Relational Databases with MySQL Week 6 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:

This week you will be working together as a **team** to create a full CRUD application.

Your console CRUD application will need to use a database to store all the application data.

As a team, decide what you want your project to do. Get instructor approval early in the week before beginning development.

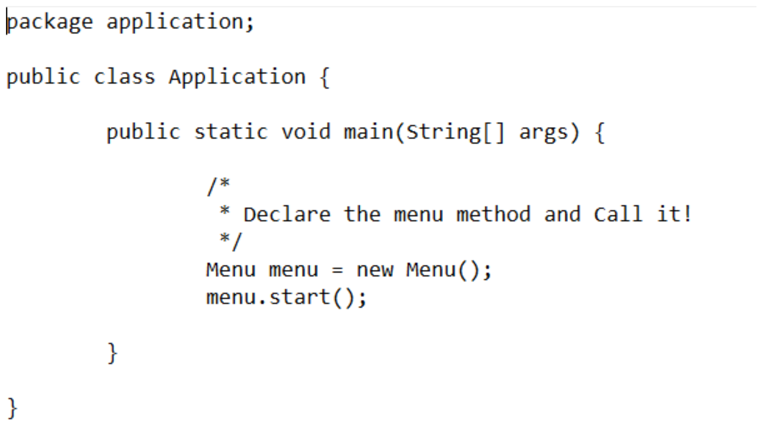
You need to have at least 3 entities.

Users should be able to interact via the console (i.e. Scanner(System.in)))

Use git to collaborate.

Everyone will be graded on their individual contributions.

Screenshots of Code:



```java
package application;

public class Application {

        public static void main(String[] args) {

                /*
                 * Declare the menu method and Call it!
                 */
                Menu menu = new Menu();
                menu.start();

        }

}
```

```java
package application;

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

import dao.AlbumDao;
import dao.ArtistDao;
import dao.CertificationDao;
import entity.Album;
import entity.Artist;
import entity.Certification;

public class Menu {
                private AlbumDao albumDao = new AlbumDao();
                private ArtistDao artistDao = new ArtistDao();
                private CertificationDao certificationDao = new CertificationDao();
                private final String DATABASE_NAME = "recording_artists";
                private Scanner scanner = new Scanner(System.in);
                private List<String> options = Arrays.asList("Album Menu",
                                                                                                    "Artist Menu",
                                                                                                    "Certification Menu");

                private List<String> album_options = Arrays.asList("Display All Albums",
                                                                                                              "Add A New Album",
                                                                                                          "Delete An Album",
                                                                                                          "Update An Album");


                private List<String> artist_options = Arrays.asList("Display All Artists",
                                                                                                           "Add A New Artist",
                                                                                                           "Delete An Artist",
                                                                                                           "Update An Artist");


                private List<String> cert_options = Arrays.asList("Display All Certifications",
                                                                                                         "Add A New Certification",
                                                                                                         "Delete A Certification",
                                                                                                         "Update A Certification");


                public void start() {
```

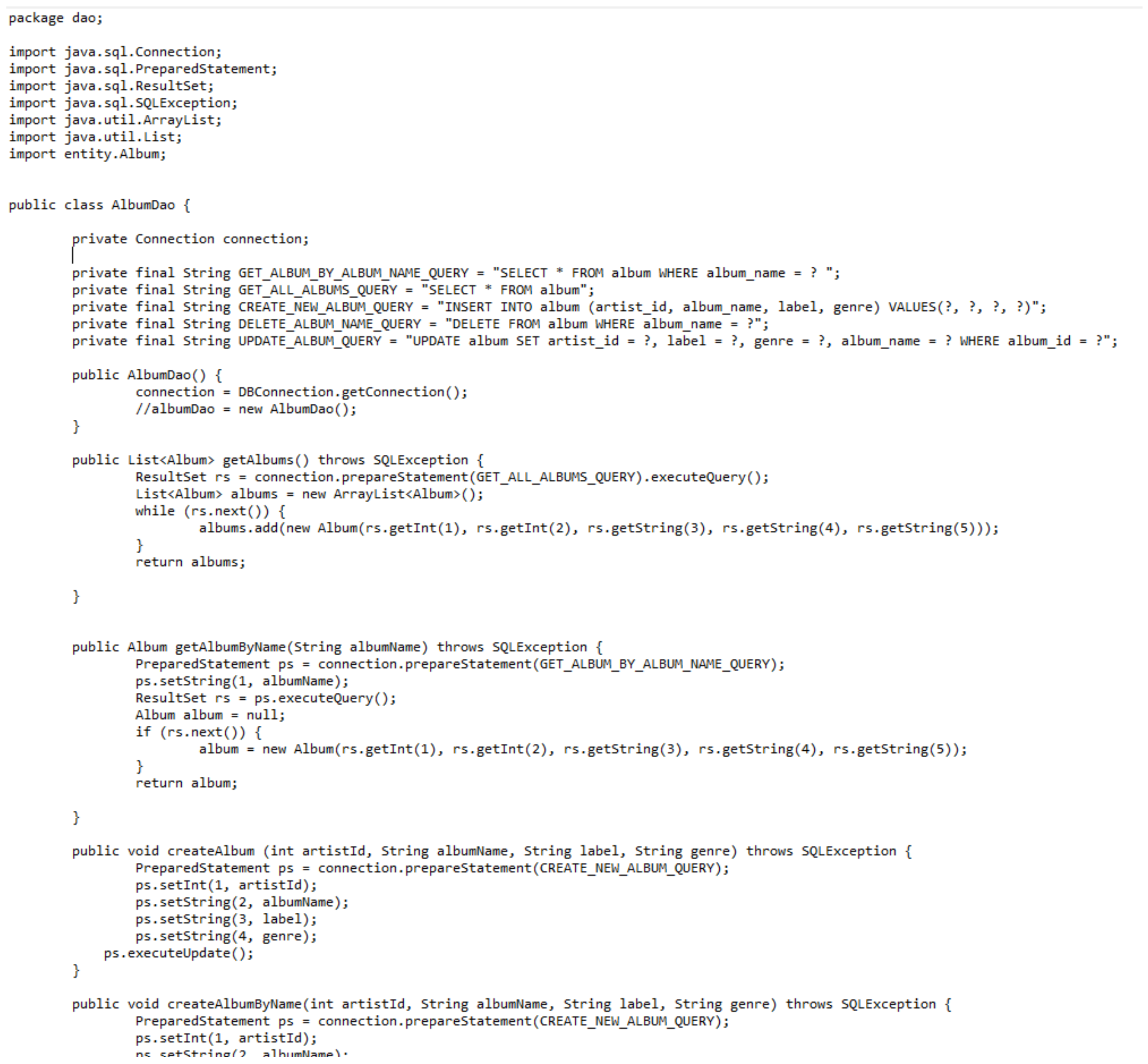
```
String selection = "";
String subselection = "";


do {
        System.out.println("---------------------------");
        System.out.println("MAIN " + DATABASE_NAME + " MENU");
        System.out.println("---------------------------");
        printMenu(options);
        scanner = new Scanner(System.in);
        selection = scanner.nextLine();

        try {

                if (selection.equals("1") ) {
                        System.out.println("----------------------");
                        System.out.println("ALBUM Information Menu");
                        System.out.println("----------------------");
                        do {
                                printMenu(album_options);
                                scanner = new Scanner(System.in);
                                subselection = scanner.nextLine();

                                if (subselection.equals("1") ) {
                                        System.out.println("\tDisplaying all albums...\n");
                                        displayAllAlbums();
                                } else if (subselection.equals("2") ) {
                                        System.out.println("\tAdding an album...\n");
                                        addNewAlbum();
                                } else if (subselection.equals("3") ) {
                                        System.out.println("\tDeleting an album...\n");
                                        deleteAlbum();
                                } else if (subselection.equals("4") ) {
                                        System.out.println("\tUpdating an album...\n");
                                        updateAlbum();
                                } else if (!(subselection.equals("-1"))) {
                                                System.out.println("Invalid Option!");
                                }

                        } while (!(subselection.equals("-1")));

                } else if (selection.equals("2") ) {
                        System.out.println("-----------------------");
                        System.out.println("ARTIST Information Menu");
                        System.out.println("-----------------------");

                        do {
                                printMenu(artist_options);
                                scanner = new Scanner(System.in);
                                subselection = scanner.nextLine();

                            if (subselection.equals("1") ) {
                                System.out.println("\tDisplaying all artists...\n");
                                        displayAllArtists();
                                } else if (subselection.equals("2") ) {
                                        System.out.println("\tAdding an artist...\n");
                                        addNewArtist();
                                } else if (subselection.equals("3") ) {
                                        System.out.println("\tDeleting an artist...\n");
                                        deleteArtist();
                                } else if (subselection.equals("4") ) {
                                        System.out.println("\tUpdating an artist...\n");
```

```
                                                } else if (subselection.equals("4") ) {
                                                        System.out.println("\tUpdating an artist...\n");
                                                        updateArtist();
                                                } else if (!(subselection.equals("-1"))) {
                                                                System.out.println("Invalid Option!");
                                                }
                                        } while (!(subselection.equals("-1")));


                                } else if (selection.equals("3") ) {
                                        System.out.println("------------------------------");
                                        System.out.println("CERTIFICATION Information Menu");
                                        System.out.println("------------------------------");

                                        do {
                                                printMenu(cert_options);
                                                scanner = new Scanner(System.in);
                                                subselection = scanner.nextLine();

                                                if (subselection.equals("1") ) {
                                                        System.out.println("\tDisplaying all certifications...\n");
                                                        displayAllCerts();
                                                } else if (subselection.equals("2") ) {
                                                        System.out.println("\tAdding a certification...\n");
                                                        addNewCert();
                                                } else if (subselection.equals("3") ) {
                                                        System.out.println("\tDeleting a certification...\n");
                                                        deleteCert();
                                                } else if (subselection.equals("4") ) {
                                                        System.out.println("\tUpdating a certification...\n");
                                                        updateCert();
                                                } else if (!(subselection.equals("-1"))) {
                                                                        System.out.println("Invalid Option!");
                                                }

                                        } while (!(subselection.equals("-1")));


                                } else if (!(selection.equals("-1"))) {
                                                System.out.println("Invalid Option!");
                                }

                        } catch(Exception e) {
                                        System.out.println("Error!");
                            e.printStackTrace();
                        }

                } while (!(selection.equals("-1")));
                System.out.println("\n\nEnd of program...\n\nThanks for using the " + DATABASE_NAME + " database!");


        }


/*
 * Method:  displayAllAlbums()
 */
private void displayAllAlbums() throws SQLException {
        /*
         * No need for input... print all album data
         */
        List<Album> albums = albumDao.getAlbums();
```

