AUT UNIVERSITY COMPUTER + MATHEMATICAL SCIENCES

Master of Computer and Information Sciences

Assessment 2 – DW Project

countdown

Building a Real-time 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

Each student has to design and implement a Real-time DW (RDW) 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 real-time shopping behaviour of their customers. Based on that the organisation can optimise their selling techniques e.g. giving of promotions on different products.

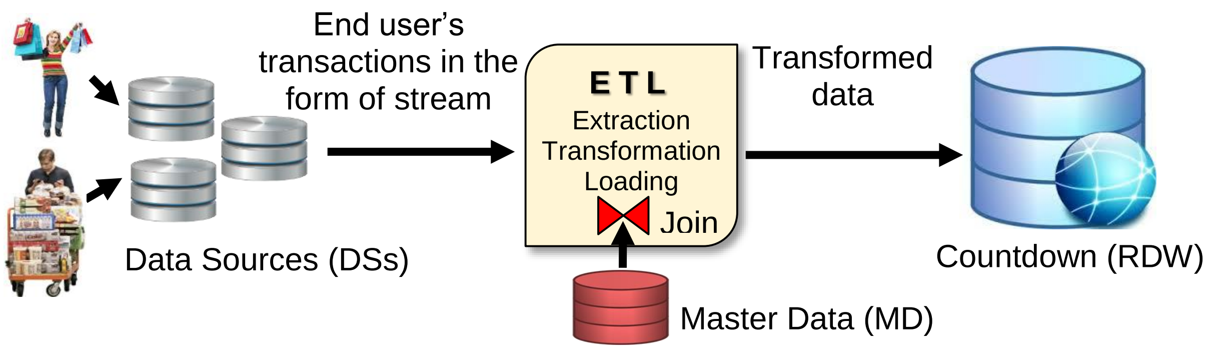

Figure 1: An overview of real-time data integration in countdown scenario

Now, to make this real-time analysis practical the customers' transactions need to reflect into RDW without any unnecessary delay – means as soon as a transaction happens on a Data Source(s) (DSs) it should be reflected into RDW. This process of reflecting the customer data into RDW is call Real-time Data Integration (RDI) as shown in Figure 1. To implement RDI we need real-time ETL (Extraction, Transformation, and Loading). Since the data generated by customers is not in the format required by RDW 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.

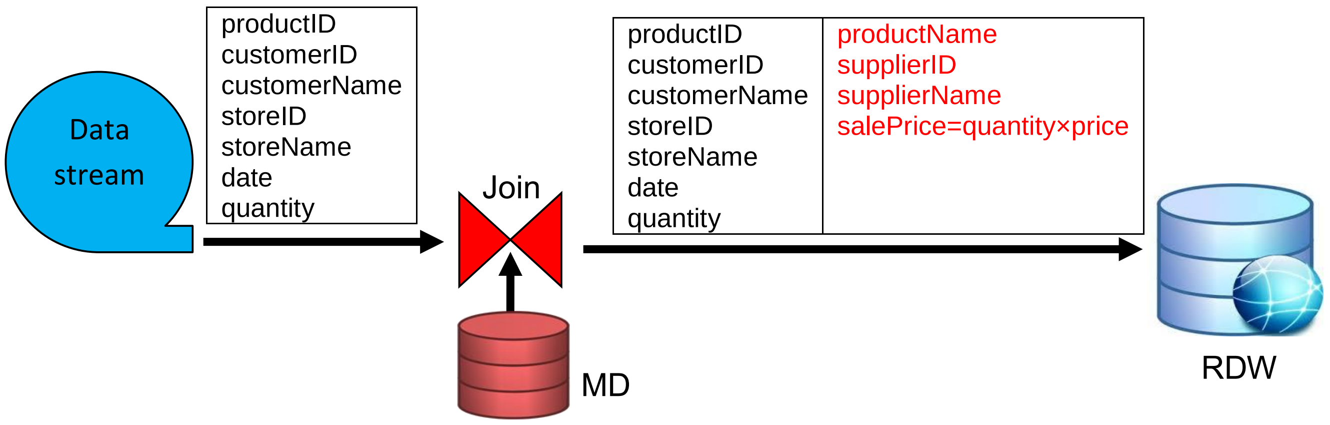

Figure 2: Enrichment example

To implement this enrichment feature in the transformation phase of ETL we need a join operator typically called Semi-Stream Join (SSJ). Also in RDW this operation needs to be performed as soon as the data receive from the data sources. In implementing the online execution of join, one important challenge is the different nature of both inputs for the join operator. The users' transactions are in the form of stream which is fast and huge in volume while the MD input is disk-based which is slow and expensive. The challenge here is to amortise this expensive disk access cost over the fast input stream. There are a number of algorithms have been introduced to address this challenge 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) – a Semi-Stream Join (SSJ)

