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.

```
create database if not exists colorsNew;
use colorsNew;
drop table if exists colorsNew;

create table colorsNew (
hexId varchar(7) not null,
colorName varchar(50) not null,
primary key (hexId)
);
```

Screenshots of Code:

Application.jav | Menu.java | ColorsNewDao.ja

```
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
```


Application.jav
Menu.java
ColorsNewDao.ja
DBConnection.ja
ColorNew.java
ckage application;
port java.sql.SQLException;
This displays a menu to allow the user to perform CRUD operations on a colorsNew table.
blic class Menu {
private ColorsNewDao colorsNewDao = new ColorsNewDao();
private Scanner scanner = new Scanner(System.in);
private List<String> options = Arrays.asList(
"1) Display Colors", // read
"2) Add Color", // create
"3) Edit Color", // update
"4) Delete Color", // delete
"5) Type 5 to Exit");
public void start() {
String selection = "";
do {
printMenu();
selection = scanner.nextLine();
try {
if (selection.equals("1")) {
listColors();
} else if (selection.equals("2")) {
addColor();
} else if (selection.equals("3")) {
editColor();
} else if (selection.equals("4")) {
deleteColor();
}
} catch (SQLException e) {

Application.jav | Menu.java | ColorsNewDao.ja | DBConnection.ja | ColorNew.java

```
43
44              System.out.println("Press enter to continue..."); // keeps menu from automatica
45                                                               // to start the menu again
46              scanner.nextLine();
47
48          } while (!selection.equals("5"));
49      }
50
51      private void editColor() throws SQLException {
52          listColors();
53          System.out.println("Enter hexId to edit: ");
54          String hexId = scanner.nextLine();
55          System.out.println("Enter new color name: ");
56          String colorName = scanner.nextLine();
57          colorsNewDao.updateColorByHex(hexId, colorName);
58      }
59
60      private void deleteColor() throws SQLException {
61          System.out.print("Enter hexId to delete: ");
62          String hexId = scanner.nextLine();
63          colorsNewDao.deleteColorByHex(hexId);
64      }
65
66      private void addColor() throws SQLException {
67          System.out.print("Enter new hexID: ");
68          String hexIdId = scanner.nextLine();
69          System.out.println("Enter new Color Name");
70          String colorNameNew = scanner.nextLine();
71          colorsNewDao.addColor(hexIdId, colorNameNew);
72      }
73
74      private void listColors() throws SQLException {
75          List<ColorNew> colors = colorsNewDao.listColors();
76          System.out.println("All colors: ");
77          //System.out.println(colors.size());
```

```
    private void listColors() throws SQLException {
        List<ColorNew> colors = colorsNewDao.listColors();
        System.out.println("All colors: ");
        //System.out.println(colors.size());
        for (ColorNew color : colors) {
            System.out.println(" " + color.getHexId() + " " + color.getName());
        }
    }

    private void printMenu() {
        System.out.println("Select an Option:\n-------------------------");
        for (String line : options) {
            System.out.println(line);

    }
```

Application.jav | Menu.java | ColorsNewDao.ja | DBConnection.ja | ColorNew.java

```java
package dao;

import java.sql.Connection;


public class ColorsNewDao {

    //private Connection connection;
    //private final String GET_COLORS_QUERY = "SELECT * FROM colorsNew";
    //private final String ADD_COLOR_QUERY = "INSERT INTO colorsNew (hexId, colorName) VA
    //private final String DELETE_COLOR_BY_HEX_QUERY = "DELETE FROM colorsNew WHERE hexI
    //private final String EDIT_COLOR_BY_HEX_QUERY = "UPDATE colorsNew SET colorName = ?

    //public ColorsNewDao() {
        //connection = DBConnection.getConnection();
    //}

    //List all colors currently in the table
    public List<ColorNew> listColors() throws SQLException {
        String sql = "SELECT * FROM colorsNew";
        Connection conn = DBConnection.instance().getConnection();
        try (PreparedStatement stmt = conn.prepareStatement(sql)) {
            List<ColorNew> colorsNew = new LinkedList<>();
            try (ResultSet rs = stmt.executeQuery()) {
                while (rs.next()) {
                    ColorNew colorNew = new ColorNew(rs.getString(1), rs.getString(2));
                    colorsNew.add(colorNew);
                }
            }
            return colorsNew;
        } catch (SQLException e) {
            throw new IllegalStateException(e.getMessage(), e);
        }
        //ResultSet rs = connection.prepareStatement(GET_COLORS_QUERY).executeQuery(); //
```