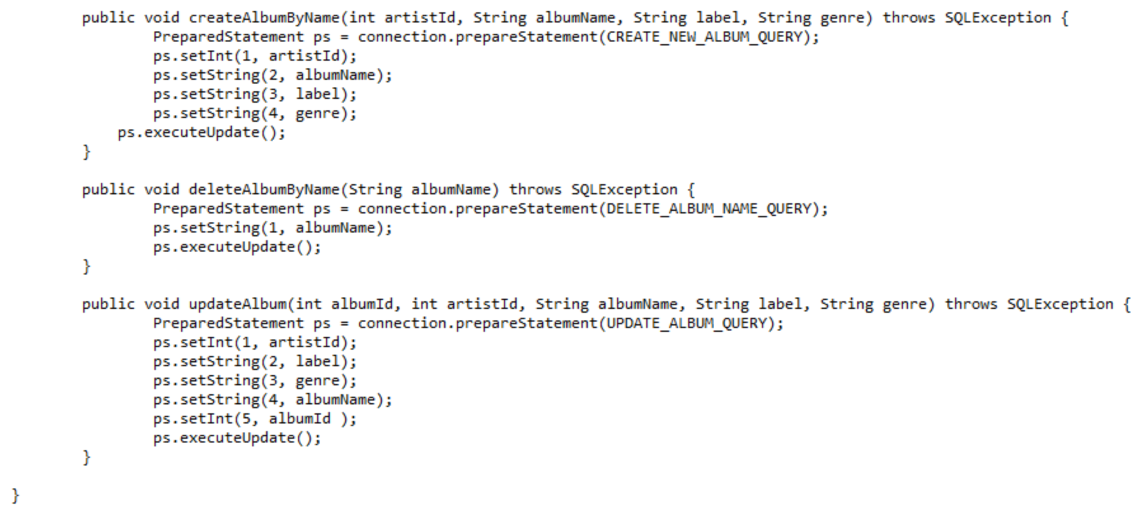
```
		 */
		List<Album> albums = albumDao.getAlbums();
		for (Album album : albums) {
			Artist artist = artistDao.getArtistById(album.getArtistId());
			System.out.println("\nName: " + album.getAlbumName() + "\n\tId: " + album.getAlbumId() + "\tArtist: " + artist.getArtistName()
					+ "\n\tLabel: " + album.getLabel() + "\tGenre: " + album.getGenre());
		}
	}

	/*
	 * Method:  addNewAlbum()
	 */
	private void addNewAlbum() throws SQLException {
		/*
		 * prompt user for all new album data
		 */
		System.out.print("Enter Album Name: ");
		String albumName = scanner.nextLine();
		System.out.print("Enter Artist: ");
		String artistName = scanner.nextLine();   // Use this to find the Artist_ID by calling
												  // artistDao.getArtistByName(artistName).getArtist_id();
		Artist artist = artistDao.getArtistByName(artistName);
		if (artist == null) {
			artistDao.createNewArtist(artistName);
			artist = artistDao.getArtistByName(artistName);
		}
		System.out.print("Enter Label: ");
		String labelName = scanner.nextLine();
		System.out.print("Enter Genre: ");
		String genre = scanner.nextLine();

		albumDao.createAlbum(artist.getArtistId(), albumName, labelName, genre);
	}

	/*
	 * Method:  deleteAlbum()
	 */
	private void deleteAlbum() throws SQLException {
		/*
		 * prompt user for album name, and confirm that they want to delete?:
		 */
		System.out.print("Enter Name of Album to Delete: ");
		String albumName = scanner.nextLine();
		Album album = albumDao.getAlbumByName(albumName);
		if (album == null) {
			System.out.println("Album doesn't exist!");
		} else {
			System.out.println("Deleting Album...");
			albumDao.deleteAlbumByName(albumName);
		}



	}

	/*
	 * Method:  updateAlbum()
	 */
	private void updateAlbum() throws SQLException {
		/*
		 * prompt user for album name, and possible input changes:
		 */
```

```
                String albumName = scanner.nextLine();
                Album album = albumDao.getAlbumByName(albumName);
                if (album == null) {
                        System.out.println("Album doesn't exist!");
                } else {
                        int albumId = album.getAlbumId();
                        System.out.print("Enter Change to Album Name: ");
                        String newName = scanner.nextLine();

                        if (newName.equals("")) {
                                newName = albumName;
                        }

                        System.out.print("Enter Name of Label to Update: ");
                        String label = scanner.nextLine();
                        System.out.print("Enter Genre to Update: ");
                        String genre = scanner.nextLine();

                        System.out.println("Updating Album...");
                        albumDao.updateAlbum(albumId, album.getArtistId(),newName,label,genre);
                }

        }

        /*
         * Method:  displayAllArtists()
         */
        private void displayAllArtists() throws SQLException {
                /*
                 * No need for input... print all artist data
                 */
                List<Artist> artists = artistDao.getArtists();
                for (Artist artist : artists) {
                        System.out.println("Id: " + artist.getArtistId() + " \tName: " + artist.getArtistName());
                }

        }

        /*
         * Method:  addNewArtist()
         */
        private void addNewArtist() throws SQLException {
                /*
                 * prompt user for all new artist data
                 */
                System.out.print("Enter New Artist: ");
                String artist_name = scanner.nextLine();
                artistDao.createNewArtist(artist_name);
        }


        /*
         * Method:  deleteArtist()
         */
        private void deleteArtist() throws SQLException {
                /*
                 * prompt user for artist name, and confirm that they want to delete?:
                 */
                System.out.print("Enter Name of Artist to Delete: ");
                String artistName = scanner.nextLine();

                System.out.println("Deleting Artist...");
```

```
	} else {


		System.out.println("Id: " + artist.getArtistId() + " \tName: " + artist.getArtistName());

		System.out.println("\t**  This will also delete all of the associated data in the " + DATABASE_NAME + " database!  **\n");
		System.out.print("Would you like to proceed, yes or no?  ");
		String response = scanner.nextLine();
		if (response.equalsIgnoreCase("yes")) {
			artistDao.deleteArtist(artistName);
		} else {
			System.out.println("Delete Artist not performed!");
		}
	}
}

/*
 * Method:  updateArtist()
 */
private void updateArtist() throws SQLException {
	/*
	 * prompt user for artist name to change, and new artist name
	 */
	System.out.print("Enter Name of Artist to Update: ");
	String artistName = scanner.nextLine();
	System.out.println("Updating Artist...");
	Artist artist = artistDao.getArtistByName(artistName);
	if (artist == null) {
		System.out.println("Artist Not Found in database: " + DATABASE_NAME);
		System.out.println("Update Artist not performed!");

	} else {

		System.out.println("Id: " + artist.getArtistId() + " \tName: " + artist.getArtistName());

		System.out.println();
		System.out.print("Enter NEW Name of Artist to Update: ");
		String newName = scanner.nextLine();
		if (newName == null) {
			System.out.println("Update Artist not performed!");
		} else {
			artistDao.updateArtist(artistName,newName);
		}
	}
}

/*
 * Method:  displayAllCerts()
 */
	private void displayAllCerts() throws SQLException {
		List<Certification> certification = certificationDao.displayAllCerts();
			for (Certification certs : certification) {
			System.out.println(certs.getCertId() + " " + certs.getAlbumId() + " " + certs.getCertStatus() + " " + certs.getCertDate());
				}
}


/*
 * Method:  addNewCert()
 */
private void addNewCert() throws SQLException {
	private void addNewCert() throws SQLException {

		System.out.print("Enter Name of Album for New Certification: ");
		String albumName = scanner.nextLine();
		Album album = albumDao.getAlbumByName(albumName);
		if (album == null) {
			System.out.println("Album Not Found in database: " + DATABASE_NAME);
			System.out.println("Add New Certification not performed!");
		} else {
			System.out.println("Enter New Certification...");
			String certStatus = scanner.nextLine();
			System.out.println("Enter Certifcation Date..");
			String certDate = scanner.nextLine();
			certificationDao.addNewCert(album.getAlbumId(), certStatus, certDate);
		}
	}

	/*
	 * Method:  deleteCert()
	 */
	private void deleteCert() throws SQLException {
		System.out.println("Enter Certification Id to delete....");
		int certId = Integer.parseInt(scanner.nextLine());
		certificationDao.deleteCert(certId);
	}

	/*
	 * Method:  updateCert()
	 */

	private void updateCert() throws SQLException {

		System.out.println("Enter Certification ID to update...");
		int certId = Integer.parseInt(scanner.nextLine());
		System.out.println("Enter New Certification Status...");
		String certStatus = scanner.nextLine();
		System.out.println("Enter New Certification Date...");
		String certDate = scanner.nextLine();
		certificationDao.updateCert(certId, certStatus, certDate);


	}


	/*
	 * Method:  printMenu()
	 */
	private void printMenu(List<String> options) {
		System.out.println("\n------------------------------------------------------------");
		System.out.println("\tPlease SELECT AN OPTION...\n\tAll requests are on the " + DATABASE_NAME + " database!\n------------------------------------------------------------");
		for (int i = 0; i<options.size(); i++) {
				System.out.println("   "+(i+1) + ") " + options.get(i));
			}
		System.out.println("  -1) Exit Menu");
		System.out.println("------------------------------------------------------------");
	}



}
```



```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import entity.Album;


public class AlbumDao {

        private Connection connection;

        private final String GET_ALBUM_BY_ALBUM_NAME_QUERY = "SELECT * FROM album WHERE album_name = ? ";
        private final String GET_ALL_ALBUMS_QUERY = "SELECT * FROM album";
        private final String CREATE_NEW_ALBUM_QUERY = "INSERT INTO album (artist_id, album_name, label, genre) VALUES(?, ?, ?, ?)";
        private final String DELETE_ALBUM_NAME_QUERY = "DELETE FROM album WHERE album_name = ?";
        private final String UPDATE_ALBUM_QUERY = "UPDATE album SET artist_id = ?, label = ?, genre = ?, album_name = ? WHERE album_id = ?";

        public AlbumDao() {
                connection = DBConnection.getConnection();
                //albumDao = new AlbumDao();
        }

        public List<Album> getAlbums() throws SQLException {
                ResultSet rs = connection.prepareStatement(GET_ALL_ALBUMS_QUERY).executeQuery();
                List<Album> albums = new ArrayList<Album>();
                while (rs.next()) {
                        albums.add(new Album(rs.getInt(1), rs.getInt(2), rs.getString(3), rs.getString(4), rs.getString(5)));
                }
                return albums;

        }


        public Album getAlbumByName(String albumName) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(GET_ALBUM_BY_ALBUM_NAME_QUERY);
                ps.setString(1, albumName);
                ResultSet rs = ps.executeQuery();
                Album album = null;
                if (rs.next()) {
                        album = new Album(rs.getInt(1), rs.getInt(2), rs.getString(3), rs.getString(4), rs.getString(5));
                }
                return album;

        }

        public void createAlbum (int artistId, String albumName, String label, String genre) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(CREATE_NEW_ALBUM_QUERY);
                ps.setInt(1, artistId);
                ps.setString(2, albumName);
                ps.setString(3, label);
                ps.setString(4, genre);
            ps.executeUpdate();
        }

        public void createAlbumByName(int artistId, String albumName, String label, String genre) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(CREATE_NEW_ALBUM_QUERY);
                ps.setInt(1, artistId);
                ps.setString(2, albumName);
```



```java
        public void createAlbumByName(int artistId, String albumName, String label, String genre) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(CREATE_NEW_ALBUM_QUERY);
                ps.setInt(1, artistId);
                ps.setString(2, albumName);
                ps.setString(3, label);
                ps.setString(4, genre);
            ps.executeUpdate();
        }

        public void deleteAlbumByName(String albumName) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(DELETE_ALBUM_NAME_QUERY);
                ps.setString(1, albumName);
                ps.executeUpdate();
        }

        public void updateAlbum(int albumId, int artistId, String albumName, String label, String genre) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(UPDATE_ALBUM_QUERY);
                ps.setInt(1, artistId);
                ps.setString(2, label);
                ps.setString(3, genre);
                ps.setString(4, albumName);
                ps.setInt(5, albumId );
                ps.executeUpdate();
        }

}
```

```java
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import entity.Artist;

public class ArtistDao {

	private Connection connection;
	private final String GET_ARTISTS_QUERY = "SELECT * FROM artist";
	private final String GET_ARTIST_BY_NAME_QUERY = "SELECT * FROM artist WHERE artist_name = ?";
	private final String GET_ARTIST_BY_ID_QUERY = "SELECT * FROM artist WHERE artist_id = ?";
	private final String CREATE_NEW_ARTIST_UPDATE = "INSERT INTO artist (artist_name) VALUES (?)";
	private final String DELETE_ARTIST_UPDATE = "DELETE FROM artist WHERE artist_name = ?";
	private final String UPDATE_ARTIST_UPDATE = "UPDATE artist SET artist_name = ? WHERE artist_id = ?";

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

	public List<Artist> getArtists() throws SQLException {
		PreparedStatement ps = connection.prepareStatement (GET_ARTISTS_QUERY);
		ResultSet rs = ps.executeQuery();
		List<Artist> artists = new ArrayList<Artist>();
		while (rs.next()) {
			artists.add(populateArtist(rs.getInt(1),rs.getString(2)));
		}
		return artists;
	}

	public Artist getArtistByName(String artistName) throws SQLException {
		PreparedStatement ps = connection.prepareStatement (GET_ARTIST_BY_NAME_QUERY);
		ps.setString(1,artistName);
		ResultSet rs = ps.executeQuery();
		if (rs.next()) {
			return populateArtist(rs.getInt(1),rs.getString(2));
		} else {
			return null;
		}
	}

	public Artist getArtistById(int artistId) throws SQLException {
		PreparedStatement ps = connection.prepareStatement (GET_ARTIST_BY_ID_QUERY);
		ps.setInt(1,artistId);
		ResultSet rs = ps.executeQuery();
		if (rs.next()) {
			return populateArtist(rs.getInt(1),rs.getString(2));
		} else {
			return null;
```

