Relational Databases with MySQL Week 10 Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document to the repository. Additionally, push an .sql file with all your queries and your Java project code to the same repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

Coding Steps:

In this week's coding activity, you will create a menu driven application backed by a MySQL database.

To start, choose one item that you like. It could be vehicles, sports, foods, etc....

Create a new Java project in Eclipse.

Create a SQL script in the project to create a database with one table. The table should be the item you picked.

Write a Java menu driven application that allows you to perform all four CRUD operations on your table.

Tips:

The application does not need to be as complex as the example in the video curriculum.

You need an option for each of the CRUD operations (Create, Read, Update, and Delete).

Remember that PreparedStatment.executeQuery() is only for Reading data and .executeUpdate() is used for Creating, Updating, and Deleting data.

Remember that both parameters on PreparedStatements and the ResultSet columns are based on indexes that start with 1, not 0.

Screenshots of Code:

Application Class:

```
1  package application;
2
3  public class Application {
4
5    public static void main(String[] args) {
6      Menu menu = new Menu();
7      menu.start();
8    }
9
10 }
11
```

Menu Class:

```java
package application;

import java.sql.SQLException;
import java.util.Arrays;
import java.util.List;
import java.util.Scanner;
import dao.QueensDao;
import entity.Queens;

public class Menu {

  private Scanner scanner = new Scanner(System.in);
  private List<String> options = Arrays.asList (
      "Add Drag Queen",
      "Display Drag Queens",
      "Update Drag Queen",
      "Delete Drag Queen",
      "Exit"
      );

  public void start() {
    String selection = "";

    do {
      printMenu();
      selection = scanner.nextLine();

      try {
        if (selection.equals("1")) {
          createDragQueen();
        } else if (selection.equals("2") ) {
          displayDragQueens();
        } else if (selection.equals("3") ) {
          updateDragQueen();
        } else if (selection.equals("4") ) {
          deleteDragQueen();
        }
      } catch (SQLException e) {
        e.printStackTrace();
      }

  } while (!selection.equals("5"));
  }
```

```
44
45    private void printMenu() {
46      System.out.println("Select an option: \n--------------");
47      for (int i=0; i < options.size(); i++) {
48        System.out.println(i + 1 + ")" + options.get(i) );;
49      }
50    }
51
52    private void createDragQueen() throws SQLException {
53      System.out.print("Add First Name: ");
54      String firstName = scanner.nextLine();
55      System.out.print("Add Last Name: ");
56      String lastName = scanner.nextLine();
57      QueensDao queensDao = new QueensDao();
58      queensDao.createNewQueen(firstName, lastName);
59    }
60
61    private void displayDragQueens() throws SQLException {
62      List<Queens> queens = QueensDao.getAll();
63      for (Queens queen : queens) {
64        System.out.println(queen.getQueenId() + ": " + queen.getFirstName() + " " + queen.getLastName());
65      }
66    }
67
68    private void updateDragQueen() throws SQLException {
69      System.out.print("Add First Name: ");
70      String firstName = scanner.nextLine();
71      System.out.print("Add Last Name: ");
72      String lastName = scanner.nextLine();
73      System.out.print("Enter Drag Queen ID to Update:");
74      int id = Integer.parseInt(scanner.nextLine());
75      QueensDao queensDao = new QueensDao();
76      queensDao.updateQueenByID(firstName, lastName, id);
77    }
78
79    private void deleteDragQueen() throws SQLException {
80      System.out.print("Enter Queen ID to delete: ");
81      int id = Integer.parseInt(scanner.nextLine());
82      QueensDao queensDao = new QueensDao();
83      queensDao.deleteQueenById(id);
84    }
85  }
86
```

DBConnection Class:

```java
package dao;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DBConnection {

  private final static String URL = "jdbc:mysql://localhost:3306/queens";
  private final static String USERNAME = "root";
  private final static String PASSWORD = "password";
  private static Connection connection;
  private static DBConnection instance;

  private DBConnection(Connection connection) {
    DBConnection.connection = connection;
  }


  public static Connection getConnection() {
    if(instance == null) {
      try {
        connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);
        instance = new DBConnection(connection);
        System.out.println("Connection Successful.");
      } catch(SQLException e) {
        e.printStackTrace();
      }
    }
    return DBConnection.connection;
  }
}

```

QueensDao Class:

