

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:

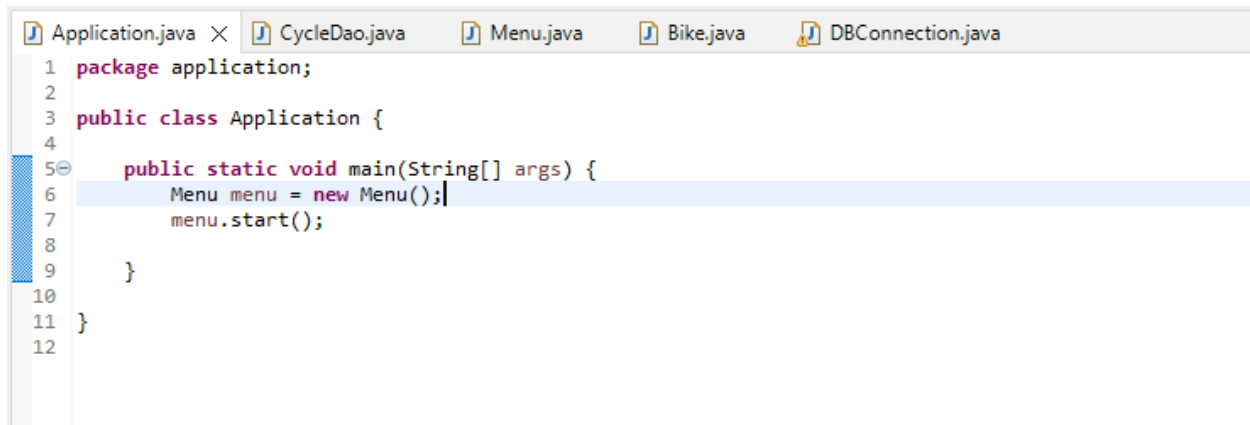
The application does not need to be as complex as the example in the video curriculum.

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

Remember that `PreparedStatement.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:

A screenshot of an IDE showing the code for Application.java. The file is part of a project with other files like CycleDao.java, Menu.java, Bike.java, and DBConnection.java. The code defines a package 'application' and a public class 'Application'. It contains a main method that creates a new Menu object and calls its start method.

```
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 }
12
```

A screenshot of an IDE showing the code for CycleDao.java. The file is part of a project with other files like Application.java, Menu.java, Bike.java, and DBConnection.java. The code defines a package 'dao' and a public class 'CycleDao'. It imports various Java classes including Connection, PreparedStatement, ResultSet, SQLException, ArrayList, and List. It also imports the Bike class from the 'entities' package. The class contains private fields for SQL queries and a constructor that initializes the connection. It also has a getBikes method that uses a PreparedStatement to retrieve data from a database and returns it as a List of Bike objects.

```
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 entities.Bike;
11
12 public class CycleDao {
13     private Connection connection;
14     private final String SHOW_ALL_BIKES_QUERY = "SELECT * FROM bikes";
15     private final String DELETE_BIKES_BY_ID_QUERY = "DELETE FROM bikes WHERE id = ?";
16     private final String CREATE_NEW_BIKE_QUERY = "INSERT INTO bikes(year, make, model, engine_size) VALUES (?, ?, ?, ?)";
17     private final String UPDATE_BIKE_BY_ID_QUERY = "UPDATE bikes SET year = ?, make = ?, model = ?, engine_size = ? WHERE id = ?";
18
19     public CycleDao() {
20         connection = DBConnection.getConnection();
21     }
22
23     public List<Bike> getBikes() throws SQLException {
24         PreparedStatement ps = connection.prepareStatement(SHOW_ALL_BIKES_QUERY);
25         ResultSet rs = ps.executeQuery();
26         List<Bike> bikes = new ArrayList<Bike>();
27
28         while (rs.next()) {
29             bikes.add(populateBike(rs.getInt(1), rs.getInt(2), rs.getString(3), rs.getString(4), rs.getInt(5)));
30         }
31         return bikes;
32     }
33 }
34
```

```

Application.java CycleDao.java X Menu.java Bike.java DBConnection.java
32     return bikes;
33 }
34
35
36 public void createNewBike(int year, String make, String model, int engineSize) throws SQLException {
37     PreparedStatement ps = connection.prepareStatement(CREATE_NEW_BIKE_QUERY);
38     ps.setInt(1, year);
39     ps.setString(2, make);
40     ps.setString(3, model);
41     ps.setInt(4, engineSize);
42     ps.executeUpdate();
43 }
44
45 public void updateBikeById(int id, int year, String make, String model, int engineSize) throws SQLException {
46     PreparedStatement ps = connection.prepareStatement(UPDATE_BIKE_BY_ID_QUERY);
47     ps.setInt(1, year);
48     ps.setString(2, make);
49     ps.setString(3, model);
50     ps.setInt(4, engineSize);
51     ps.setInt(5, id);
52     ps.executeUpdate();
53 }
54
55 public void deleteBikeById(int id) throws SQLException {
56     PreparedStatement ps = connection.prepareStatement(DELETE_BIKES_BY_ID_QUERY);
57     ps.setInt(1, id);
58     ps.executeUpdate();
59 }
60 }
61
62 private Bike populateBike(int id, int year, String make, String model, int engine_size) throws SQLException {
63     return new Bike(id, year, make, model, engine_size);
64 }
65 }
66 }
67

