

Assessment 2-DW PROJECT

Building and analysing a DW for BigFoods Stores in NZ



October 28, 2016

15891556

Sheng Wang

# 

# Project Overview

The data warehouse is built for BigFoods that is one of the biggest supermarket in NZ. It is very important for BigFoods to analyse behaviours of customers because there are thousands of customers. It can improve selling skills by analysing data from data warehouse.

In order to build the data warehouse, star scheme involving the fact table and dimensional tables are created for specific business activities. In order to input data into data warehouse, INLJ algorithm is implemented to fulfil the feature by the join operator called Semi-Stream Join(SSJ). The data from TRANSACTIONS table needs to be processed in three layers of ETL (Extraction, Transformation and Loading). Every time INLJ reads 50 tuples from transaction table as input data into a cursor. Each tuple in the cursor and relevant attributes in Masterdata table are read by Product\_ID. All records in dimensional tables are not duplicated by checking records of the table before inputting data.

# Star-scheme

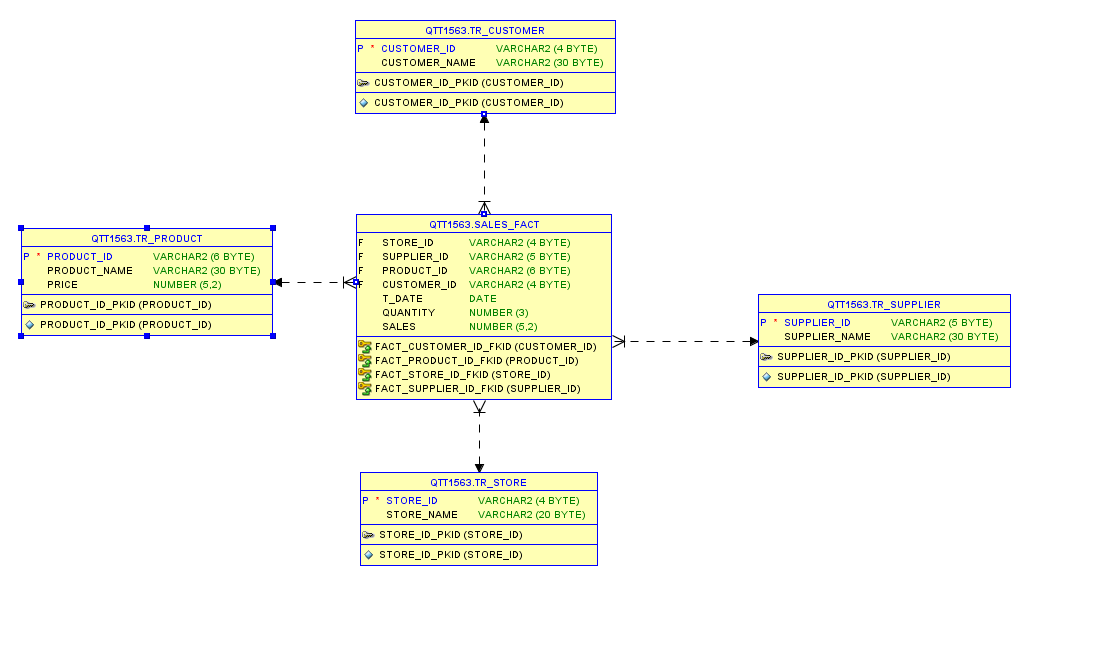


Figure 1: star scheme

The data warehouse is built according to principles of star scheme as Figure 1 shows above to create tables including fact tables and dimensional tables. The fact table mainly stores IDs which are also constraints from other dimensional tables and also the most important value –total sales in terms of date, stores, suppliers and customers. The first step is to drop all tables to make sure there are no more tables with the same name existing in data warehouse. What needs to be noticed is drop fact table firstly because it has not only primary-key constraints but also foreign-key constraints which define their relationship to other dimensional tables. What is more, fact table needs to be created at last because foreign-key constraints determine some other relational tables are connected to it and these dimensional tables need to be created firstly. Transaction table and date table are not necessary because in my perspective, transaction table just stores transactional ID and no more value can be stored inside and date table may store date ID and data just like customer table, but there is no date ID in transaction table and masterdata table.

# INLJ algorithm

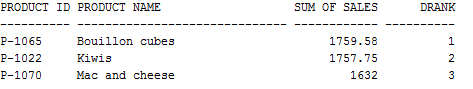
INLJ algorithm is created by PL/SQL language and the cursor is used to receive all each rows from transactions table. The concept of ETL (Extraction, Transformation, and Loading) is implemented to fulfil the feature. There are two loops to run it in INLJ algorithm. The outer loop is used to input limited 50 tuples from a cursor to index variable TRAN\_ATT whose type is rows of the transaction table. In the inner loop from the first one to the last one (50), variable R is declared as each one in this cycle and attributes of each one can be called by being added TRAN\_ATT(R).

Because one user can purchase different products, one store can sell different products and so on, there will be many same rows when data is extracted to dimensional tables. V\_COUNT is declared to monitor the whole process when data is input into dimensional tables. Select………Into……… is unique syntax in PL/SQL and clause “where” is used to query all rows in dimensional tables and if it finds the same row, count (\*) which is one will be assigned to V\_COUNT. If no rows are found, count (\*) which is 0 will be assigned to V\_COUNT. By using clause “if” to identifying V\_COUNT, the data is inserted if it is 0. The data will be refused to insert into dimensional tables if V\_COUNT is 1.