Application.jav | Menu.java | ColorsNewDao.ja | DBConnection.ja | ColorNew.java

```
43                                                                                  /
44          //List<ColorNew> colorNew = new ArrayList<ColorNew>();
45
46          //while (rs.next()) {
47          //  colorNew.add(colorList(rs.getString(1), rs.getString(2)));
48      //  }
49      //  return colorNew;
50      }
51
52      //add a color to the table with a hexId and color name
53      public void addColor(String hexId, String colorName) throws SQLException {
54          String sql = "INSERT INTO colorsNew (hexId, colorName) VALUES (?,?)";
55          Connection conn = DBConnection.instance().getConnection();
56
57          try (PreparedStatement stmt = conn.prepareStatement(sql)) {
58              stmt.setString(1, hexId);
59              stmt.setString(2, colorName);
60              stmt.executeUpdate();
61          } catch (SQLException e) {
62          }
63
64      }
65  // delete a color in the table - must look up using hexId
66      public void deleteColorByHex(String hexId) {
67          String sql = "DELETE FROM colorsNew WHERE hexId = ?";
68          Connection conn = DBConnection.instance().getConnection();
69          try (PreparedStatement ps = conn.prepareStatement(sql)) {
70          ps.setString(1, hexId);
71          if (ps.executeUpdate() == 0) {
72              throw new IllegalStateException("Color with hexId " + hexId + " does not exis
73          }
74      } catch (SQLException e) {
75          throw new IllegalStateException(e.getMessage(), e);
76      }
```

```
    }
//edit a color in the table by changing the color name, looking it up by hexid
    public void updateColorByHex(String colorName, String hexId) throws SQLException
        String sql = "UPDATE colorsNew SET colorName = ? WHERE hexId = ?";
        Connection conn = DBConnection.instance().getConnection();

        try (PreparedStatement ps = conn.prepareStatement(sql)) {
        ps.setString(1, colorName);
        ps.setString(2, hexId);
        ps.executeUpdate();

        }
    }

    //private ColorNew colorList(String hexId, String colorName) {
        //return new ColorNew(hexId, colorName);
    //}

}
```

