Thomas Le

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....
ddrsongs

Create a new Java project in Eclipse.

Done

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

Done

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.

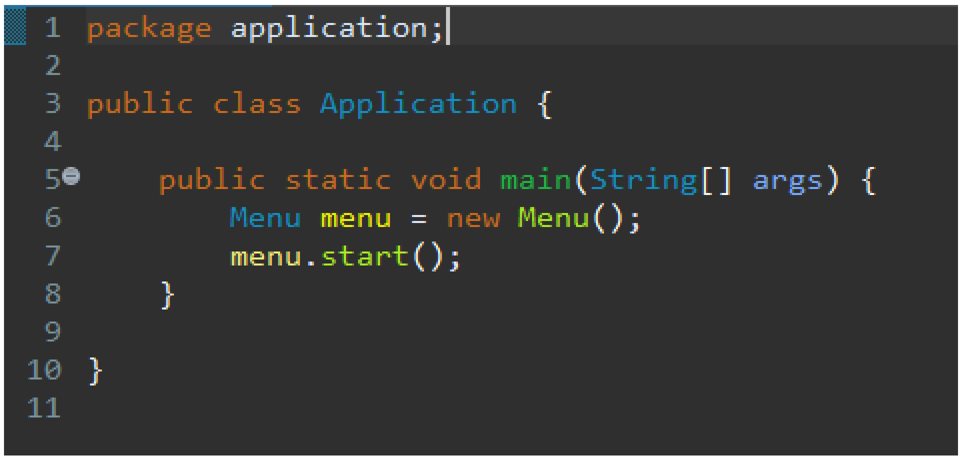
You need an option for each of the CRUD operations (Create, Read, Update, and Delete).

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.

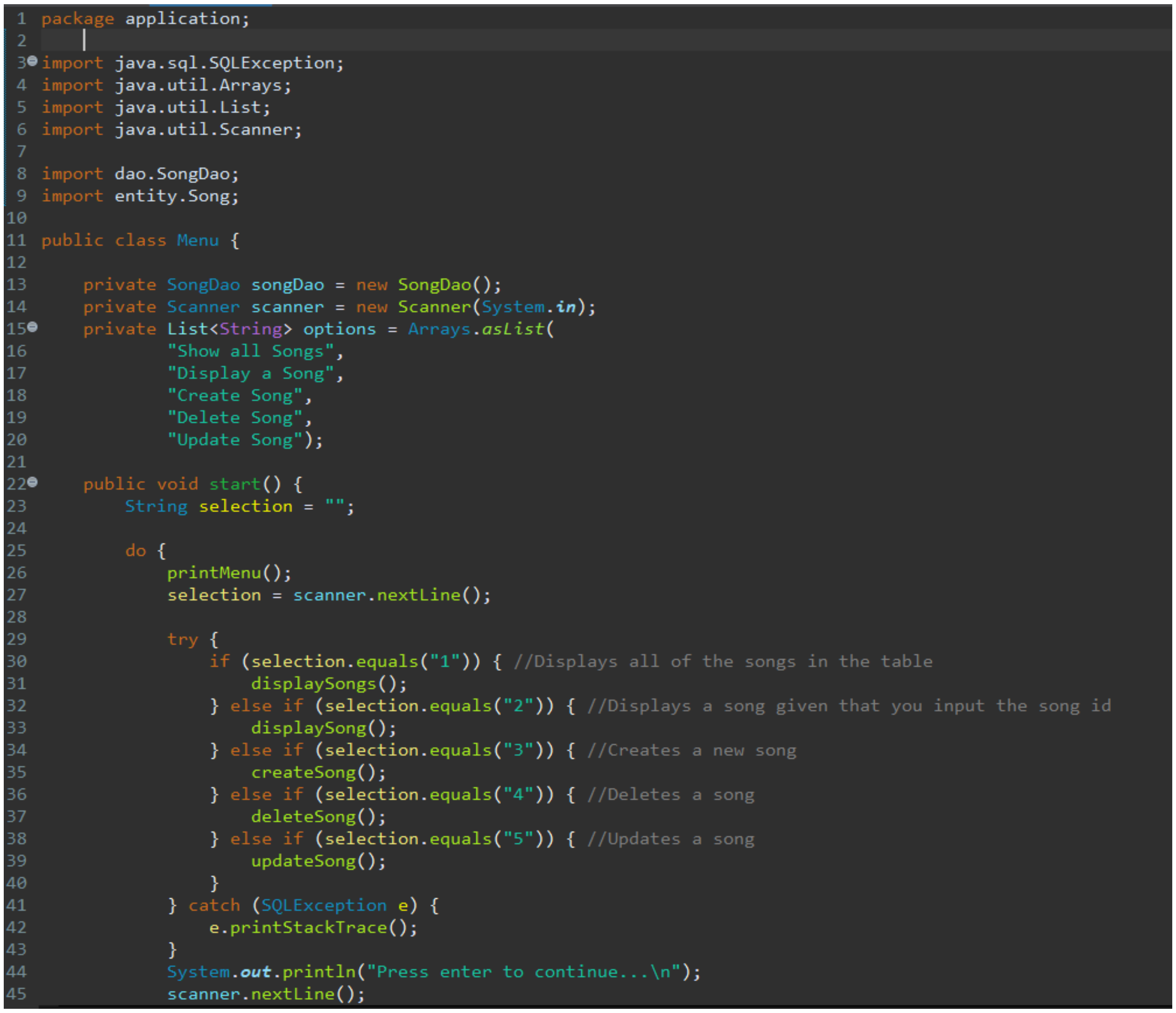
Screenshots of Code:

Application

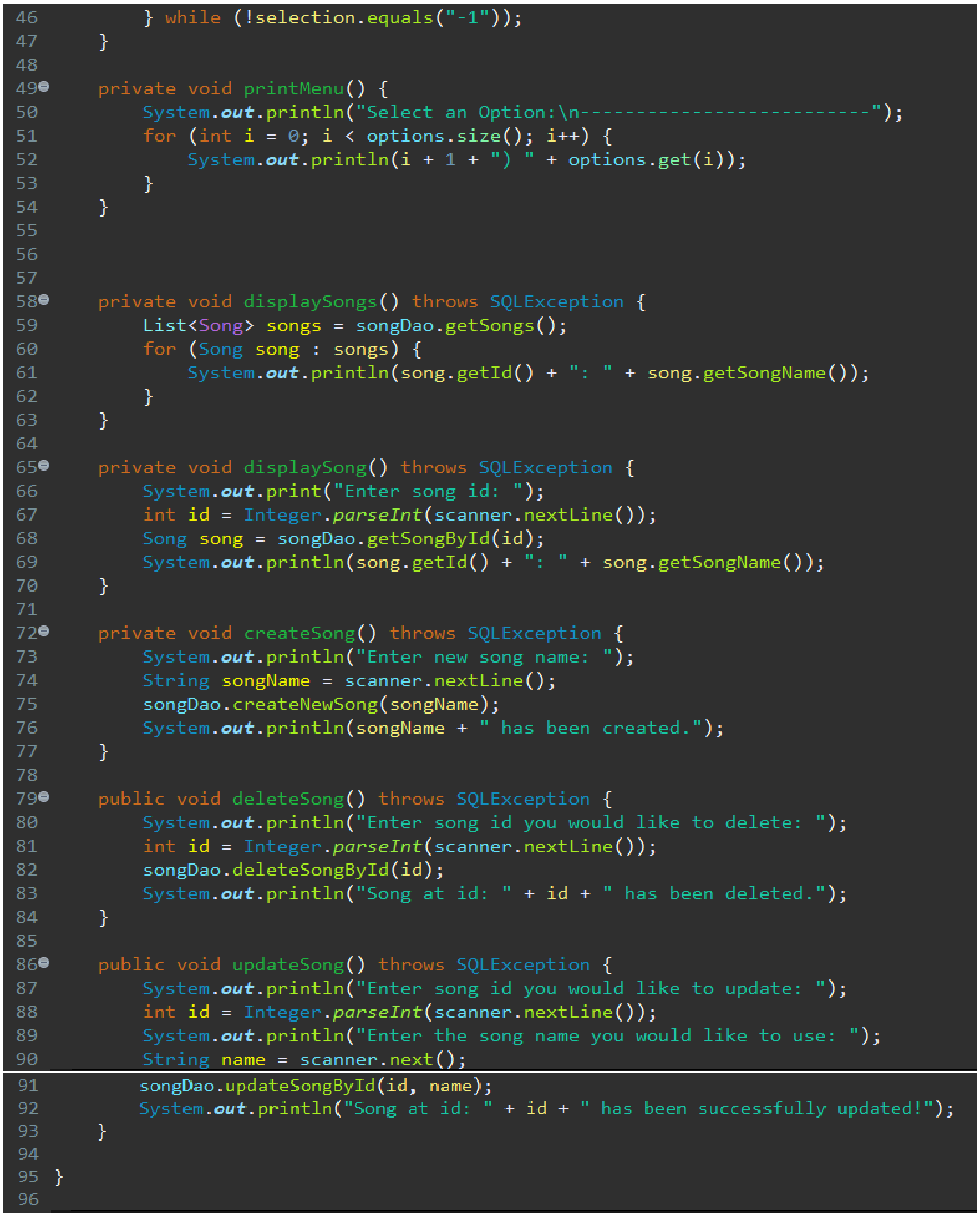


```java
package application;

public class Application {

    public static void main(String[] args) {
        Menu menu = new Menu();
        menu.start();
    }

}