# OLAP Queries

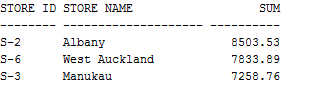
According to assignment requirements, there are many valuable sales data that need to be generated in terms of various periods of time, different stores and customers.

*Q1a Determine the top 3 products in Dec 2014 in terms of total sales:*



Firstly, select attributes from other relative table such as PRODUCT\_NAME and meanwhile use “group” to calculate sum value of these specified products, PRODUCT\_ID and PRODUCT\_NAME need to be added in “GROUP BY” otherwise error will be generated. DENSE RANK () is used to order all displayed rows and select top 3 rows which are answers.

*Q1b Determine the top 3 stores in Dec 2014 in terms of total sales*



This question is similar to Q1a, but I use ROWNUM () to solve the question. There are no differences between ROWNUM () and DENSE RANK () to solve the question because there is no the same value.

Q2 Determine which store produced highest sales in the whole year?



This is a similar question to two previous questions (Q1a, Q1b).

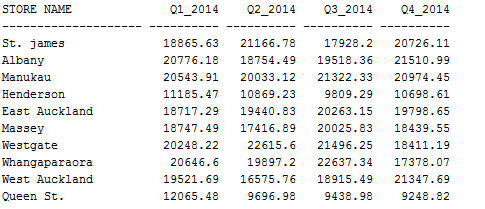
Q3 How many sales transactions were there for the product that generated maximum sales

revenue in 2014? Also identify the a) product quantity sold and b) supplier name

The question is based on the previous one(Q2). What is important in my perspective is how to display all data in one row by using two sub tables (T1, T2) which are generated and related to each other by using the same ROWNUM.

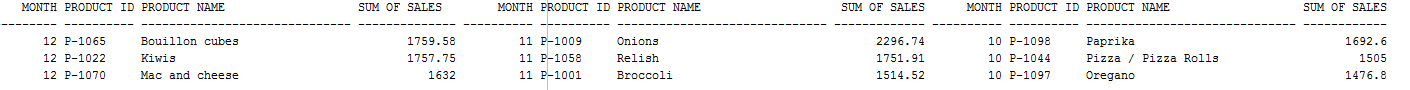
Q4 Present the quarterly sales analysis for all stores using drill down query concepts, resulting

in a report.



Firstly, divide the year into four quarters using LIKE and four quarter sub tables (T1, T2, T3, T4) are generated. In order to connect four tables, ROWNUM ID needs to be declared just like the previous question.

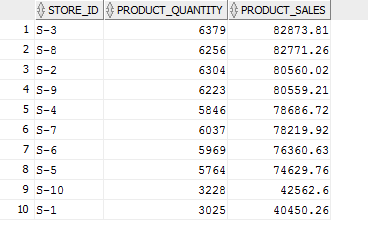
Q5 Determine the top 3 products for a particular month (say Dec 2014), and for each of the 2 months before that, in terms of total sales



Firstly, set a parameter ( :MONTH) to receive input from user. Then use left outer join to make sub tables T1, T2, T3 connect each other to make top 3 products in each three months in terms of total value to display in the first three rows.

Q6 Create a materialised view with name “STOREANALYSIS” that presents the productwise

sales analysis for each store.



# The materialised view called STOREANALYSIS is used to extract information from SALES\_FACT to generate total value of each store in 2014 and these value are also ordered by descent way in order for business analysis.

# What I learnt from the project

After creating a complete data warehouse, I have sound knowledge in terms of Star Scheme, INLJ and OLAP queries. Data warehouse is made of star scheme consisting of fact table and dimensional tables. The fact table is used to store primary keys like ID from all other dimensional tables and important value like total sales and dimensional tables are used to store unnecessary data like product name or store name. When we create tables, we need to pay attention to the order of creating these because of existence of constraints. Last but not least, we can’t drop dimensional tables before the fact table because there are constraints that are related to dimensional tables in the fact table.

After I create the Star Scheme, I get more familiar with syntax of PL/SQL and deeper concept of ETL (Extraction, Transformation, and Loading) and Semi-Stream Join(SSJ). Firstly, I figure out how to define data type in PL/SQL and how to put a limited number of data into cursor in outer loop from the data source. In order to avoid duplicating data in dimensional tables in the inner loop, we need to figure out how to set up a parameter to check if it already exists in dimensional tables. Because data in the fact table is unique, there is no need to set parameter to check if it is duplicated before I insert data into the fact table. What is more, data from Transaction table needs to be merged with MASTERDATA table by finding the same attribute (PRODUCT\_ID) as an index in both tables to input relevant attributes into dimensional tables and the fact table.

After I completed OLAP queries, I have deeper knowledge of DRILLDOWN and ROLLUP concepts to generate results of specific total value according to assignment requirements. I also get more familiar with slicing, dicing, drilldown and materialising view concepts. I also understand differences between View and Materialized View and how, when, why to use the Materialized View. In order to review and modify code later, OLAP queries need to be concise in the formal format. Naming a table is a good option because it is easy to invoke it in other places such as SALES\_FACT A. For some specific problems, some data in different tables need to be displayed in the same row by using ROWNUM for sub tables. What I also learnt from the OLAP queries are to distinguish between RANK and DENSE RANK and ROWNUM to get top products or stores that can create the biggest profit under specified period of time. Owing to huge coverage of last questions, it is a good choice to divide into small tasks. For example, when I was doing Q4, I split into small tasks to generate total sale of all products in the first season. When I was doing Q5, I need to define a parameter to receive an input. “EXTRACT” is also what I learnt to solve this question to get the month from the data from the fact table.