```

```

Application.java CycleDao.java Menu.java X Bike.java DBConnection.java
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.CycleDao;
9 import entities.Bike;
10
11 public class Menu {
12
13
14     private CycleDao cycleDao = new CycleDao();
15     private Scanner scanner = new Scanner(System.in);
16     private List<String> options = Arrays.asList("Display Bikes", "Add Bike", "Delete Bike",
17         "Update Bike", "EXIT");
18
19     public void start() {
20         String selection = "";
21
22         do {
23             printMenu();
24             selection = scanner.nextLine();
25             try {
26                 if (selection.equals("1")) {
27                     displayBikes();
28                 } else if (selection.equals("2")) {
29                     addBike();
30                 } else if (selection.equals("3")) {
31                     deleteBike();
32                 } else if (selection.equals("4")) {
33                     updateBike();
34                 }
35             }
36             catch (SQLException e) {

```

Application.java CycleDao.java Menu.java X Bike.java DBConnection.java

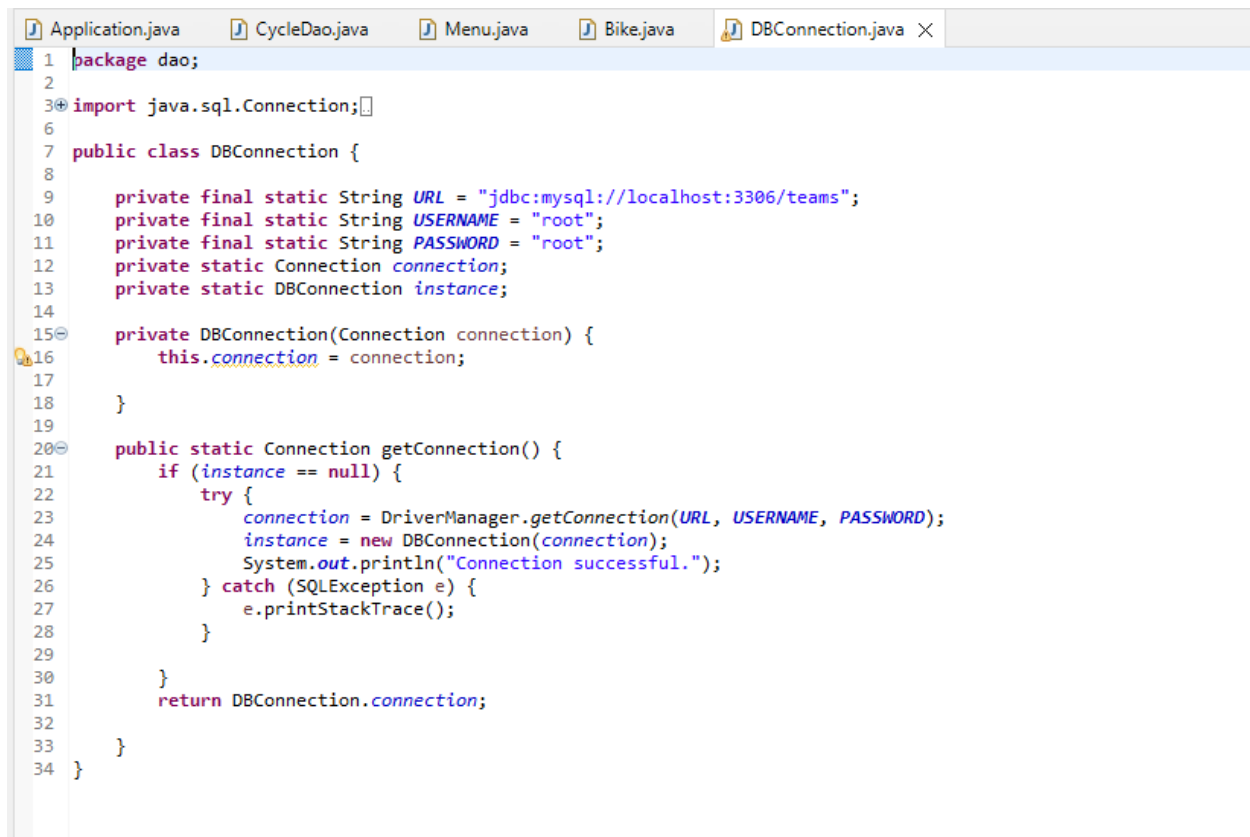
```
36         catch (SQLException e) {
37             e.printStackTrace();
38         }
39
40
41         System.out.println("Press enter to continue...");
42         scanner.nextLine();
43     } while (!selection.equals("5"));
44 }
45
46
47 private void printMenu() {
48     System.out.println("Select an Option:\n-----");
49     for (int i = 0; i < options.size(); i++) {
50         System.out.println(i + 1 + " " + options.get(i));
51     }
52 }
53
54
55 private void displayBikes() throws SQLException {
56     List<Bike> bikes = CycleDao.getBikes();
57     System.out.println("Year\t" + "Make\t" + "Model\t" + "CC\t" + "ID");
58     for (Bike bike : bikes) {
59         System.out.println(
60             bike.getYear() + "\t" + bike.getMake() + " "
61             + bike.getModel() + "\t "
62             + bike.getEngineSize() + "\t" + bike.getId());
63     }
64 }
65
66 private void addBike() throws SQLException {
67     System.out.print("Enter year of new Bike: ");
68     int year = Integer.parseInt(scanner.nextLine());
69     System.out.print("Enter make of new Bike: ");
70     String make = scanner.nextLine();
71     System.out.print("Enter model of new Bike: ");
72     String model = scanner.nextLine();
73     System.out.print("Enter engine size of new Bike: ");
74     int engineSize = Integer.parseInt(scanner.nextLine());
75     CycleDao.createNewBike(year, make, model, engineSize);
76 }
77
78 private void deleteBike() throws SQLException {
79     System.out.print("Enter Bike ID to Delete: ");
80     int id = Integer.parseInt(scanner.nextLine());
81     CycleDao.deleteBikeById(id);
```