The MESHJOIN (Mesh Join) algorithm has been introduced by Polyzotis in 2008 with the objective to amortise the slow disk access over fast stream data. The stream serves as the build input for a hash table and the disk-based MD serves as the probe input. MESHJOIN performs a staggered execution of the hash table build and the join is performed over many stream tuples in order to amortize disk access cost.

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 stream tuples in memory**. The queue** is used to keep the record of all stream tuples in memory with respect to their arrival times. The queue has same number of partitions as MD to make sure that each stream tuple has joined with the whole MD before leaving the join operator. **The stream-buffer** is used to accumulate the stream data meanwhile the algorithm completes one iteration. Usually the size of this component is very small e.g. 100KB in this project should be sufficient. The memory for each of above component can be calculated using a cost model which is also one of the tasks in this project.

The crux of MESHJOIN is that with every loop step a new chunk of stream tuples 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 stream 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 stream 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.

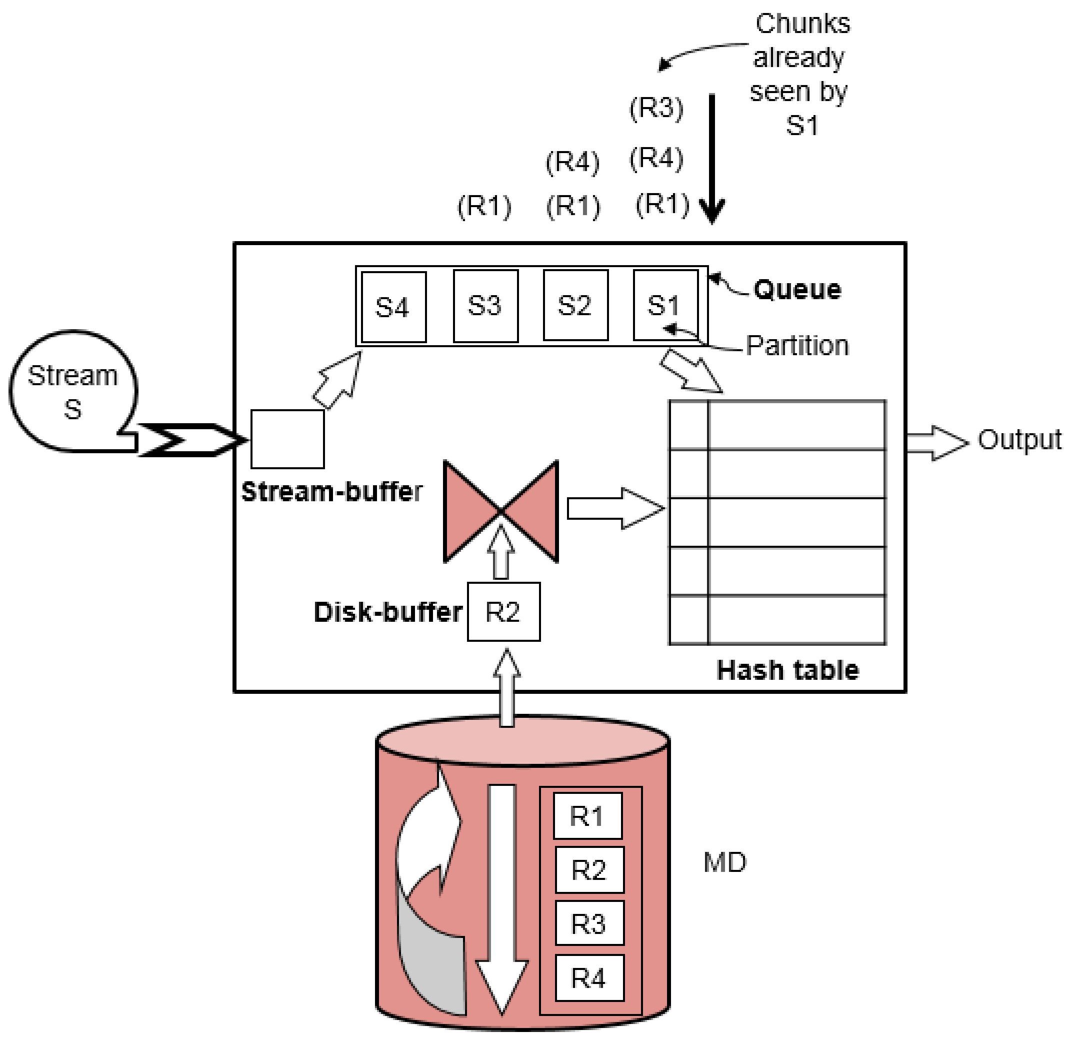

Figure 3: Working of MESHJOIN when R_2 is in memory but not yet processed

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 stream tuples 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 modeling 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. Credentials for accessing AUT Oracle database server

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. Data and memory specifications

You will select any 100 countdown products in this project. Therefore your stream and MD will be related to these 100 products. The sizes of each stream tuple and MD tuple can be different and depend on the number of attributes and their data types. Figure 2 will help you to select the attributes for both the stream data and MD. You will divide MD into 10 equal size partitions 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 depend on the total memory allocated for the algorithm. However, in this project you will allocate 5MB memory in total for all MESHJOIN components. Furthermore, the memory for each component can be determined using the cost model.

7. Tasks break-up

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

1. Creating and populating of MD
   You will create a MD table with name *masterData*. You will select appropriate attributes for the table based on Figure 2. The attribute *productID* would be a primary key in the table.
2. Identifying of appropriate dimension tables, fact table, and their attributes for the sales scenario presented in Figure 2 and based on that creating of star-schema for RDW with appropriate primary and foreign keys.
3. Implementing of following three independent threads using Java language:
   a. Generating of stream data – countdown shopping cart data. You will generate a stream of random sales transactions based on your selected 100 countdown product.
   b. Implementing of the MESHJOIN algorithm in the transformation phase of ETL