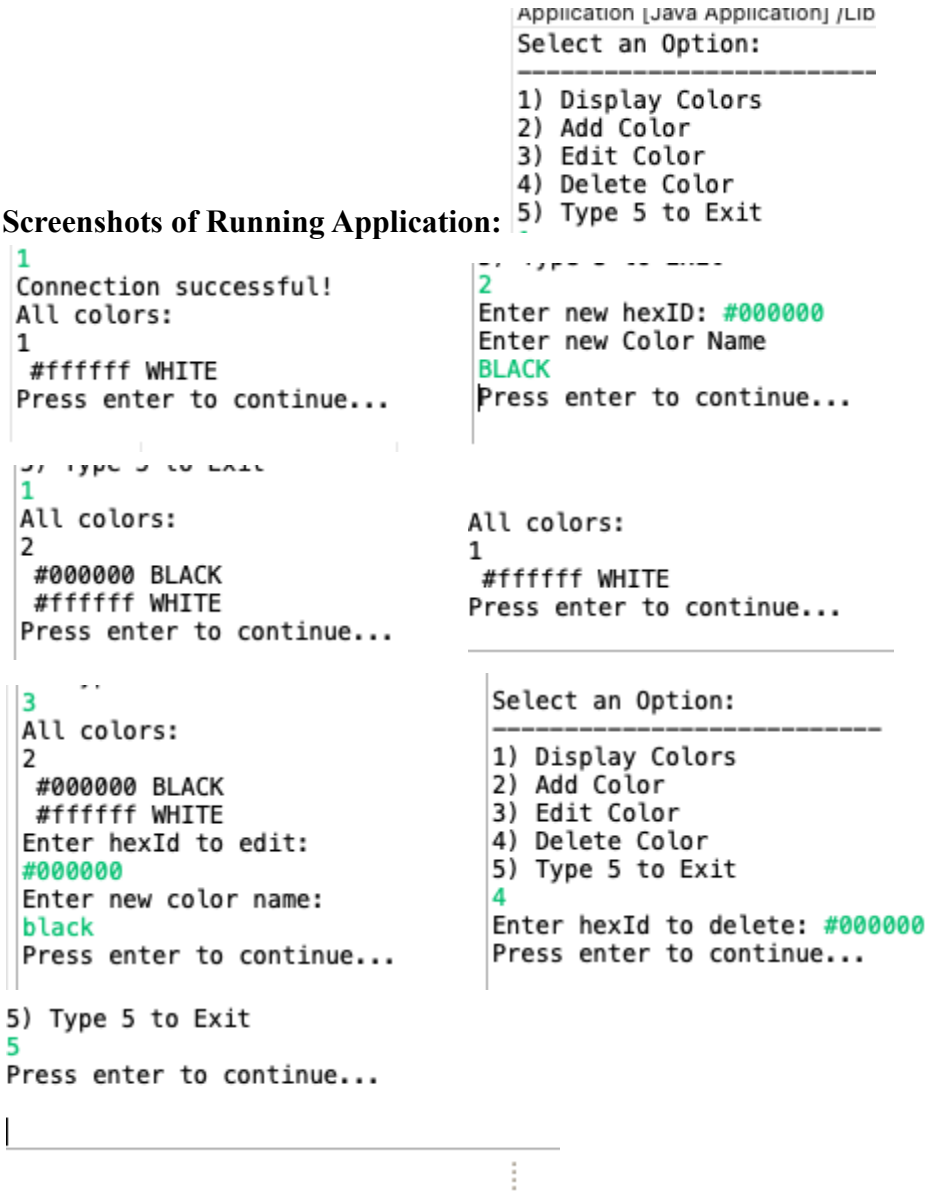
Application.jav | Menu.java | ColorsNewDao.ja | DBConnection.ja | ColorNew.java

```java
package dao;

import java.sql.Connection;
//This demonstrates the Singleton Design Pattern. A single DbConnection object is created

public final class DBConnection {

    private final static String URL = "jdbc:mysql://localhost:3306/colorsNew";
    private final static String USERNAME = "root";
    private final static String PASSWORD = "Codingclass2";
    private Connection connection;
    private final static DBConnection instance = new DBConnection();

    private DBConnection() {
    }
    public static DBConnection instance() {
        return instance;
    }

    public Connection getConnection() {
        //The connection is only created if it does not exist. Otherwise it is reused.
        if (connection == null) {
            try {
                connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);
                System.out.println("Connection successful!");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        return connection;
    }

}
```

Application.jav | Menu.java | ColorsNewDao.ja | DBConnection.ja | ColorNew.java

```java
package entity;

public class ColorNew {
    private String hexId;
    private String colorName;

    public ColorNew(String hexId, String colorName) {
        this.hexId = hexId;
        this.colorName = colorName;
    }
//return hexId
    public String getHexId() {
        return hexId;
    }

    public void setHexId(String hexId) {
        this.hexId = hexId;
    }
//return color name
    public String getName() {
        return colorName;
    }

    public void setName(String colorName) {
        this.colorName = colorName;
    }

}
```

Screenshots of Running Application:



URL to GitHub Repository: https://github.com/cperrine19/Week10MySQL

****NOTE** There is something currently wrong with my ability to push to github. I am working on fixing it**