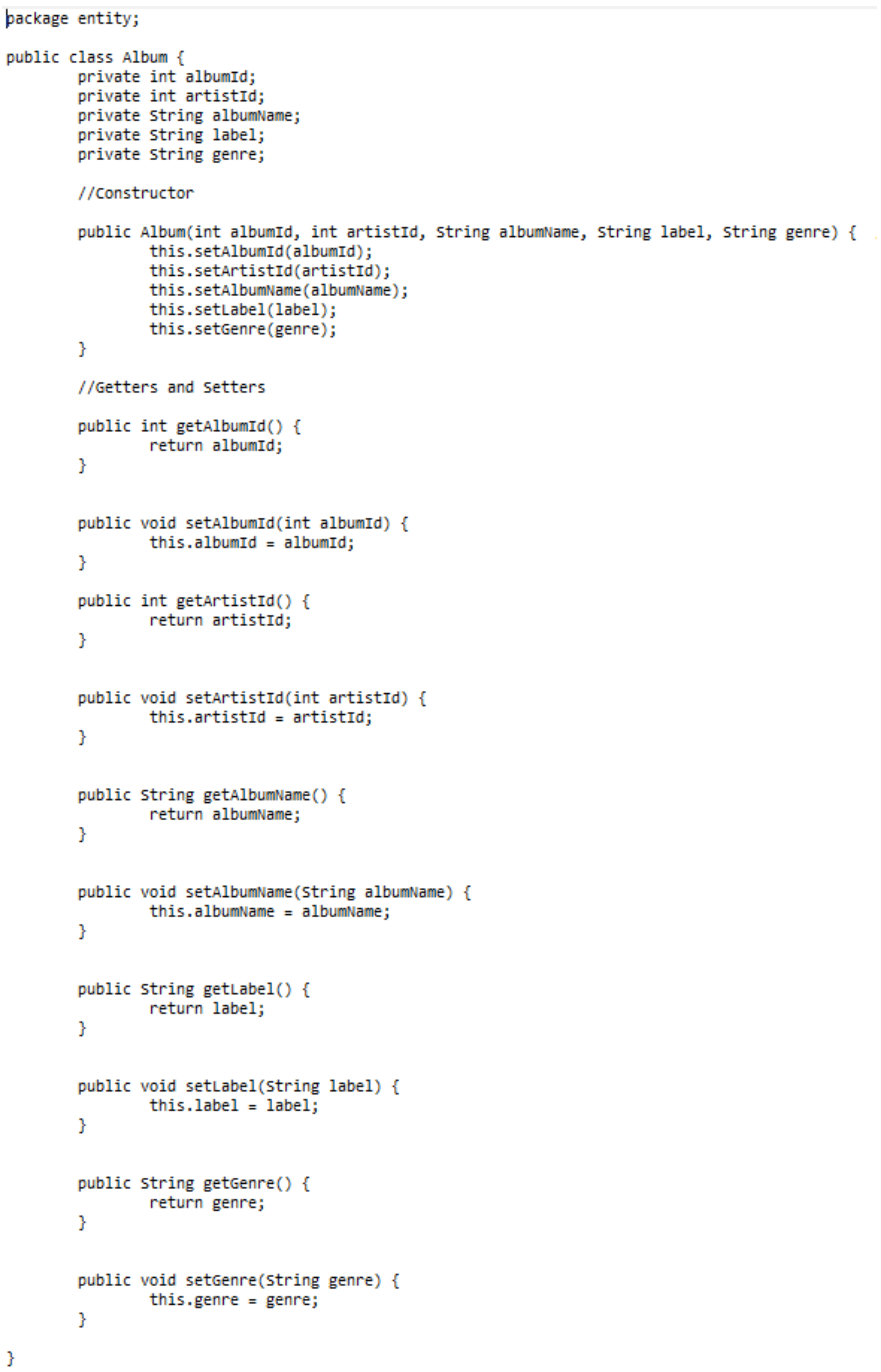
```
        public Artist getArtistById(int artistId) throws SQLException {
                PreparedStatement ps = connection.prepareStatement (GET_ARTIST_BY_ID_QUERY);
                ps.setInt(1,artistId);
                ResultSet rs = ps.executeQuery();
                if (rs.next()) {
                        return populateArtist(rs.getInt(1),rs.getString(2));
                } else {
                        return null;
                }
        }

        public void createNewArtist(String artistName) throws SQLException {
                PreparedStatement ps = connection.prepareStatement (CREATE_NEW_ARTIST_UPDATE);
                ps.setString(1, artistName);
                ps.executeUpdate();
        }

        public void deleteArtist(String artistName) throws SQLException {
                PreparedStatement ps = connection.prepareStatement (DELETE_ARTIST_UPDATE);
                ps.setString(1, artistName);
                ps.executeUpdate();
        }

        public void updateArtist(String artistName, String newName) throws SQLException {
                Artist artist = getArtistByName(artistName);
                PreparedStatement ps = connection.prepareStatement (UPDATE_ARTIST_UPDATE);
                ps.setString(1, newName);


                ps.setInt(2, artist.getArtistId());

                ps.executeUpdate();
        }



        private Artist populateArtist(int artistId, String artistName) {
                return new Artist(artistId,artistName);

        }
}
```

```
package dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

import entity.Certification;

public class CertificationDao {

        private Connection connection;
        private final String GET_CERTS_QUERY = "SELECT * FROM certification";
        private final String CREATE_NEW_CERT_QUERY = "INSERT INTO certification(album_id, cert_status, cert_date) VALUES (?,?,?)";
        private final String UPDATE_CERT_QUERY = "UPDATE certification SET cert_status = ?, cert_date = ? WHERE cert_id = ?";
        private final String DELETE_CERT_QUERY = "DELETE FROM certification WHERE cert_id = ?";

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

        public List<Certification> displayAllCerts() throws SQLException {
                ResultSet rs = connection.prepareStatement(GET_CERTS_QUERY).executeQuery();
                List<Certification> certifications = new ArrayList<Certification>();

                        while (rs.next()) {
                                certifications.add(populateCertifications(rs.getInt(1), rs.getInt(2), rs.getString(3), rs.getString(4)));
                        }
                        return certifications;
                }

        public
        void addNewCert (int albumId, String certStatus, String certDate) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(CREATE_NEW_CERT_QUERY);
                ps.setInt(1,albumId);
                ps.setString(2, certStatus);
                ps.setString(3, certDate);
                ps.executeUpdate();
        }
```

```
        public
        void addNewCert (int albumId, String certStatus, String certDate) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(CREATE_NEW_CERT_QUERY);
                ps.setInt(1,albumId);
                ps.setString(2, certStatus);
                ps.setString(3, certDate);
                ps.executeUpdate();
        }
```