```

Menu

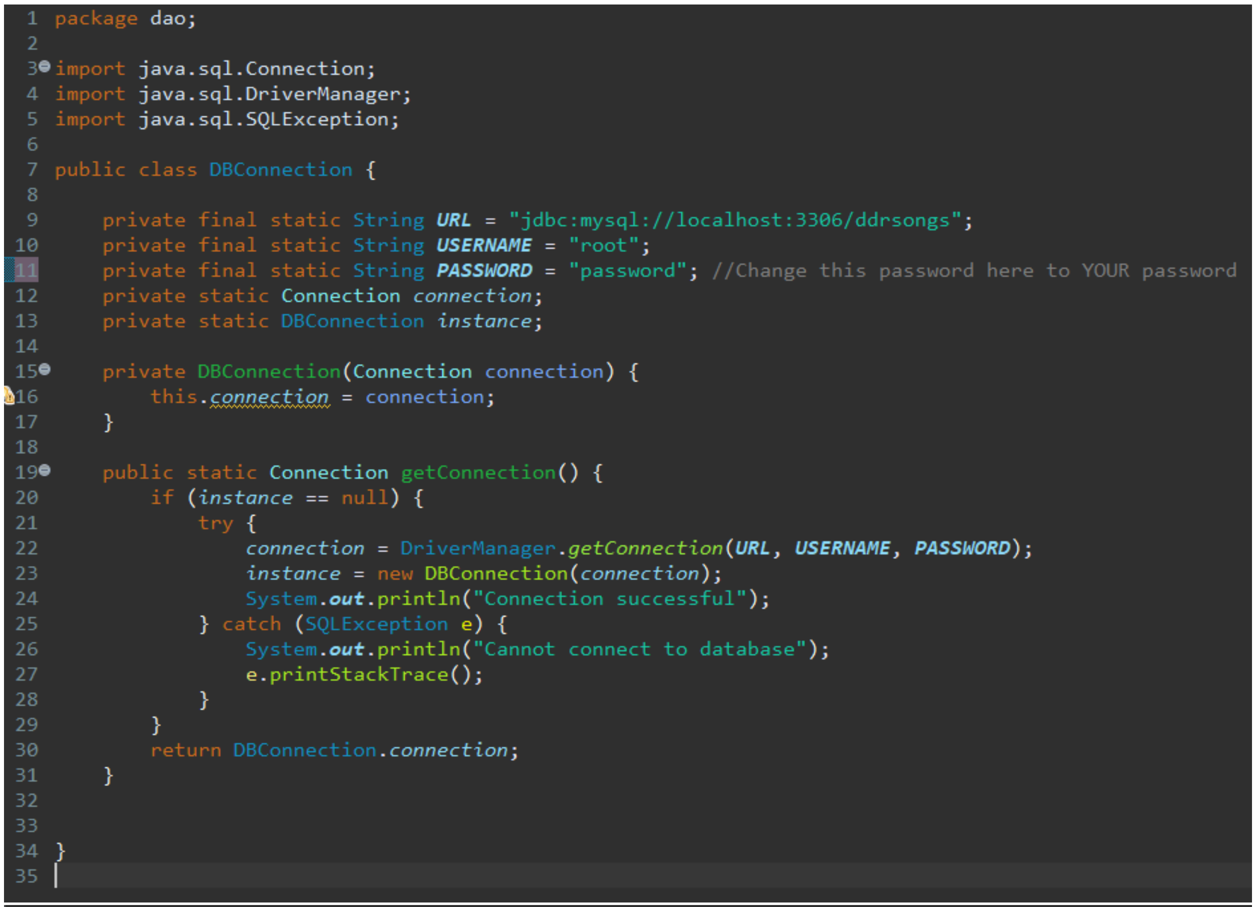


```java
package application;

import java.sql.SQLException;
import java.util.Arrays;
import java.util.List;
import java.util.Scanner;

import dao.SongDao;
import entity.Song;

public class Menu {

    private SongDao songDao = new SongDao();
    private Scanner scanner = new Scanner(System.in);
    private List<String> options = Arrays.asList(
            "Show all Songs",
            "Display a Song",
            "Create Song",
            "Delete Song",
            "Update Song");

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

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

            try {
                if (selection.equals("1")) { //Displays all of the songs in the table
                    displaySongs();
                } else if (selection.equals("2")) { //Displays a song given that you input the song id
                    displaySong();
                } else if (selection.equals("3")) { //Creates a new song
                    createSong();
                } else if (selection.equals("4")) { //Deletes a song
                    deleteSong();
                } else if (selection.equals("5")) { //Updates a song
                    updateSong();
                }
            } catch (SQLException e) {
                e.printStackTrace();
            }
            System.out.println("Press enter to continue...\n");
            scanner.nextLine();
```