Application.java CycleDao.java Menu.java X Bike.java DBConnection.java

```
52 }
53
54
55 private void displayBikes() throws SQLException {
56     List<Bike> bikes = CycleDao.getBikes();
57     System.out.println("Year\t" + "Make\t" + "Model\t" + "CC\t" + "ID");
58     for (Bike bike : bikes) {
59         System.out.println(
60             bike.getYear() + "\t" + bike.getMake() + " "
61             + bike.getModel() + "\t "
62             + bike.getEngineSize() + "\t" + bike.getId());
63     }
64 }
65
66 private void addBike() throws SQLException {
67     System.out.print("Enter year of new Bike: ");
68     int year = Integer.parseInt(scanner.nextLine());
69     System.out.print("Enter make of new Bike: ");
70     String make = scanner.nextLine();
71     System.out.print("Enter model of new Bike: ");
72     String model = scanner.nextLine();
73     System.out.print("Enter engine size of new Bike: ");
74     int engineSize = Integer.parseInt(scanner.nextLine());
75     CycleDao.createNewBike(year, make, model, engineSize);
76 }
77
78 private void deleteBike() throws SQLException {
79     System.out.print("Enter Bike ID to Delete: ");
80     int id = Integer.parseInt(scanner.nextLine());
81     CycleDao.deleteBikeById(id);
82 }
83 //
84 private void updateBike() throws SQLException {
85     System.out.print("Enter year of Bike to change: ");
86     int year = Integer.parseInt(scanner.nextLine());
87     System.out.print("Enter make of Bike to change: ");
88     String make = scanner.nextLine();
89     System.out.print("Enter model of new Bike to change: ");
90     String model = scanner.nextLine();
91     System.out.print("Enter engine size of Bike to change: ");
92     int engineSize = Integer.parseInt(scanner.nextLine());
93     System.out.print("Enter ID of bike you wish to change: ");
94     int id = Integer.parseInt(scanner.nextLine());
95     CycleDao.updateBikeById(id, year, make, model, engineSize);
96 }
97
98 }
99
```

