

Relational Databases with MySQL Week 5 Coding 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, with your Java project code, to the 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 `PreparedStatement.executeQuery()` is only for Reading data and `.executeUpdate()` is used for Creating, Updating, and Deleting data.

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

Screenshots of Code:

```
1 package application;
2
3 import java.sql.SQLException;
4 import java.util.Arrays;
5 import java.util.List;
6 import java.util.Scanner;
7
8 import dao.VehicleDao;
9 import entity.Vehicle;
10
11 public class Menu {
12
13
14     private VehicleDao vehicleDao = new VehicleDao();
15     private Scanner scanner = new Scanner(System.in);
16     private List<String> options = Arrays.asList(
17         "Add Vehicle",
18         "Display Vehicles",
19         "Update Vehicle",
20         "Delete Vehicle"
21     );
22
23     public void start() {
24         String selection = "";
25
26         do {
27             printMenu();
28             selection = scanner.nextLine();
29
30             try {
31                 if (selection.equals("1")) {
32                     createVehicle();
33                 } else if (selection.equals("2")) {
34                     displayVehicles();
35                 } else if (selection.equals("3")) {
36                     updateVehicle();
37                 } else if (selection.equals("4")) {
38                     deleteVehicle();
39                 }
40             } catch (SQLException e) {
41                 e.printStackTrace();
42             }
43
44             System.out.println("Press enter to continue....");
45             scanner.nextLine();
46         } while (!selection.equals("-1"));
47     }
48
49     private void printMenu() {
50         System.out.println("Select an Option:-");
51         for (int i = 0; i < options.size(); i++) {
52             System.out.println(i + 1 + " -> " + options.get(i));
53         }
54     }
55
56     private void displayVehicles() throws SQLException {
57         List<Vehicle> vehicles = vehicleDao.getVehicles();
58         for (Vehicle vehicle : vehicles) {
59             System.out.println(vehicle.getVehicleId() + " : " + vehicle.getMake());
60         }
61     }
62
63     private void updateVehicle() throws SQLException {
64         System.out.println("Enter vehicle id: ");
65         int id = Integer.parseInt(scanner.nextLine());
66         System.out.println("Enter new vehicle make: ");
67         String make = scanner.nextLine();
68         vehicleDao.updateVehicleById(id, make);
69     }
70
71     private void createVehicle() throws SQLException {
72         System.out.println("Enter new vehicle make:");
73         String vehicleMake = scanner.nextLine();
74         vehicleDao.createNewVehicle(vehicleMake);
75     }
76
77     private void deleteVehicle() throws SQLException {
78         System.out.println("Enter vehicle id to delete:");
79         int id = Integer.parseInt(scanner.nextLine());
80         vehicleDao.deleteVehicleById(id);
81     }
82 }
83
```

```
1 package entity;
2
3 public class Vehicle {
4
5     private int vehicleId;
6     private String make;
7
8
9     public Vehicle(int vehicleId, String make) {
10         this.setVehicleId(vehicleId);
11         this.setMake(make);
12     }
13
14     public int getVehicleId() {
15         return vehicleId;
16     }
17
18     public void setVehicleId(int vehicleId) {
19         this.vehicleId = vehicleId;
20     }
21
22     public String getMake() {
23         return make;
24     }
25
26     public void setMake(String make) {
27         this.make = make;
28     }
29 }
30
31
```

```
1 package dao;
2
3 import java.sql.Connection;
4 import java.sql.PreparedStatement;
5 import java.sql.ResultSet;
6 import java.sql.SQLException;
7 import java.util.ArrayList;
8 import java.util.List;
9
10 import entity.Vehicle;
11
12 public class VehicleDao {
13
14     private Connection connection;
15     //private VehicleDao vehicleDao;
16     private final String GET_VEHICLES_QUERY = "SELECT * FROM vehicles";
17     private final String UPDATE_VEHICLE_BY_ID_QUERY = "UPDATE vehicles SET make = ? WHERE id = ?";
18     private final String CREATE_NEW_VEHICLE_QUERY = "INSERT INTO vehicles (make) VALUES (?)";
19     private final String DELETE_VEHICLE_BY_ID_QUERY = "DELETE FROM vehicles WHERE id = ?";
20
21     public VehicleDao() {
22         connection = DBConnection.getConnection();
23     }
24
25     public List<Vehicle> getVehicles() throws SQLException {
26         PreparedStatement ps = connection.prepareStatement(GET_VEHICLES_QUERY);
27         ResultSet rs = ps.executeQuery();
28         List<Vehicle> vehicles = new ArrayList<Vehicle>();
29
30         while (rs.next()) {
31             Vehicle vehicle = new Vehicle(rs.getInt(1), rs.getString(2));
32             vehicles.add(vehicle);
33         }
34         return vehicles;
35     }
36
37     public void updateVehicleById(int id, String make) throws SQLException {
38         PreparedStatement ps = connection.prepareStatement(UPDATE_VEHICLE_BY_ID_QUERY);
39         ps.setInt(1, id);
40         ps.setString(2, make);
41         ps.executeUpdate();
42     }
43
44     public void createNewVehicle(String make) throws SQLException {
45         PreparedStatement ps = connection.prepareStatement(CREATE_NEW_VEHICLE_QUERY);
46         ps.setString(1, make);
47         ps.executeUpdate();
48     }
49
50     public void deleteVehicleById(int id) throws SQLException {
51         PreparedStatement ps = connection.prepareStatement(DELETE_VEHICLE_BY_ID_QUERY);
52         ps.setInt(1, id);
53         ps.executeUpdate();
54     }
55 }
56
```

```
1 package dao;
2
3 import java.sql.Connection;
4 import java.sql.DriverManager;
5 import java.sql.SQLException;
6
7 public class DBConnection {
8
9
10     private final static String URL = "jdbc:mysql://localhost:3306/vehicles";
11     private final static String USERNAME = "root";
12     private final static String PASSWORD = "NonePass";
13     private static Connection connection;
14     private static DBConnection instance;
15
16     private DBConnection(Connection connection) {
17         this.connection = connection;
18     }
19
20     public static Connection getConnection() {
21         if (instance == null) {
22             try {
23                 connection = DriverManager.getConnection(URL, USERNAME, PASSWORD);
24                 instance = new DBConnection(connection);
25                 System.out.println("Connection successful.");
26             } catch (SQLException e) {
27                 e.printStackTrace();
28             }
29         }
30         return DBConnection.connection;
31     }
32 }
33
```

```
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
```

Screenshots of Running Application:

```
Application (2) [Java Application] /Library/Java/J
Connection successful.
Select an Option:
-----
1) Add Vehicle
2) Display Vehicles
3) Update Vehicle
4) Delete Vehicle
2
2: Lexus
3: Audi
Press enter to continue....
```

URL to GitHub Repository:

<https://github.com/dwold/Week11Assignment>