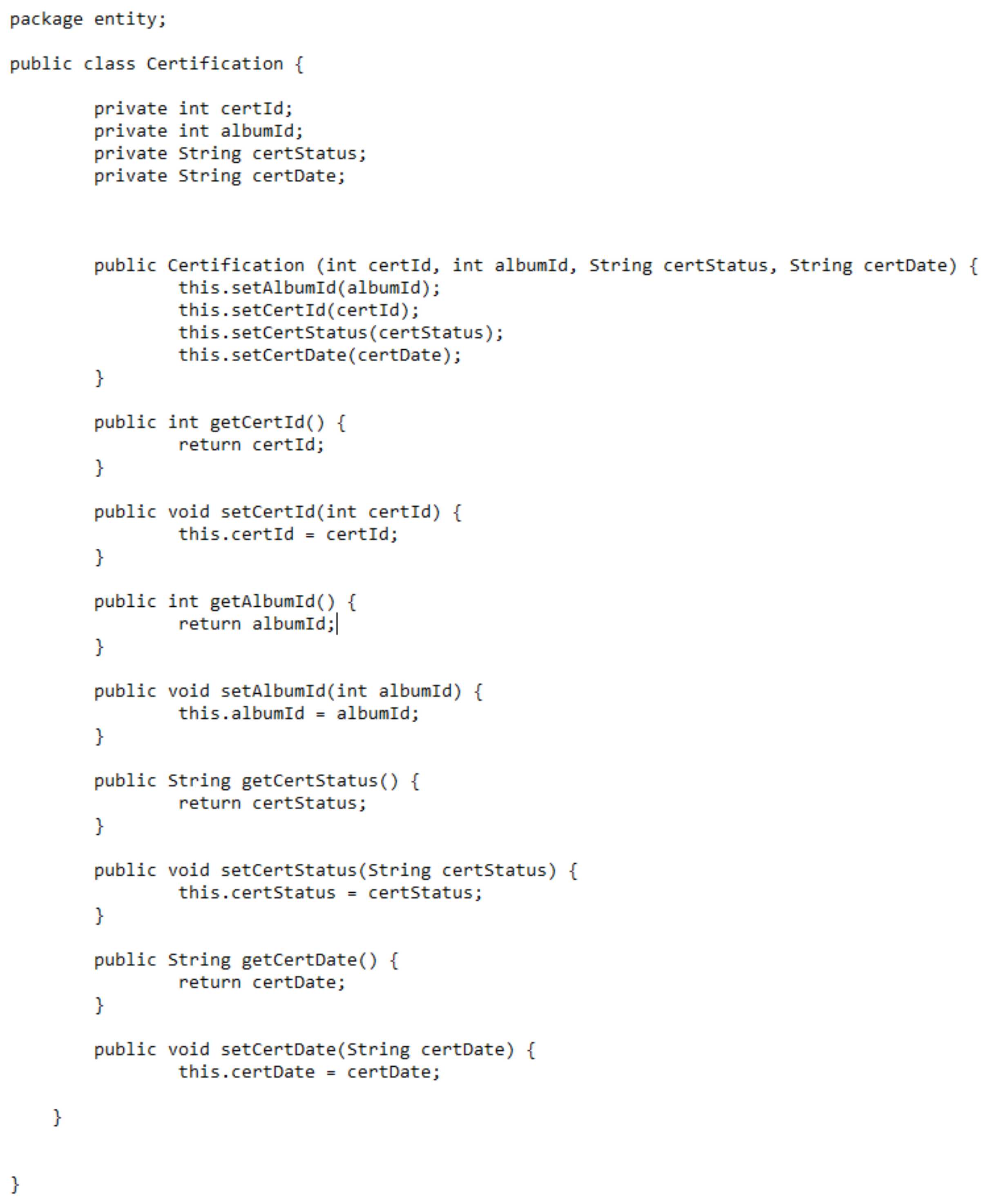
```
        public void updateCert (int certId, String certStatus, String certDate) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(UPDATE_CERT_QUERY);
                ps.setString(1, certStatus);
                ps.setString(2, certDate);
                ps.setInt(3, certId);
                ps.executeUpdate();
        }

        public void deleteCert (int certId) throws SQLException {
                PreparedStatement ps = connection.prepareStatement(DELETE_CERT_QUERY);
                ps.setInt(1, certId);
                ps.executeUpdate();
        }
        private Certification populateCertifications(int certId , int albumId, String certStatus, String certDate) {
                return new Certification (certId, albumId, certStatus, certDate);
        }
}
```

```
/*
 * Promineo Tech BESD Bootcamp
 * MySQL Week 6 Coding Assignment
 * Group Project:  John, Kendall, & Lisa
 *
 * DBConnection Class using JavaSE-1.8
 */

package dao;

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

public class DBConnection {

        private static Connection connection;
        private final static String URL = "jdbc:mysql://localhost:3306/recording_artists";
        private final static String USER = "root";
        private static String password = "";
        private static Scanner scanner = new Scanner(System.in);
        private static DBConnection instance;

        /*
         * Constructor
         */

        private DBConnection (Connection connection) {
                this.connection = connection;
        }


        public static Connection getConnection() {
                boolean success = false;
                do {
                        if (instance == null) {
                                try {
                                        System.out.println("UserName: " + USER);
                                        System.out.print("Enter your password: ");
                                        password = scanner.nextLine();
                                        connection = DriverManager.getConnection(URL,USER,password);
                                        instance = new DBConnection(connection);
                                        System.out.println("\n\n\n\n");
                                        success = true;
                                } catch (SQLException e) {
                                        System.out.println("\n\nERROR:  Please try again!\n\n");
                                }
                    } else {
                        success = true;
                    }
                } while (success == false);

                return DBConnection.connection;
```



```java
package entity;

public class Album {
        private int albumId;
        private int artistId;
        private String albumName;
        private String label;
        private String genre;

        //Constructor

        public Album(int albumId, int artistId, String albumName, String label, String genre) {
                this.setAlbumId(albumId);
                this.setArtistId(artistId);
                this.setAlbumName(albumName);
                this.setLabel(label);
                this.setGenre(genre);
        }

        //Getters and Setters

        public int getAlbumId() {
                return albumId;
        }


        public void setAlbumId(int albumId) {
                this.albumId = albumId;
        }

        public int getArtistId() {
                return artistId;
        }


        public void setArtistId(int artistId) {
                this.artistId = artistId;
        }


        public String getAlbumName() {
                return albumName;
        }


        public void setAlbumName(String albumName) {
                this.albumName = albumName;
        }


        public String getLabel() {
                return label;
        }


        public void setLabel(String label) {
                this.label = label;
        }


        public String getGenre() {
                return genre;
        }


        public void setGenre(String genre) {
                this.genre = genre;
        }

}
```