Application.java CycleDao.java Menu.java Bike.java × DBConnection.java

```
1 package entities;
2
3 public class Bike {
4
5     private int id;
6     private int year;
7     private String make;
8     private String model;
9     private int engineSize;
10
11     public Bike(int id, int year, String make, String model, int engineSize) {
12         this.setId(id);
13         this.setYear(year);
14         this.setMake(make);
15         this.setModel(model);
16         this.setEngineSize(engineSize);
17     }
18     public int getId() {
19         return id;
20     }
21     public void setId(int id) {
22         this.id = id;
23     }
24     public int getYear() {
25         return year;
26     }
27     public void setYear(int year) {
28         this.year = year;
29     }
30     public String getMake() {
31         return make;
32     }
33     public void setMake(String make) {
34         this.make = make;
35     }
36     public String getModel() {
37         return model;
38     }
39     public void setModel(String model) {
40         this.model = model;
41     }
42     public int getEngineSize() {
43         return engineSize;
44     }
45     public void setEngineSize(int engineSize) {
46         this.engineSize = engineSize;
47     }
48 }
49
```



```
1 package dao;
2
3 import java.sql.Connection;
4
5
6
7 public class DBConnection {
8
9     private final static String URL = "jdbc:mysql://localhost:3306/teams";
10    private final static String USERNAME = "root";
11    private final static String PASSWORD = "root";
12    private static Connection connection;
13    private static DBConnection instance;
14
15    private DBConnection(Connection connection) {
16        this.connection = connection;
17    }
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 }
34 }
```

Screenshots of Running Application:

Application (3) [Java Application] C:\Program Files\Amazon Corretto\jdk17.0.2_8\bin\javaw.exe (Apr 28, 2022, 8:33:04 PM)

Connection successful.

Select an Option:

- 1) Display Bikes
- 2) Add Bike
- 3) Delete Bike
- 4) Update Bike
- 5) EXIT

1

Year	Make	Model	CC	ID
2018	Kawasaki	Vulcan	650	2
2005	Honda	Shadow	1100	3
1991	Honda	CR250	250	6
2022	Yamaha	MT-10	1000	7

Press enter to continue...

Select an Option:

- 1) Display Bikes
- 2) Add Bike
- 3) Delete Bike
- 4) Update Bike
- 5) EXIT

2

Enter year of new Bike: 2009

Enter make of new Bike: Suzuki

Enter model of new Bike: Bandit

Enter engine size of new Bike: 1250

Press enter to continue...

Select an Option:

- 1) Display Bikes
- 2) Add Bike
- 3) Delete Bike
- 4) Update Bike
- 5) EXIT

1

Year	Make	Model	CC	ID
2018	Kawasaki	Vulcan	650	2
2005	Honda	Shadow	1100	3
1991	Honda	CR250	250	6
2022	Yamaha	MT-10	1000	7
2009	Suzuki	Bandit	1250	8

Press enter to continue...

Select an Option:

- 1) Display Bikes
- 2) Add Bike
- 3) Delete Bike
- 4) Update Bike
- 5) EXIT

3

Enter BIke ID to Delete: 2

Press enter to continue...

Select an Option:

- 1) Display Bikes
- 2) Add Bike
- 3) Delete Bike
- 4) Update Bike
- 5) EXIT

1

Year	Make	Model	CC	ID
2005	Honda	Shadow	1100	3
1991	Honda	CR250	250	6
2022	Yamaha	MT-10	1000	7
2009	Suzuki	Bandit	1250	8

Press enter to continue...

```

Select an Option:
-----
1) Display Bikes
2) Add Bike
3) Delete Bike
4) Update Bike
5) EXIT
1
Year    Make    Model  CC    ID
2005    Honda  Shadow  1100  3
1991    Honda  CR250   250   6
2022    Yamaha MT-10  1000  7
2009    Suzuki Bandit 1250  8
Press enter to continue...

Select an Option:
-----
1) Display Bikes
2) Add Bike
3) Delete Bike
4) Update Bike
5) EXIT
4
Enter year of Bike to change: 2018
Enter make of Bike to change: Suzuki
Enter model of new Bike to change: SV650
Enter engine size of Bike to change: 650
Enter ID of bike you wish to change: 3
Press enter to continue...

Select an Option:
-----
1) Display Bikes
2) Add Bike
3) Delete Bike
4) Update Bike
5) EXIT
1
Year    Make    Model  CC    ID
2018    Suzuki SV650   650   3
1991    Honda  CR250   250   6
2022    Yamaha MT-10  1000  7
2009    Suzuki Bandit 1250  8
Press enter to continue...

```

URL to GitHub Repository:

https://github.com/dfleeman/Week10_Assignment