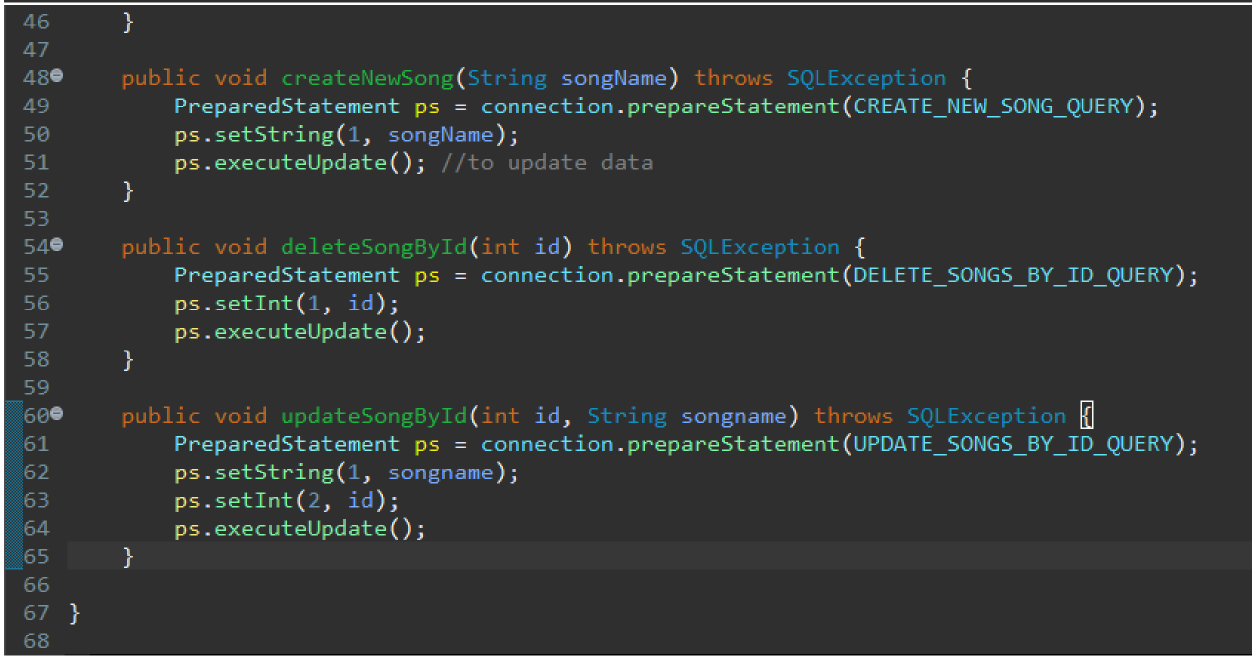


```java
        } while (!selection.equals("-1"));
    }

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



    private void displaySongs() throws SQLException {
        List<Song> songs = songDao.getSongs();
        for (Song song : songs) {
            System.out.println(song.getId() + ": " + song.getSongName());
        }
    }

    private void displaySong() throws SQLException {
        System.out.print("Enter song id: ");
        int id = Integer.parseInt(scanner.nextLine());
        Song song = songDao.getSongById(id);
        System.out.println(song.getId() + ": " + song.getSongName());
    }

    private void createSong() throws SQLException {
        System.out.println("Enter new song name: ");
        String songName = scanner.nextLine();
        songDao.createNewSong(songName);
        System.out.println(songName + " has been created.");
    }

    public void deleteSong() throws SQLException {
        System.out.println("Enter song id you would like to delete: ");
        int id = Integer.parseInt(scanner.nextLine());
        songDao.deleteSongById(id);
        System.out.println("Song at id: " + id + " has been deleted.");
    }

    public void updateSong() throws SQLException {
        System.out.println("Enter song id you would like to update: ");
        int id = Integer.parseInt(scanner.nextLine());
        System.out.println("Enter the song name you would like to use: ");
        String name = scanner.next();
        songDao.updateSongById(id, name);
        System.out.println("Song at id: " + id + " has been successfully updated!");
    }

}