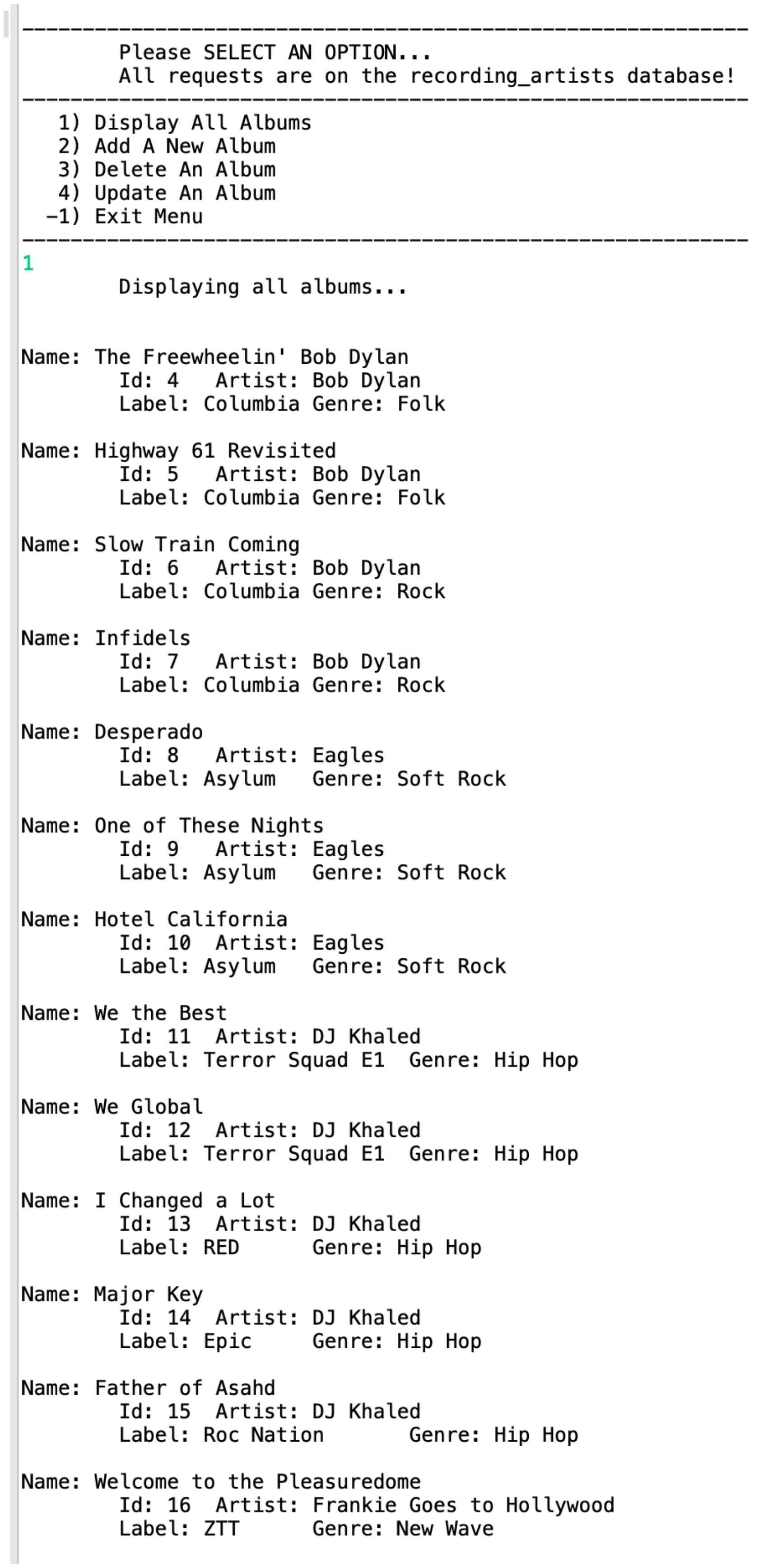
```
package entity;

public class Artist {

        private  int artistId;
        private String artistName;


        /*
         * Constructor
         */

        public Artist (int artistId, String artistName) {
                this.artistId = artistId;
                this.artistName = artistName;

        }

        /*
         * Getters & Setters
         */


        public int getArtistId() {
                return artistId;
        }

        public void setArtistId(int artistId) {
                this.artistId = artistId;
        }

        public String getArtistName() {
                return artistName;
        }

        public void setArtistName(String artistName) {
                this.artistName = artistName;

        }
```



```
package entity;

public class Certification {

        private int certId;
        private int albumId;
        private String certStatus;
        private String certDate;



        public Certification (int certId, int albumId, String certStatus, String certDate) {
                this.setAlbumId(albumId);
                this.setCertId(certId);
                this.setCertStatus(certStatus);
                this.setCertDate(certDate);
        }

        public int getCertId() {
                return certId;
        }

        public void setCertId(int certId) {
                this.certId = certId;
        }

        public int getAlbumId() {
                return albumId;
        }

        public void setAlbumId(int albumId) {
                this.albumId = albumId;
        }

        public String getCertStatus() {
                return certStatus;
        }

        public void setCertStatus(String certStatus) {
                this.certStatus = certStatus;
        }

        public String getCertDate() {
                return certDate;
        }

        public void setCertDate(String certDate) {
                this.certDate = certDate;

    }


}
```