   c. Loading of data into RDW. Make sure your loading program only insets data in dimension tables if this doesn't exists before.

4. Applying OLAP queries on your RDW successfully. You will run a number of OLAP queries on your RDW to test the correctness of your star-schema for each dimension and fact tables.
5. Calculating of cost model (both memory and processing costs) for the MESHJOIN algorithm. The memory cost model should be generic and it should be able to calculate the cost for any memory settings. In processing cost model you will calculate the total cost in terms of time for all MESHJOIN operations. Based on the cost model you will also calculate the service rate for the MESHJOIN algorithm.
6. Writing of project report that should include project overview, MESHJOIN (algorithm and cost model), schema for RDW, your OLAP queries with outputs and what did you learn from the project?

8. What to submit

Each student has to submit the following files:

1. *createMD* – SQL script file to create and populate MD table, *masterData*
2. *createRDW* –SQL script file to create star-schema for RDW
   Note: for points 1 and 2 your scripts should drop the table(s) if they already exist in the database.
3. *streamGenerator* – Java file to generate stream data
4. *meshJoin* – Java file that implements the MESHJOIN algorithm
5. *dataLoader* – Java file that loads output of MESHJOIN into RDW
6. *queriesRDW* – SQL script file containing of all your OLAP queries
7. *projectReport* – a doc file containing all contents described in point 6 under the task break-up section
8. *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 and student ID e.g. John-12345.

9. 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.

10. 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 (indicative)**
createMD – SQL script file to create and populate MD	/5
The script should create and populate *masterData* table and if this table exists already the script should drop that. It should also apply primary key on the right attribute.	
createRDW –SQL script file to create star-schema for RDW	/15
The script should create all dimensions' and fact tables table in RDW 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.	
streamGenerator – Java file to generate stream data	/10
The program should generate a continuous stream of random shopping cart transactions from the selected list of 100 products.	
meshJoin – Java file that implements the MESHJOIN algorithm	/30
All components of the MESHJOIN algorithm should be implemented correctly. The assigning of memory to each component should be accurate based on the cost model.	
dataLoader – Java file that loads output of MESHJOIN into RDW	/10
By running this file the output of the MESHJOIN algorithm should be loaded into RDW correctly. Also the loading program only insets data in dimension tables if this doesn't exists before.	
queriesRDW – SQL script file containing of all your OLAP queries	/10
This should include a sufficient number of OLAP queries to critically test the correctness of star-schema i.e. all dimension and fact tables and their relationships.	
projectReport – a doc file containing all contents described in point 6 under the task break-up section	/15
Report must contains project overview, MESHJOIN (algorithm and cost model), schema for RDW, 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