CSE 601 Project 1: Report

Data WareHouse/OLAP System

Team Members:

Lingbing Song
UBIT: lingbing
Person #: 50169024

Xiangtong He
UBIT: xiangton
Person #: 36593830

Le Fang
UBIT: lefang
Person #: 50166024

Data WareHouse/OLAP System

1 Introduction

A *Data Warehouse* (DW) is a central repository for integrated data collected from a vast body of disparate data sources by using Extraction-Transformation-Loading (ETL) process. To extract data from various sources, DW can be physically utilized for storing and/or copying various types of data and reporting data analysis in an intelligent manner. There are two major design approaches for DW systems, i.e., Bottom-up design and Top-down design. In the Bottom-up approach, data marts are created first and could then be integrated into a comprehensive DW. In contrary, Top-down approach leads to the unavailability of data marts for users until creating completely structured DWs. It is well known that *Online Analytical Processing (OLAP)* has been widely used to process transactions in a large volume by means of much less complex queries, which is in contrast to the traditional *Online Transaction Processing (OLTP)*. OLAP is a superior approach to access the aggregated data stored in a multidimensional data model and answer multi-dimensional analytical (MDA) queries swiftly. In general, the research and developement on modern intelligent data-mining and decision support systems has been centered on improving more efficient DW integrated OLAP technique since the optimized DW architectures allows the data scientists to achieve excellent query performance, organize and disambiguate redunant data, keep the data information consistently and etc.

In this project, we need to implement a clinical DW based on a modified schema. The original data files are imported into SQLite database system by using SQL in Python and the relationship between different tables has also been created. There are five data space, including Clinical data space, Sample data space, Microarray and proteomic data space, Experiment data space and Gene data space. Accordingly, a large number of biomedical information for 6 diseases and 60 patients should be retrieved from the original data files. Here SQL language with Python will be utilized to implement OLAP operations and carry out the corresponding caculation under t-statistics and F-statistics.

2 Schema Design

The DW for this project are primarily based on BioStar schema. To improve the efficiency of data analysis, the shared key elements, e.g., ds_id, p_id, pb_id, s_id, as well as some important elements, e.g., name, from the aforementioned 5 fact tables should be considered simutaneously. Since it is not necessary to join every secondary table associated with the fact table with respect to the different data spaces, we adopted both query and subquery to

intelligently reconstruct the corrsponding JOIN operation among tables. This modification would help increase the efficiency on collecting targeted relationship among different tables with some in bioStar schema. The aforementioned efficient queries can be regarded as an equivalence to the newly created tables with the modfied schema constiuted by the shared key elements. In the following part, we are going to explain this by using the following SQL.

```
SELECT Clinical_Fact.p_id, Microarray_Fact.exp, Gene_Fact.cl_id
FROM Clinical_Fact JOIN Microarray_Fact JOIN Probe JOIN Gene_Fact
ON Microarray_Fact.s_id = Clinical_Fact.s_id and Microarray_Fact.pb_id = Probe.pb_id
 and Probe.UID = Gene_Fact.UID
WHERE Microarray_Fact.mu_id = "001" and Gene_Fact.cl_id = "00002" and
Clinical_Fact.p_id in
(SELECT Clinical_Fact.p_id
FROM Disease JOIN Clinical_fact
ON Disease.ds_id = Clinical_fact.ds_id
WHERE Disease.name = "ALL" )
ORDER BY Clinical_Fact.p_id
```

This queries is specifically designed for Part 2 Q.3, which can effectively identify the experiental values for each sample of patients. Here subquery corresponds to the Patient.p_id that is related to ALL in equivalent to a new Table “Patient with ALL”, the major query part can be considered as another new Table “Microarray_Value” (as shown in Fig. 1) both of which share the p_id. In addition, we do not perform “JOIN” operation for Cluster, Gene, Measurementunit and Patient Table. This does improve the efficiency of data mining.

Microarray_Value table

p_id	exp

Patient with ALL table

p_id	Disease name
	ALL

Figure 1. JOIN this two newly designed table can be virtually implemented by our SQL.

Complexity: Here we make an analysis on the efficiency of our queries for the table Microarray_fact with m rows and n colomns. The origianl time complexity will be $O(m^n)$ in the worse case. When we consider the table Microarray_value and perform JOIN operation, the correspondingcomplexity will be reduced to $O(m^2)$ in the worse case. However, the

complexity can be further improved close to *O(m)* by using a hash table to look up p_id in Microarray_Value table based on the same p_id in the Patient with ALL table.