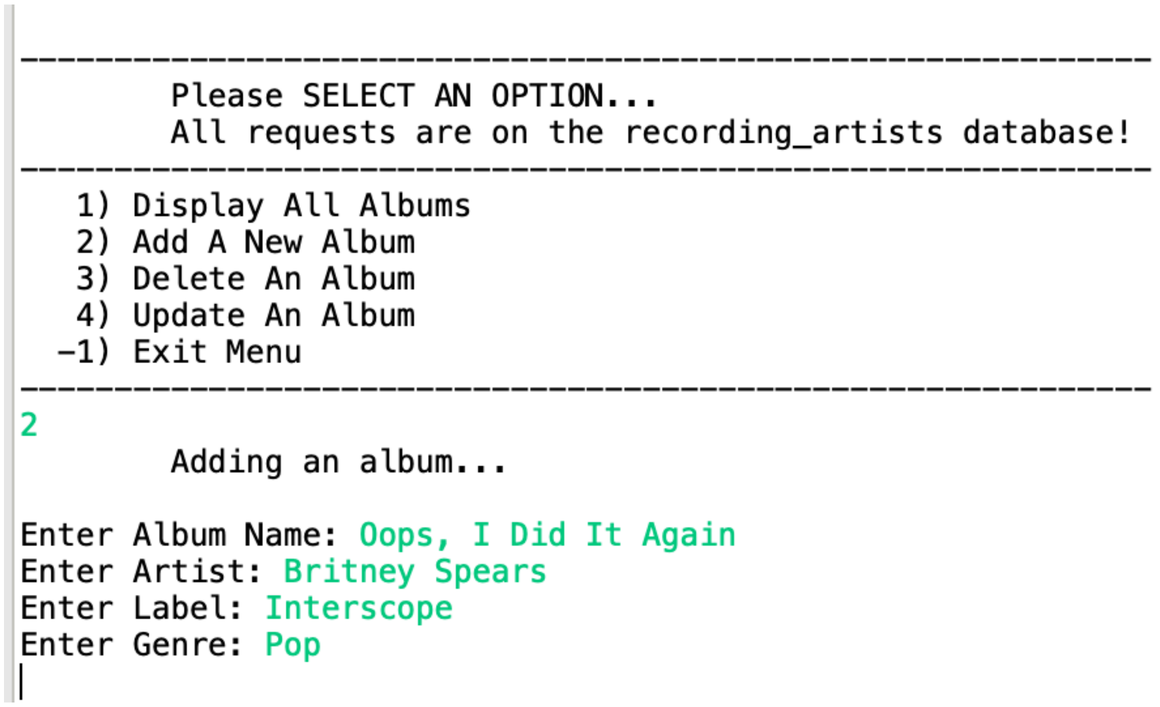
Screenshots of Running Application:

```
---------------------------
MAIN recording_artists MENU
---------------------------

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Album Menu
   2) Artist Menu
   3) Certification Menu
  -1) Exit Menu
------------------------------------------------------------
1
----------------------
ALBUM Information Menu
----------------------

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Albums
   2) Add A New Album
   3) Delete An Album
   4) Update An Album
  -1) Exit Menu
------------------------------------------------------------
```