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import entity.Queens;

public class QueensDao {
  private static Connection connection;
  private final String CREATE_NEW_QUEEN = "INSERT INTO queens(first_name, last_name) VALUES(?,?)";
  private final static String GET_QUEENS_QUERY = "SELECT * FROM queens";
  private final String UPDATE_QUEEN_BY_ID = "UPDATE queens SET first_name=?, last_name=? WHERE id=?";
  private final String DELETE_QUEEN_BY_ID = "DELETE FROM queens WHERE id = ?";

  public QueensDao() {
    connection = DBConnection.getConnection();
  }

  public void createNewQueen(String firstName, String lastName) throws SQLException {
    PreparedStatement ps = connection.prepareStatement(CREATE_NEW_QUEEN);
    ps.setString(1, firstName);
    ps.setString(2, lastName);
    ps.executeUpdate();
  }

   public static List<Queens> getAll() throws SQLException {
     ResultSet rs = connection.prepareStatement(GET_QUEENS_QUERY).executeQuery();
     List<Queens> all = new ArrayList<Queens>();

     while(rs.next()) {
       all.add(new Queens(rs.getInt(1), rs.getString(2), rs.getString (3)));
     }
     return all;
   }
```

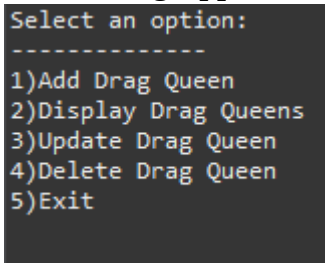
```java


  public void updateQueenByID(String firstName, String lastName, int id) throws SQLException {
    PreparedStatement ps = connection.prepareStatement(UPDATE_QUEEN_BY_ID);
    ps.setString(1, firstName);
    ps.setString(2, lastName);
    ps.setInt(3, id);
    ps.executeUpdate();
  }

  public void deleteQueenById(int id) throws SQLException {
    PreparedStatement ps = connection.prepareStatement(DELETE_QUEEN_BY_ID);
    ps.setInt(1, id);
    ps.executeUpdate();
  }


  }

```

Queens Class:

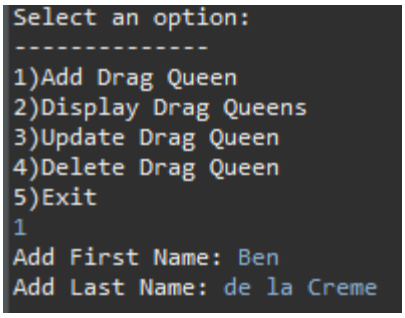
```java
package entity;

public class Queens {
  private int queenId;
  private String firstName;
  private String lastName;

  public Queens(int queenId, String firstName, String lastName) {
    this.setQueenId(queenId);
    this.setFirstName(firstName);
    this.setLastName(lastName);
  }

  public int getQueenId() {
    return queenId;
  }

  public void setQueenId(int queenId) {
    this.queenId = queenId;
  }

  public String getFirstName() {
    return firstName;
  }

  public void setFirstName(String firstName) {
    this.firstName = firstName;
  }

  public String getLastName() {
    return lastName;
  }

  public void setLastName(String lastName) {
    this.lastName = lastName;
  }

}

```

Screenshots of Running Application:

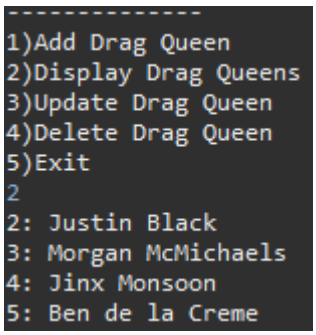
Start Menu:



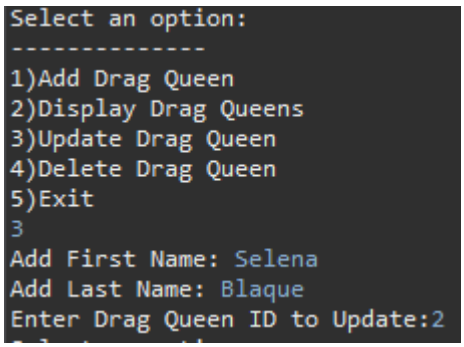
1 – Add Drag Queen

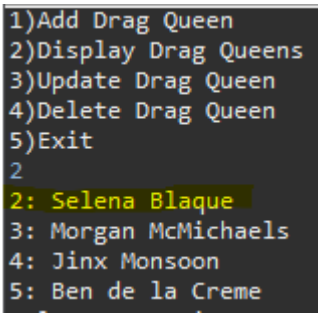


2 – Display Drag Queens

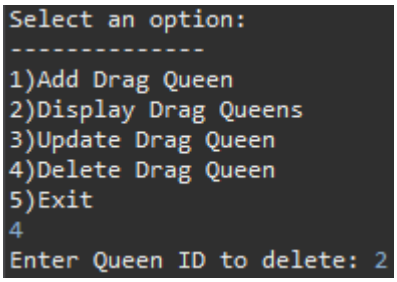


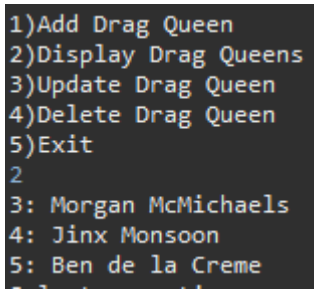
3 – Update Drag Queen





4 – Delete Drag Queen





URL to GitHub Repository:

https://github.com/KT-Trailblazer/WK10CA