RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical and Computer Engineering

Program: Computer Engineering

Course Number	**CPS 510**
Course Title	**Database Systems I**
Semester/Year	**Fall 2021**
Instructor	**Dr. A. Abhari**

Assignment NO.	**9**

Report Title	GUI AND UI MENU SHELL

Section No.	01
Group No.	58
Submission Date	2nd December, 2021
Due Date	2nd December, 2021

Name	Student No	Signature
Naga Sreeja Kurra	500920707	K.N.S
Raiyyan Siddiqui	500892699	R.S
Ethan Ting	500894358	E.T

**By signing above you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at:*

http://www.ryerson.ca/senate/policies/pol60.

DESCRIPTION

USER INTERFACE (Mac Terminal)
This assignment is based on the menu unix shell. The code from the D2L posted from the A5 template shell has been executed using the mac terminal and connecting it to the VPN to the Ryerson.

The Menu shell works on Drop table, Create Table, Populate Table and View Table, and other commands like exit and force quit for the SQL Developer.
The queries have been entered according to the table in their respective places.

Below is the following code for the menu shell:

```
#!/bin/sh
MainMenu() {

   while [ "$CHOICE" != "START" ]
   do

      clear
      echo "================================================================="
      echo "| Oracle All Inclusive Tool |"
      echo "| Main Menu - Select Desired Operation(s): |"
      echo "| <CTRL-Z Anytime to Enter Interactive CMD Prompt> |"
      echo "================================================================="
      echo " $IS_SELECTEDM M) View Manual"
      echo " "
      echo " $IS_SELECTED1 1) Drop Tables"
      echo " $IS_SELECTED2 2) Create Tables"
      echo " $IS_SELECTED3 3) Populate Tables"
      echo " $IS_SELECTED4 4) View Tables"
      echo " "
      echo " $IS_SELECTEDX X) Force/Stop/Kill Oracle DB"
      echo " "
      echo " $IS_SELECTEDE E) End/Exit"
      echo "Choose: "

      read CHOICE
 if [ "$CHOICE" = "0" ]
      then
         echo "Nothing Here"
      elif [ "$CHOICE" = "1" ]
     bash drop.sh
         Pause
      elif [ "$CHOICE" = "2" ]
      then
         bash create.sh
         Pause
      elif [ "$CHOICE" = "3" ]
      then
         bash insert.sh
         Pause
      elif [ "$CHOICE" = "4" ]
```

```
      then
         bash view.sh
         Pause
      elif [ "$CHOICE" = "E" ]
      then
         exit
       fi
   done
#--COMMENTS BLOCK--
# Main Program
#--COMMENTS BLOCK--
ProgramStart()
{
   StartMessage
while [ 1 ]
   do
      MainMenu
   done
}
ProgramStart
```

```
 ================================================================
 | Oracle All Inclusive Tool |
 | Main Menu - Select Desired Operation(s): |
 | <CTRL-Z Anytime to Enter Interactive CMD Prompt> |
 ================================================================
   M) View Manual

   1) Drop Tables
   2) Create Tables
   3) Populate Tables
   4) View Tables

   X) Force/Stop/Kill Oracle DB

   E) End/Exit
 Choose:
```

The above figure gives the output for the above menu shell code.

DROP TABLE

```
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib
sqlplus64 "Username/Password(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs$
DROP TABLE STUDENT
DROP TABLE PATIENT_CHECK1
DROP TABLE PATIENT_CHECK2
exit;
EOF
```

CREATE TABLE

```
!/bin/sh
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib
sqlplus64 "Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs$
CREATE TABLE Nurse_Name_Table (ID varchar(255) REFERENCES Employee_Type_Table(Emp_ID), E_Name varchar(255))",
                "CREATE TABLE Doctor_Specialty (D_ID varchar(255) REFERENCES Employee_Type_Table(Emp_ID),
Ext_Number varchar(255), Specialty varchar(255))",
                "CREATE TABLE Employee_Login_Details (Emp_ID varchar(255) REFERENCES Employee_Type_Table(Emp_ID),
Username varchar(255),Pword varchar(255))",*/
                "CREATE TABLE Employee_Personal_Details (Emp_ID varchar(255) REFERENCES
Employee_Type_Table(Emp_ID),Gender varchar(255),Address varchar(255))
```