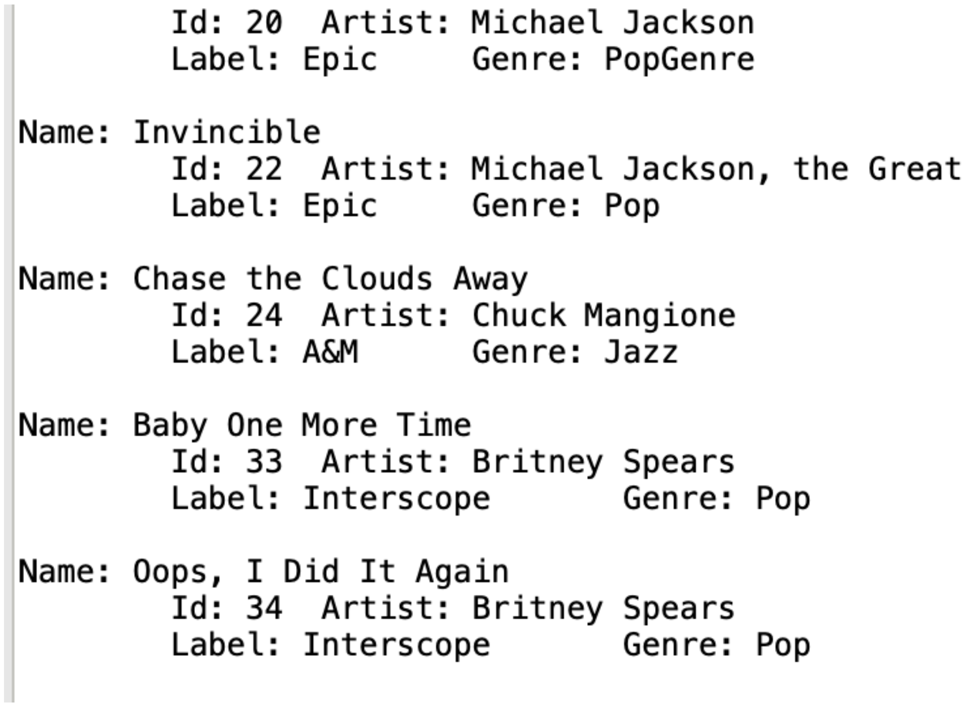


```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Albums
   2) Add A New Album
   3) Delete An Album
   4) Update An Album
  -1) Exit Menu
------------------------------------------------------------
1
        Displaying all albums...


Name: The Freewheelin' Bob Dylan
        Id: 4   Artist: Bob Dylan
        Label: Columbia Genre: Folk

Name: Highway 61 Revisited
        Id: 5   Artist: Bob Dylan
        Label: Columbia Genre: Folk

Name: Slow Train Coming
        Id: 6   Artist: Bob Dylan
        Label: Columbia Genre: Rock

Name: Infidels
        Id: 7   Artist: Bob Dylan
        Label: Columbia Genre: Rock

Name: Desperado
        Id: 8   Artist: Eagles
        Label: Asylum   Genre: Soft Rock

Name: One of These Nights
        Id: 9   Artist: Eagles
        Label: Asylum   Genre: Soft Rock

Name: Hotel California
        Id: 10  Artist: Eagles
        Label: Asylum   Genre: Soft Rock

Name: We the Best
        Id: 11  Artist: DJ Khaled
        Label: Terror Squad E1  Genre: Hip Hop

Name: We Global
        Id: 12  Artist: DJ Khaled
        Label: Terror Squad E1  Genre: Hip Hop

Name: I Changed a Lot
        Id: 13  Artist: DJ Khaled
        Label: RED      Genre: Hip Hop

Name: Major Key
        Id: 14  Artist: DJ Khaled
        Label: Epic     Genre: Hip Hop

Name: Father of Asahd
        Id: 15  Artist: DJ Khaled
        Label: Roc Nation       Genre: Hip Hop

Name: Welcome to the Pleasuredome
        Id: 16  Artist: Frankie Goes to Hollywood
        Label: ZTT      Genre: New Wave
```

