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).

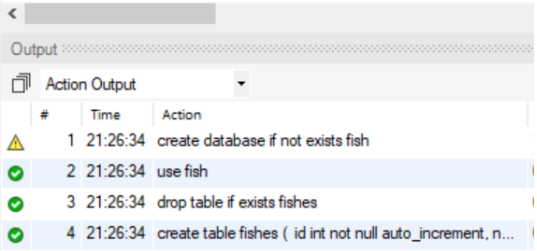
Remember that PreparedStatment.executeQuery() is only for Reading data and .executeUpdate() is used for Creating, Updating, and Deleting data.

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

Screenshots of Code:

Creating the Database:

```
1  create database if not exists fish;
2
3  use fish;
4
5  drop table if exists fishes;
6
7  create table fishes (
8      id int not null auto_increment,
9      name varchar(50) not null,
10     primary key(id)
11     );
12
```



Application to start the Menu

```
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 }
```

Class assigning variables to the data from SQL

```java
package entity;

public class Fishes {

    private int fishId;
    private String name;

    public Fishes(int fishId, String name) {
        this.setFishId(fishId);
        this.setName(name);
    }

    public int getFishId() {
        return fishId;
    }

    public void setFishId(int fishId) {
        this.fishId = fishId;
    }

    public String getName() {
        return name;
    }

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

Establishing a connection to the database

```java
package dao;

import java.sql.Connection;

public class DBConnection {


    private final static String URL = "jdbc:mysql://localhost:3306/fish";
    private final static String USERNAME = "root";
    private final static String PASSWORD = "Colton@11";

    private static Connection connection;
    private static DBConnection instance;

    private DBConnection(Connection connection) {
        this.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"
                        + "\n");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        return DBConnection.connection;
    }
```

Java Data Access Objects Class (SQL Queries)

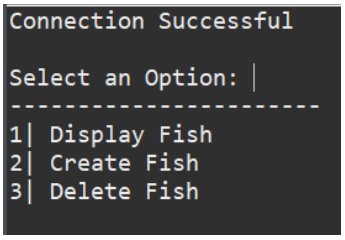
```java
package dao;

import java.sql.Connection;

public class FishesDAO {

    private Connection connection;

    private final String GET_FISHES_QUERY = "SELECT * FROM fishes";

    private final String CREATE_NEW_FISH_QUERY = "INSERT INTO fishes(name) VALUES (?)";

    private final String DELETE_FISH_BY_ID_QUERY = "DELETE FROM fishes WHERE id = ?";


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

    public List<Fishes> getFish() throws SQLException {
        ResultSet rs = connection.prepareStatement(GET_FISHES_QUERY).executeQuery();
        List<Fishes> fish = new ArrayList<Fishes>();

        while (rs.next()) {
            fish.add(populateFishes(rs.getInt(1), rs.getString(2)));
        }
    return fish;
    }

    private Fishes populateFishes(int id, String name) {
        return new Fishes(id, name);
    }

    public void createFish(String fishName) throws SQLException {
    PreparedStatement ps = connection.prepareStatement(CREATE_NEW_FISH_QUERY);
    ps.setString(1, fishName);
    ps.executeUpdate();
    }

    public void deleteFish(int id) throws SQLException {
        PreparedStatement ps = connection.prepareStatement(DELETE_FISH_BY_ID_QUERY);
        ps.setInt(1, id);
        ps.executeUpdate();
```

The Menu

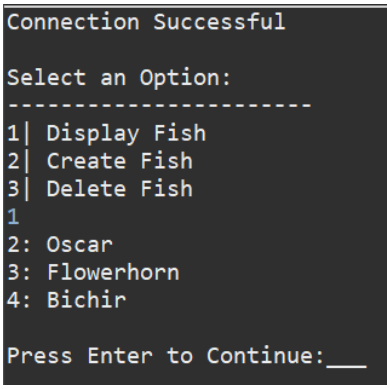
```java
package application;

import java.sql.SQLException;

public class Menu {

    private Scanner sc = new Scanner(System.in);

    private FishesDAO FishesDAO = new FishesDAO();

    private List<String> menuOptions = Arrays.asList("Display Fish", "Create Fish", "Delete Fish");


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

        do {
            printMenu();
            selection = sc.nextLine();
            try {
                if (selection.equals("1")) {
                    displayFish();
                } else if (selection.equals("2")) {
                    createFish();
                } else if (selection.equals("3")) {
                    deleteFish();
                }

            } catch (SQLException e) {
                e.printStackTrace();
            }
            System.out.println("\nPress Enter to Continue:___");
            sc.nextLine();
        } while (!selection.equals("-1"));
    }


    private void printMenu() {
        System.out.println("Select an Option: \n----------------------");
        for (int i = 0; i < menuOptions.size(); i++) {
            System.out.println(i + 1 + "| " + menuOptions.get(i));
        }
    }

    private void displayFish() throws SQLException {
        List<Fishes> fish = FishesDAO.getFish();
        for (Fishes fishes : fish) {
            System.out.println(fishes.getFishId() + ": " + fishes.getName());
        }
    }

    private void createFish() throws SQLException {
        System.out.println("Enter a new Fish:");
        String fishName = sc.nextLine();
        FishesDAO.createFish(fishName);
    }

    private void deleteFish() throws SQLException {
    System.out.println("Enter the Fish ID you wish to delete:");
    int id = Integer.parseInt(sc.nextLine());
    FishesDAO.deleteFish(id);
    }

```

Screenshots of Running Application:

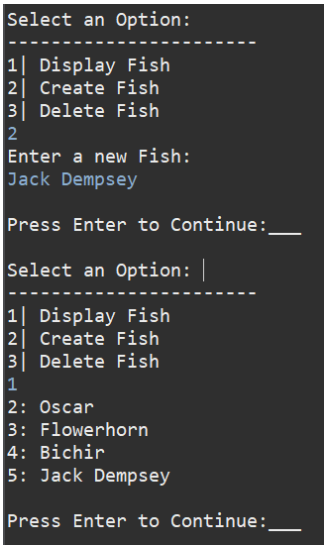
Connecting to the database:



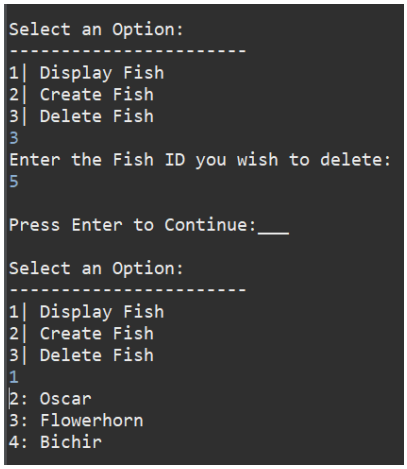
Displaying entries in the table:



Creating new Entries in the table:



Deleting entries in the table:



URL to GitHub Repository:

https://github.com/coltonrood/Week10