INSERT TABLE

```
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib
sqlplus64 "Username/Password(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs$
INSERT INTO CLINIC_INFO VALUES(1,'West Clinic','1062 West Street, Toronto','7528825732')"
INSERT INTO CLINIC_INFO VALUES(2,'East Clinic','888 East Street, Waterloo','8471920594')"
INSERT INTO MEDICINE_TABLE VALUES(86, 'Amoxicillin')
INSERT INTO MEDICINE_TABLE VALUES(222, 'Cefaclor')
INSERT INTO Employee_Type_Table VALUES(1, 'Nurse')
INSERT INTO Employee_Login_Details VALUES(1, 'test', 'test')
exit;
EOF
```

VIEW TABLE

```
!/bin/sh
#export LD_LIBRARY_PATH=/usr/lib/oracle/12.1/client64/lib
sqlplus64 "Username/Password@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(Host=oracle.scs$
SELECT * FROM CLINIC_INFO",
SELECT * FROM MEDICAL_RECORD",
CREATE VIEW male_employees AS SELECT EMP_ID, EMP_NAME, GENDER FROM EMPLOYEE_PERSONAL_DETAILS
WHERE GENDER = 'M'",
CREATE VIEW medicine_with_id_over_100 AS SELECT MED_ID, MED_NAME FROM MEDICINE_TABLE WHERE MED_ID >
100"
exit;
EOF
```

JAVA USER INTERFACE (BONUS)

This assignment is based on the Menu Unix Shell using Java template code. The code from the D2L posted from the A9 JdbcConnectionDBMStemplate has been executed using Netbeans and connecting it to the VPN to the Ryerson.

The Java Code which is being attached below has been modified according to the SQL commands and it has been designed according to the menu. The tables have been created, dropped, viewed and executed.