3 Results

3.1 part II:

Q1

- The number of patients who had "ALL" is **13**
  SELECT COUNT(Clinical_Fact.p_id) FROM Clinical_Fact WHERE Clinical_Fact.ds_id in (select Disease.ds_id from Disease where Disease.name = "ALL")
- The number of patients who had "leukemia" is **27**
  SELECT COUNT(Clinical_Fact.p_id) FROM Clinical_Fact WHERE Clinical_Fact.ds_id in (select Disease.ds_id from Disease where Disease.type = "leukemia")
- The number of patients who had "tumor" is **53**
  SELECT COUNT(Clinical_Fact.p_id) FROM Clinical_Fact WHERE Clinical_Fact.ds_id in (select Disease.ds_id from Disease where Disease.description = "tumor")

Q2. List the types of drugs which have been applied to the patients with "tumor":
['Drug Type 001', 'Drug Type 002', 'Drug Type 003', 'Drug Type 004', 'Drug Type 005', 'Drug Type 006', 'Drug Type 007', 'Drug Type 008', 'Drug Type 009', 'Drug Type 010', 'Drug Type 011', 'Drug Type 012', 'Drug Type 013', 'Drug Type 014', 'Drug Type 015', 'Drug Type 016', 'Drug Type 017', 'Drug Type 018', 'Drug Type 019', 'Drug Type 020']

Q3. For each sample of patiens with "ALL", list the mRNA valus of probes in cluster is "00002" for each experiment with measure unit id = "001". The total number is 325 values, and the corresponding details can be shown after running our code.

Q4. For probes belonging to GO with id = "0012502", caculate the t statistics of teh expression values between patients with "ALL" and without "ALL": **-1.00712677668**, the corresponding p value is **0.314065698191**

Q5. For probes belonging to GO with id = "0007154", caculate the F statistics of the expresssion values among patients with "ALL", "AML", "colon tumor" and "breast tumor" is **3.13891213105**, the corresponding p value is **0.024681499479**

Q6. For probes belonging to Go with id = "0007154", calculate the average correlation of the expression values between two patients with "ALL", and caculate the average correaltion of the expresssion values between one "ALL" patient and one "AML" patient.

- *The average correlation of the expression values between two patients with "ALL" is* **0.143544347502**

- *The average correlation of the expression values between one "ALL" patient and one "AML" patient is* **-0.00347560083193**

3.2 Part III:

Q1. Given a specific disease, find the informative genes (for example taking "ALL"):
Inforamtive Genes are shown below (38 in total):
[**'0001433276', '0004826120', '0011333636', '0013947282', '0015295292', '0016073088', '0018493181', '0021633757', '0024984526', '0028863379', '0031308500', '0031997186', '0037998407', '0040567338', '0041333415', '0041464216', '0043866587', '0045926811', '0047276861', '0048199244', '0052948490', '0053478188', '0058672549', '0058792011', '0060661836', '0065772884', '0069156037', '0074496827', '0075434512', '0075492172', '0083398521', '0085557586', '0087592194', '0088257558', '0088596261', '0092443312', '0094113401', '0097606543'**]

Q2. Use informative genes to classify a new patient (considering five test cases in test_samples.txt)
Test Results are shown below:
Test1 can be classified as ALL
Ttest_indResult(statistic=18.475170049327343, pvalue=3.019397577627644e-24)

Test2 can be classified as ALL
Ttest_indResult(statistic=6.508247454372413 4, pvalue=3.2547484669054254e-08)

Test3 can NOT be classified as ALL
Ttest_indResult(statistic=-0.28923968846596682, pvalue=0.77357051847198743)

Test4 can be classified as ALL
Ttest_indResult(statistic=19.160580316731863, pvalue=5.9265057752048745e-25)

Test5 can be classified as ALL
Ttest_indResult(statistic=-3.0309549645491964, pvalue=0.0038238124509267775)

4 Reference

1. *Student's t-test, https://en.wikipedia.org/wiki/Student%27s_t-test*
2. *Data warehouse, https://en.wikipedia.org/wiki/Data_warehouse*
3. *OLAP, https://en.wikipedia.org/wiki/Online_analytical_processing*
4. *Scipy statstical documentation, http://docs.scipy.org/doc/scipy/reference/stats.html*
5. *Python sqlite3 documentation, https://docs.python.org/3.5/library/sqlite3.html*