**Lab 2 – Group Lab (100 points)**

**Part 1: Practice on subquery**

Use university data as it is used in Lab 1. Please fill in the SQL and Result columns of the following table for each query. Please submit this duly filled template to the dropbox by midnight of the due date.

Query (20 points) SQL Result

1. Find the age of the oldest student who is either a History major or enrolled in a course taught by ‘I. Teach’.

select max(s.age) from student s where s.age in

((select s.age from student s where s.major = "History")

union ( select s.age from student s, enroll e, class c, faculty f where s.snum=e.snum and e.cname=c.cname and c.fid=f.fid and f.fname="I. Teach"));

+------+

| age |

+------+

| 20 |

+------+

2. Find the names of all classes that either meet in room R128 or have five or more students enrolled.

(select e.cname from enroll e group by e.cname having count(e.snum)>=5) union (select c.cname from class c where c.room='R128');

+--------------------------+

| cname |

+--------------------------+

| Database Systems |

| Operating System Design |

| Archaeology of the Incas |

| Dairy Herd Management |

| Data Structures |

| Intoduction to Math |

| Patent Law |

+--------------------------+

3. Find the names of all students who are enrolled in two classes that meet at the same time.

select s.sname from student s, enroll e, class c where s.snum = e.snum and e.cname=c.cname and c.cname in

(select cname from class c where meets\_at in ( select meets\_at from class c group by c.meets\_at having count(\*)>=2))

group by s.sname having count(\*)>1;

+-----------------------+

| sname |

+-----------------------+

| Luis Hernandez |

+-----------------------+

4. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

select f.fname from faculty f where f.fid in

( select f.fid from faculty f, class c, enroll e where f.fid=c.fid and c.cname =e.cname group by f.fid having count(c.fid)<5);

+------------------+

| fname |

+------------------+

| John Williams |

| Elizabeth Taylor |

| Mary Johnson |

| William Moore |

| Patricia Jones |

| Richard Jackson |

| Jennifer Thomas |

+------------------+

**Practice on Stored procedure (10 point each)**

1. (i) Write a stored procedure to return the number of faculty from Faculty table.

delimiter $$

drop procedure if exists getTotalFaculty;

create procedure getTotalFaculty(INOUT total INT)

begin

select count(\*) into total

from faculty;

end $$

delimiter;

(ii) Write Java code to get call this procedure from Java and display the total number of faculty from Java console.

SEE CODE AT BOTTOM OF PAGE.

1. (i) Write a stored procedure to return information about a student, given a student id.

delimiter $$

drop procedure if exists getStudentById;

create procedure getStudentById(INOUT studentList varchar(4000), IN studentID INT)

begin

declare isDone integer default 0;

declare VARsname varchar(255) default "";

declare VARmajor varchar(255) default "";

declare VARlevel varchar(255) default "";

declare VARage varchar(255) default "";

declare studentCursor cursor for

select sname, major, level, age from student

where snum = studentID;

declare continue handler

for not found set isDone = 1;

open studentCursor;

getList: loop

fetch studentCursor into VARsname, VARmajor, VARlevel, VARage;

if isDone = 1 then

leave getList;

end if;

set studentList = concat("","\n", studentList);

set studentList = concat(VARage, studentList);

set studentList = concat(VARlevel,", ", studentList);

set studentList = concat(VARmajor,", ", studentList);

set studentList = concat(VARsname,", ", studentList);

end loop getList;

set studentList = concat("Name, Major, Standing, Age","\n", studentList);

close studentCursor;

end $$  
delimiter ;

(ii) Write Java code to ask users to enter a student id, call this procedure and display the information on Java console.

SEE CODE AT BOTTOM OF PAGE.

