will also be 10 while the size of each

partition (in terms of tuples' pointers) will be 50 as in each iteration the algorithm will load 50 tuples from transaction table to memory.

Following are the **credentials** that you need to connect to AUT Oracle database server.

1. You need to download Oracle JDBC connector from Oracle website or just use the following link: http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-112010-090769.html
2. Host name: oracle2.aut.ac.nz
3. Port: 1521
4. Database name: msdbs
5. User name: your_AUT_LoginID
6. Password: warehouse

6. Implementation of MESHJOIN

To implement MESHJOIN algorithm you will implement the following steps.

1. Read 50 tuples from TRANSACTIONS table as an input data into the hash table with their join attribute values in the queue.
2. Load next MD partition into the disk buffer. After the last partition the next partition will be the first partition.
3. Look-up each tuple from the disk buffer to the hash table.
4. If the tuple matches, add the required attributes (mentioned in Figure 2) into the transaction tuple.
5. The transaction tuple with new attributes will then be loaded into DW. Before loading the tuple into DW you will check whether the dimensions tables already contains this information. If yes then only update the fact table otherwise update both dimensions and fact tables.
6. After completing the look-up of all disk buffer tuples into the hash table, you need to remove all join attribute values from the last partition of the queue along with their transaction tuples from the hash table.
7. Repeat steps 1 to 6 until you load all the data from TRANSACTIONS table to DW.

7. DW analysis

Once the entire data has been loaded into DW, apply the following analysis to your DW using OLAP queries.

1. Which product generated maximum sales in Dec, 2014?
2. Which store produced highest sales in the whole year?
3. Determine the supplier name for the most popular product based on sales.
4. Presents the quarterly sales analysis for all stores using drill down query concepts.
5. Create a materialised view with name "STOREANALYSIS" that present the product-wise sales analysis for each store.

8. Tasks break-up

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

1. Identifying of appropriate dimension tables, fact table, and their attributes for the sales scenario presented in Figure 2. Based on that creating of star-schema for DW with appropriate primary and foreign keys. To keep the attribute name and their data types consistent in DW, consult the structure of tables TRANSACTIONS and MASTERDATA provided in Figure 4.
2. Implementing of the MESHJOIN algorithm and loading of 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 of project report that should include project overview, MESHJOIN algorithm, schema for DW, your OLAP queries with outputs 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 drop the table(s) if they already exist in the database.
2. *meshJoin* – Java file that implements the MESHJOIN algorithm
3. *queriesDW* – SQL script file containing all of your OLAP queries
4. *projectReport* – a doc file containing all contents described in point 4 under the task break-up section
5. *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. John-12345v1.

10. When to submit

Due date: Friday, 30th Oct 2015

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 autonline.

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 of any duplication or identical material the AUT cheating policy will be applied.

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

Appendix

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.	
Implementing of MESHJOIN	/30
Java file *meshJoin* 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	/35
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.	/15
Report must contains project overview, MESHJOIN algorithm, schema for DW, your OLAP queries with outputs 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.	
Late submission penalty	-/5
TOTAL MARKS	/100