```

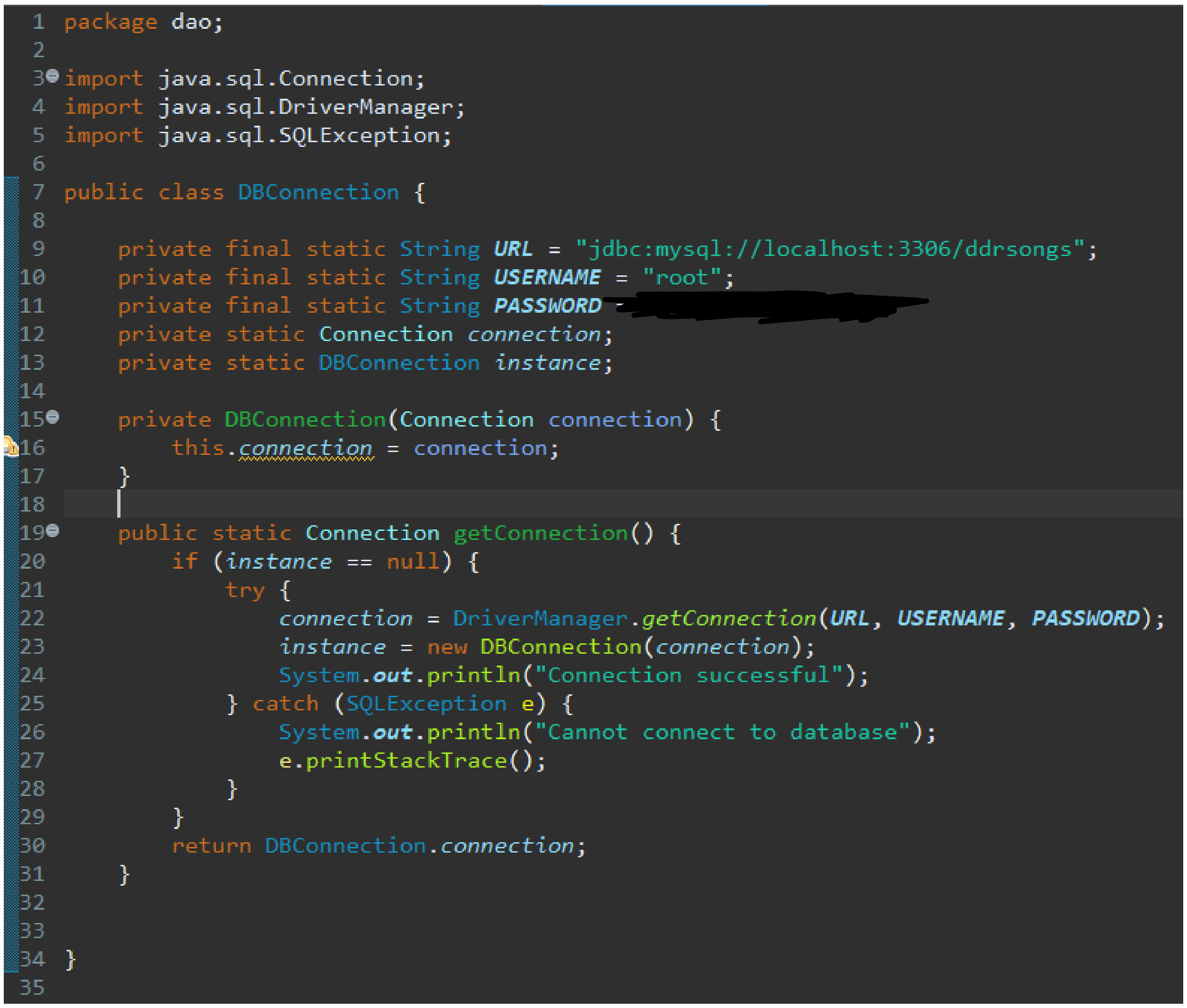
SongDao



```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/ddrsongs";
    private final static String USERNAME = "root";
    private final static String PASSWORD = "password"; //Change this password here to YOUR password
    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");
            } catch (SQLException e) {
                System.out.println("Cannot connect to database");
                e.printStackTrace();
            }
        }
        return DBConnection.connection;
    }


}