Below is the following code for the Java UI:

```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package assignment.pkg9;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;
import java.util.ArrayList;
import java.util.Scanner;


/**
 *
 * @author sreejachowdary
 */
public class jdbcconnection {

    public static void main(String[] args) {

        Connection conn1 = null;

        try {
            // registers Oracle JDBC driver - though this is no longer required
            // since JDBC 4.0, but added here for backward compatibility
            Class.forName("oracle.jdbc.OracleDriver");


         //   String dbURL1 = "jdbc:oracle:thin:username/password@oracle.scs.ryerson.ca:1521:orcl";  // that is school Oracle database and you can only use it

         String username = "nkurra";
         String password = "05250707";

            String dbURL1 = "jdbc:oracle:thin:" + username + "/" + password + "@oracle.cs.ryerson.ca:1521:orcl";


                        conn1 = DriverManager.getConnection(dbURL1);
            if (conn1 != null) {
                System.out.println("Connected with connection #1");
            }

                        String query = "select NAME, NUM from TESTJDBC";

                        try (Statement stmt = conn1.createStatement()) {
                            Scanner in = new Scanner(System.in);
                            char inp;
                            boolean stay = true;
                            System.out.println("1 - Create Tables\n2 - PopulateTables\n3 - Query Tables\n4 - Drop Tables\ne - Exit");
                            while (stay) {
                                System.out.print("=> ");
                                inp = in.next().charAt(0);
                                switch (inp) {
                                    case '1':
                                        createTables(stmt);
                                        break;
                                    case '2':
                                        populateTables(stmt);
                                        break;
                                    case '3':
                                        queryTables(stmt);
                                        break;
                                    case '4':
                                        dropTables(stmt);
                                        break;
                                    case 'e':
                                        stay = false;
                                        System.out.println("Exiting Program");
                                        break;
                                    default:
                                        System.out.println("Invalid Input");

                                }
```

```
                    } catch (SQLException e) {
                            System.out.println("Error " + e.getErrorCode());
                            //e.printStackTrace();
                    }



        } catch (ClassNotFoundException | SQLException ex) {
        } finally {
            try {
                if (conn1 != null && !conn1.isClosed()) {
                    conn1.close();
                }

            } catch (SQLException ex) {
            }
        }


    }

    public static void createTables(Statement stmt) throws SQLException {
        String[] queries = {/*"CREATE TABLE Nurse_Name_Table (ID varchar(255) REFERENCES Employee_Type_Table(Emp_ID), E_Name varchar(255))",
                            "CREATE TABLE Doctor_Specialty (D_ID varchar(255) REFERENCES Employee_Type_Table(Emp_ID), Ext_Number varchar(255), Specialty varchar(255))",
                            "CREATE TABLE Employee_Login_Details (Emp_ID varchar(255) REFERENCES Employee_Type_Table(Emp_ID), Username varchar(255),Pword varchar(255))",
                            "CREATE TABLE Employee_Personal_Details (Emp_ID varchar(255) REFERENCES Employee_Type_Table(Emp_ID),Gender varchar(255),Address varchar(255))
        for (String query : queries) {
            stmt.executeUpdate(query);
        }
        System.out.println("Tables Created");
    }

    public static void populateTables(Statement stmt) throws SQLException {
        String[] queries = {//"INSERT INTO CLINIC_INFO VALUES(1,'West Clinic','1062 West Street, Toronto','7528825732')",
                            //"INSERT INTO CLINIC_INFO VALUES(2,'East Clinic','888 East Street, Waterloo','8471920594')",
                            "INSERT INTO MEDICINE_TABLE VALUES(86, 'Amoxicillin')",
                            "INSERT INTO MEDICINE_TABLE VALUES(222, 'Cefaclor')",
                            //"INSERT INTO Employee_Type_Table VALUES(1, 'Nurse')",
                            //"INSERT INTO Employee_Login_Details VALUES(1, 'test', 'test')"
                            };
        for (String query : queries) {
            stmt.executeUpdate(query);
        }
        System.out.println("Tables Populated");
    }

    public static void queryTables(Statement stmt) throws SQLException {
        String[] queries = {"SELECT * FROM CLINIC_INFO",
                            "SELECT * FROM MEDICAL_RECORD",
                            /*"CREATE VIEW male_employees AS SELECT EMP_ID, EMP_NAME, GENDER FROM EMPLOYEE_PERSONAL_DETAILS WHERE GENDER = 'M'",
                            "CREATE VIEW medicine_with_id_over_100 AS SELECT MED_ID, MED_NAME FROM MEDICINE_TABLE WHERE MED_ID > 100",*/};
        for (String query : queries) {
            System.out.println(query);
            ResultSet rs = stmt.executeQuery(query);
            ResultSetMetaData rsmd = rs.getMetaData();
            ArrayList<String> columns = new ArrayList<>();
            for(int i = 1; i <= rsmd.getColumnCount(); i++) {
                columns.add(rsmd.getColumnName(i));
                System.out.print(rsmd.getColumnName(i));
                if (i != rsmd.getColumnCount())
                    System.out.print(", ");
            }
            System.out.println();
            while (rs.next()) {
                for (String c : columns) {
                    System.out.print(rs.getString(c));
                    if (!columns.get(columns.size() - 1).equals(c))
                        System.out.print(", ");
                }
                System.out.println();
            }
```

```
                    }
                    System.out.println();
                }
                System.out.println();
            }
            System.out.println("Tables Queried");
        }

        public static void dropTables(Statement stmt) throws SQLException {
            String[] queries = {
                                "DROP TABLE STUDENT"
                                /*"DROP TABLE PATIENT_CHECK1",
                                "DROP TABLE PATIENT_CHECK2"*/};
            for (String query : queries) {
                stmt.executeUpdate(query);
            }
            System.out.println("Tables Dropped");
        }

    }
```

The above is the code for the Java GUI for the menu created.

The following output will be based on the tables created, queried, dropped and inserted.

```
run:
Connected with connection #1
1 - Create Tables
2 - PopulateTables
3 - Query Tables
4 - Drop Tables
e - Exit
=> 1
Tables Created
```

```
=> 3
SELECT * FROM CLINIC_INFO
CLINIC_ID, CLINIC_NAME, CLINIC_ADDRESS, CLINIC_PHONE_NUMBER
1, West Clinic, 1062 West Street, Toronto, 7528825732
2, East Clinic, 888 East Street, Waterloo, 8471920594

SELECT * FROM MEDICAL_RECORD
MEDICAL_RECORD_ID, APPOINTMENT, P_ID, PATIENT_DETAILS
1, 20210720, 3, Fever
2, 20210422, 2, Stomach Pain
3, 20210104, 1, Cold,Cough

Tables Queried
```

```
Connected with connection #1
1 - Create Tables
2 - PopulateTables
3 - Query Tables
4 - Drop Tables
e - Exit
=> 4
Tables Dropped
=>

Connected with connection #1
1 - Create Tables
2 - PopulateTables
3 - Query Tables
4 - Drop Tables
e - Exit
=> 2
Tables Populated
```

These are the views created for the assignment 9 for the GUI. The results have been shown during the demo.