1. (i) Write a stored procedure using cursor to print out information on all classes.

delimiter $$

drop procedure if exists getClasses;

create procedure getClasses(INOUT classList varchar(4000))

begin

declare isDone integer default 0;

declare VARcname varchar(255) default "";

declare VARmeets\_at varchar(255) default "";

declare VARroom varchar(255) default "";

declare classCursor cursor for

select cname, meets\_at, room from class;

declare continue handler

for not found set isDone = 1;

open classCursor;

getList: loop

fetch classCursor into VARcname, VARmeets\_at, VARroom;

if isDone = 1 then

leave getList;

end if;

set classList = concat("","\n", classList);

set classList = concat(VARroom, classList);

set classList = concat(VARmeets\_at,", ", classList);

set classList = concat(VARcname,", ", classList);

end loop getList;

set classList = concat("Class Name, Times, Room","\n", classList);

close classCursor;

end $$  
delimiter ;

(ii) Write Java code to call this procedure from Java

SEE CODE AT BOTTOM OF PAGE.

(iii) Write a SQL statement to print out all information on all classes

SELECT cname, meets\_at, room

FROM class;

(iv) Write Java code to get this SQL statement executed.

SEE CODE AT BOTTOM OF PAGE.

Compare the approach 1 which contains the steps (i) and (ii) and approach 2 which contains the step (iii) and (iv). Please give your comments/thought on which situations which approach should be used.

Approach 1 is much easier to implement on the Java side of things, but is much more difficult to implement on the SQL side. Approach 2 is more difficult on the Java side, in terms of the code to interpret and show the results, but it is easier to write the SQL query on the Java side. I think that in most scenarios it makes more sense to use the Approach 2 because it is easier for Java developers to comprehend how to use, but Approach 1 would be better to use if the Java developers are not supposed to know about field names in the database.

All the stored procedure, screenshots (Java, mysql) should be stored/cut-pasted to a Word document and submit to Lab 2 dropbox by the due date.

**JAVA CODE:**

package MySQLDemo;

import java.util.Scanner;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.ResultSetMetaData;

import java.sql.SQLException;

import java.sql.Statement;

import java.sql.CallableStatement;

import java.sql.Types;

public class storedProcedures {

static final String databasePrefix ="cs366-2211\_paprockisj04";

static final String netID ="paprockisj04";

static final String hostName ="washington.uww.edu";

static final String databaseURL ="jdbc:mariadb://"+hostName+"/"+databasePrefix;

static final String password="sp3619";

private Connection connection = null;

private Statement statement = null;

private ResultSet resultSet = null;

public void Connection(){

try {

Class.forName("org.mariadb.jdbc.Driver");

System.out.println("databaseURL"+ databaseURL);

connection = DriverManager.getConnection(databaseURL, netID, password);

System.out.println("Successfully connected to the database");

}

catch (ClassNotFoundException e) {

e.printStackTrace();

}

catch (SQLException e) {

e.printStackTrace();

}

}

public void storeProcedure(String procedure) {

try {

statement = connection.createStatement();

int total =0;

CallableStatement myCallStmt = connection.prepareCall("{call " + procedure + " (?)}");

myCallStmt.registerOutParameter(1,Types.BIGINT);

myCallStmt.execute();

total = myCallStmt.getInt(1);

System.out.println("The procedure " + procedure + " returned: "+ total);

}

catch (SQLException e) {

e.printStackTrace();

}

}

public void storeProcedure(String procedure, String input) {

try {

statement = connection.createStatement();

CallableStatement myCallStmt;

String result = "";

if(input == "") {

myCallStmt = connection.prepareCall("{call " + procedure + " (?)}");

}else {

myCallStmt = connection.prepareCall("{call " + procedure + " (?," + input + ")}");

}

myCallStmt.setString(1,"");

myCallStmt.registerOutParameter(1,Types.VARCHAR);

myCallStmt.execute();

result = myCallStmt.getString(1);

System.out.println("The procedure " + procedure + " returned: "+ result);

}

catch (SQLException e) {

e.printStackTrace();

}

}

public void simpleQuery(String sqlQuery) {

try {

statement = connection.createStatement();

resultSet = statement.executeQuery(sqlQuery);