```

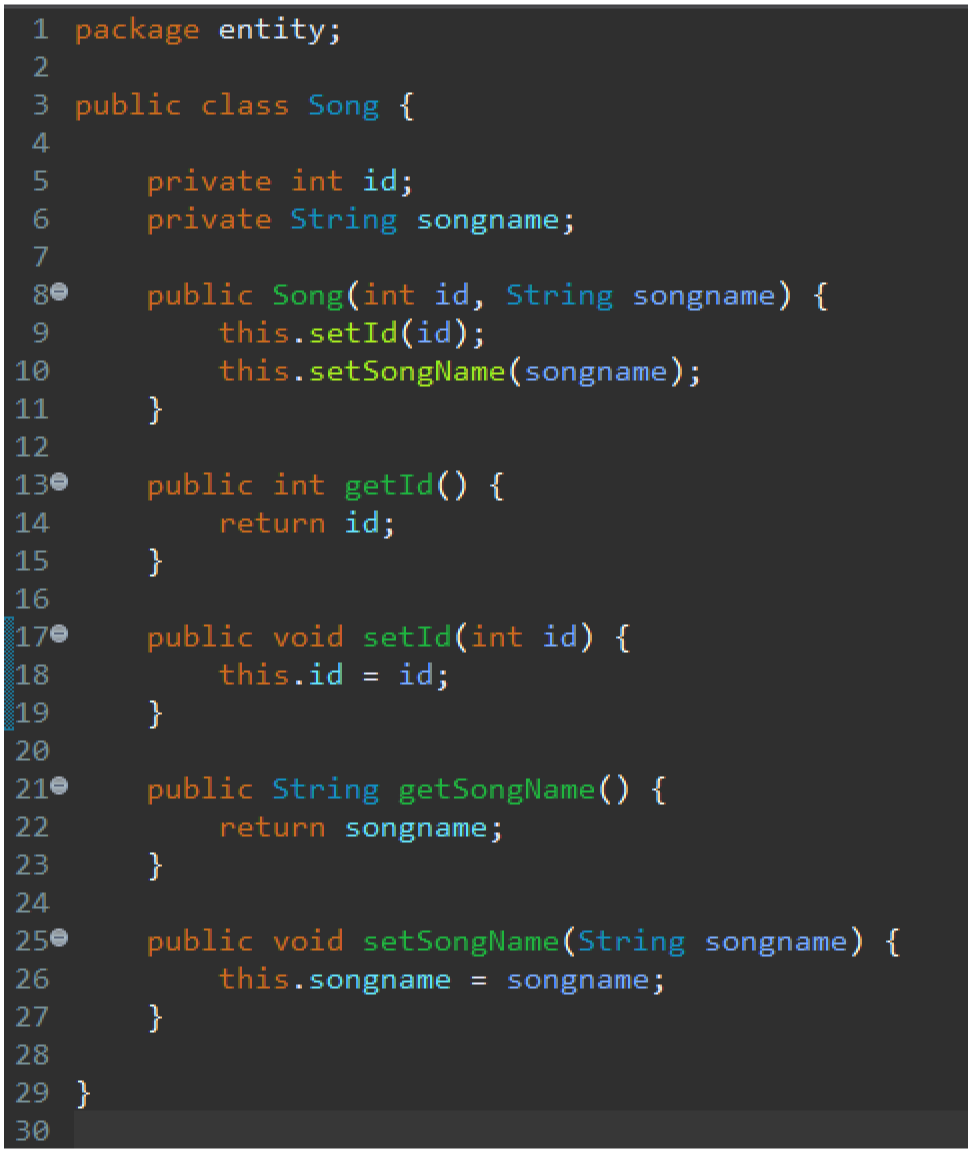


```java
    }

    public void createNewSong(String songName) throws SQLException {
        PreparedStatement ps = connection.prepareStatement(CREATE_NEW_SONG_QUERY);
        ps.setString(1, songName);
        ps.executeUpdate(); //to update data
    }

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

    public void updateSongById(int id, String songname) throws SQLException {
        PreparedStatement ps = connection.prepareStatement(UPDATE_SONGS_BY_ID_QUERY);
        ps.setString(1, songname);
        ps.setInt(2, id);
        ps.executeUpdate();
    }

}

```



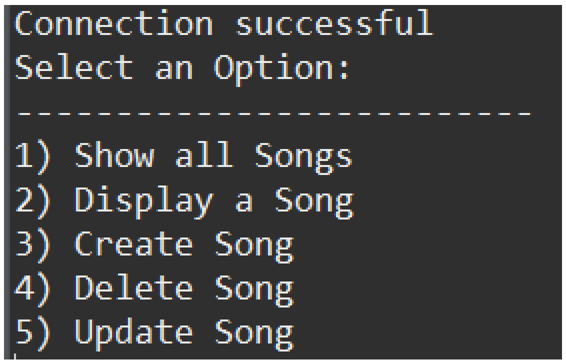
```java
package dao;

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

public class DBConnection {

    private final static String URL = "jdbc:mysql://localhost:3306/ddrsongs";
    private final static String USERNAME = "root";
    private final static String PASSWORD [redacted]
    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");
            } catch (SQLException e) {
                System.out.println("Cannot connect to database");
                e.printStackTrace();
            }
        }
        return DBConnection.connection;
    }


}

```

SONG



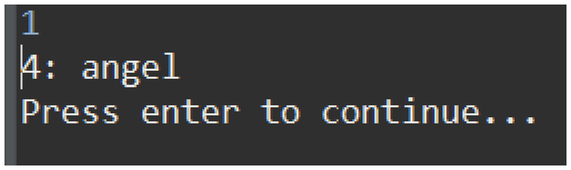
```java
package entity;

public class Song {

    private int id;
    private String songname;

    public Song(int id, String songname) {
        this.setId(id);
        this.setSongName(songname);
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getSongName() {
        return songname;
    }

    public void setSongName(String songname) {
        this.songname = songname;
    }

}
```

Screenshots of Running Application:
1. Showing the application when it’s ran



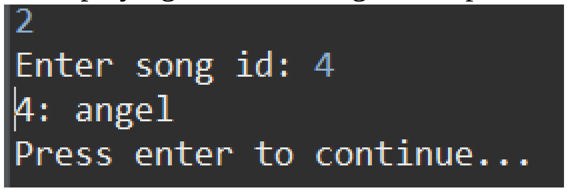
```
Connection successful
Select an Option:
-------------------------
1) Show all Songs
2) Display a Song
3) Create Song
4) Delete Song
5) Update Song
```

2. Displaying the show all songs function.



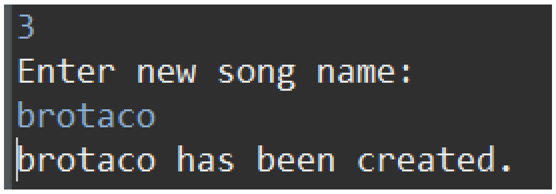
```
1
4: angel
Press enter to continue...
```

3. Displaying the show song with inputted id function.



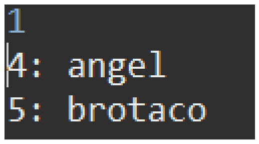
```
2
Enter song id: 4
4: angel
Press enter to continue...
```

4. Entering a new song name called brotaco



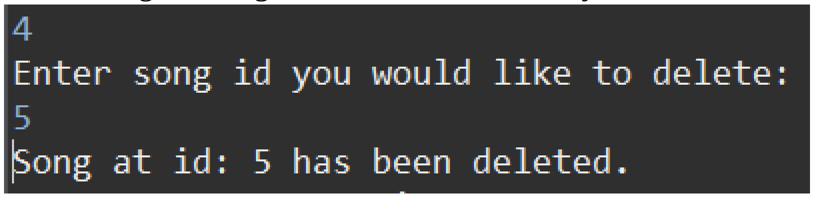
```
3
Enter new song name:
brotaco
brotaco has been created.
```

5. Verifying that brotaco is inside of the table



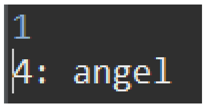
```
1
4: angel
5: brotaco
```

6. Deleting the song brotaco at id 5 that was just created.



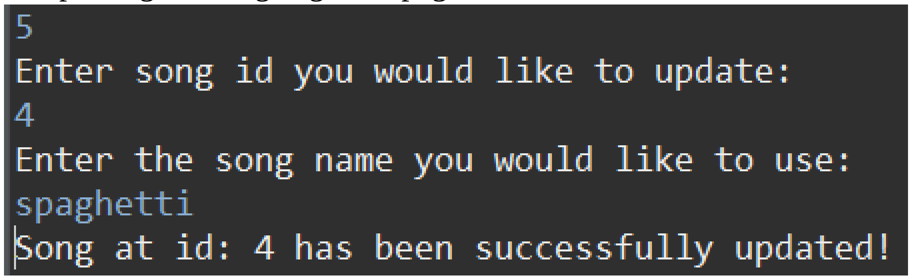
```
4
Enter song id you would like to delete:
5
Song at id: 5 has been deleted.
```

7. Verifying it was deleted from the table



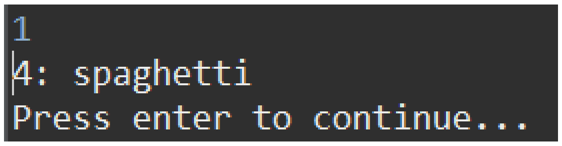
```
1
4: angel
```

8. Updating the song angel to spaghetti



```
5
Enter song id you would like to update:
4
Enter the song name you would like to use:
spaghetti
Song at id: 4 has been successfully updated!
```

9. Verifying that it actually has been updated.



```
1
4: spaghetti
Press enter to continue...
```

URL to GitHub Repository:

https://github.com/AkemiTCGyt/PromineoTechWeek10CodingAssignment