ResultSetMetaData metaData = resultSet.getMetaData();

int columns = metaData.getColumnCount();

for (int i=1; i<= columns; i++) {

System.out.print(metaData.getColumnName(i)+"\t");

}

System.out.println();

while (resultSet.next()) {

for (int i=1; i<= columns; i++) {

System.out.print(resultSet.getObject(i)+"\t\t");

}

System.out.println();

}

}

catch (SQLException e) {

e.printStackTrace();

}

}

public static void main(String args[]) {

storedProcedures procObj = new storedProcedures();

procObj.Connection();

String procedure1 = "getTotalFaculty";

String procedure2 = "getStudentById";

String procedure3 = "getClasses";

String studentID = "51135593";

procObj.storeProcedure(procedure1);

Scanner myObj = new Scanner(System.in);

System.out.println("Enter Student ID: ");

studentID = myObj.nextLine();

procObj.storeProcedure(procedure2, studentID);

procObj.storeProcedure(procedure3, "");

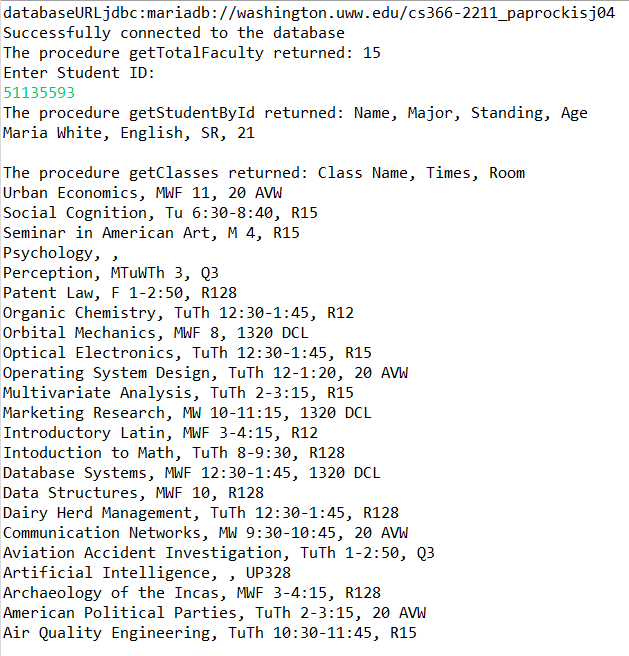
String sqlQuery = "SELECT cname, meets\_at, room FROM class;";

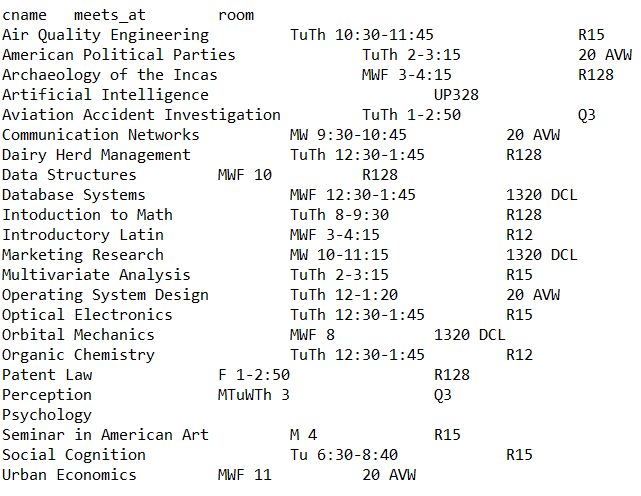
procObj.simpleQuery(sqlQuery);

}

}

**JAVA CONSOLE SCREENSHOTS:**





**Part 2 (50 points):**

Do research on one of the following DBMS: Oracle, DB2, Microsoft SQL Server. You will be doing research using google, books, articles on one of these three DBMS (choose one). Your deliverable is at least a 2 page report in which you discuss the following points on each DBMS:

1. Overview
2. DBMS Installation and Setup
3. Data type, support for built-in functions
4. Storage support
5. Indexing
6. Security