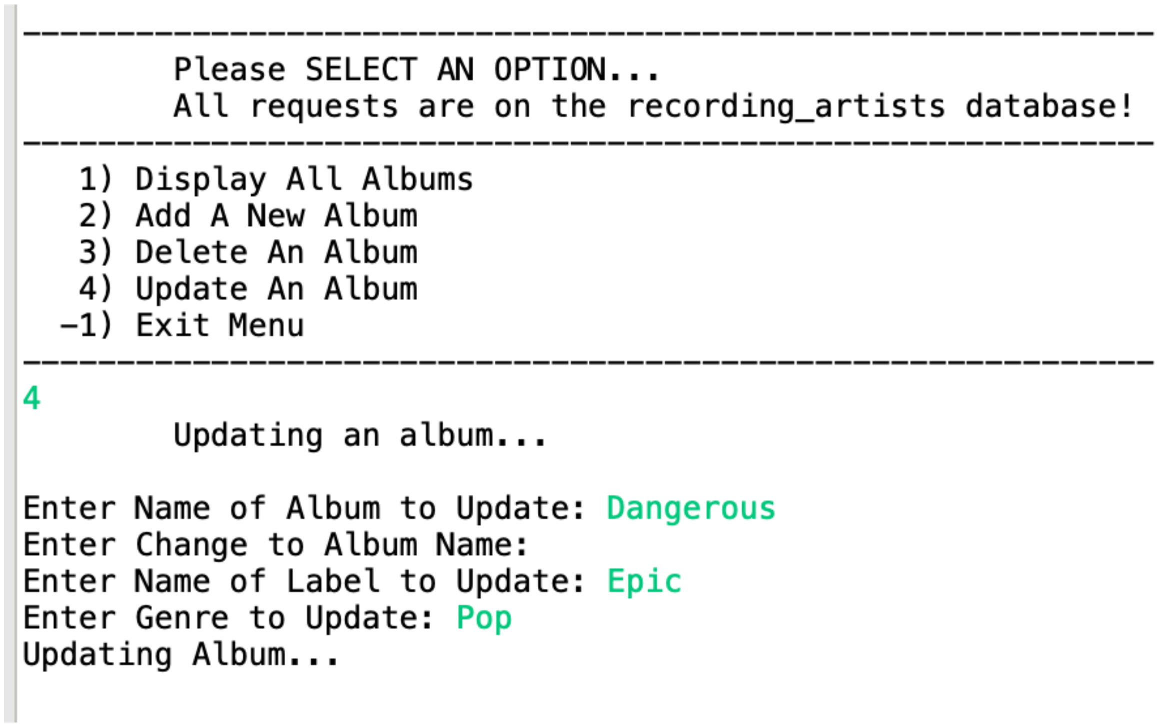
```
Name: Liverpool
        Id: 17  Artist: Frankie Goes to Hollywood
        Label: ZTT      Genre: New Wave

Name: Thriller
        Id: 18  Artist: Michael Jackson
        Label: Epic Records     Genre: Pop

Name: Bad
        Id: 19  Artist: Michael Jackson
        Label:  Genre:

Name: Dangerous
        Id: 20  Artist: Michael Jackson
        Label: Epic     Genre: PopGenre

Name: Invincible
        Id: 22  Artist: Michael Jackson, the Great
        Label: Epic     Genre: Pop

Name: Chase the Clouds Away
        Id: 24  Artist: Chuck Mangione
        Label: A&M      Genre: Jazz

Name: Baby One More Time
        Id: 33  Artist: Britney Spears
        Label: Interscope       Genre: Pop
```



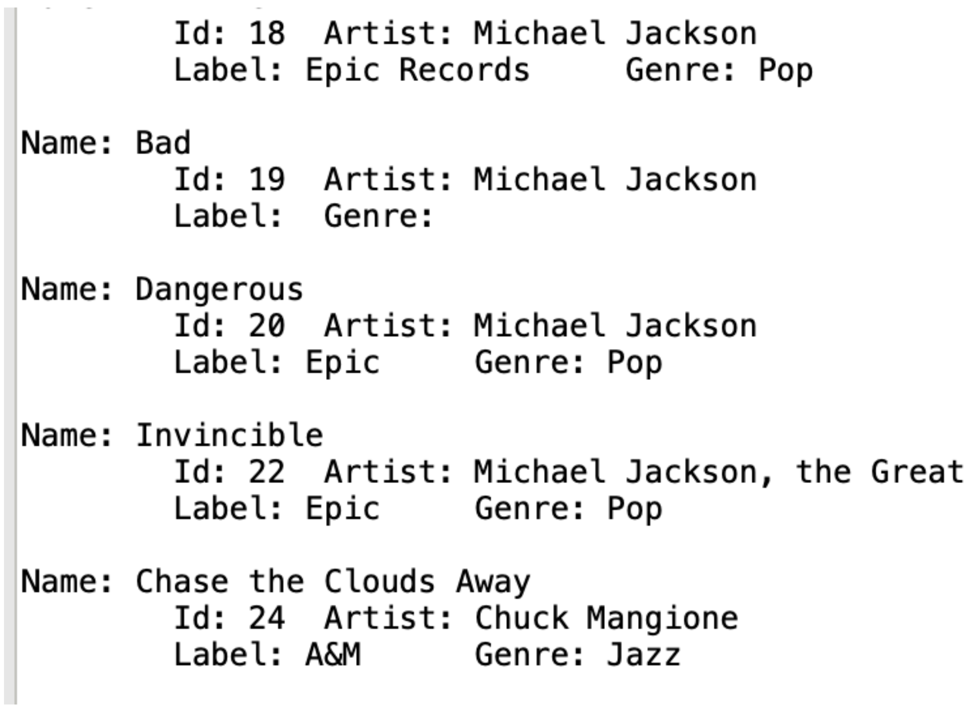
```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Albums
   2) Add A New Album
   3) Delete An Album
   4) Update An Album
  -1) Exit Menu
------------------------------------------------------------
2
        Adding an album...

Enter Album Name: Oops, I Did It Again
Enter Artist: Britney Spears
Enter Label: Interscope
Enter Genre: Pop
```



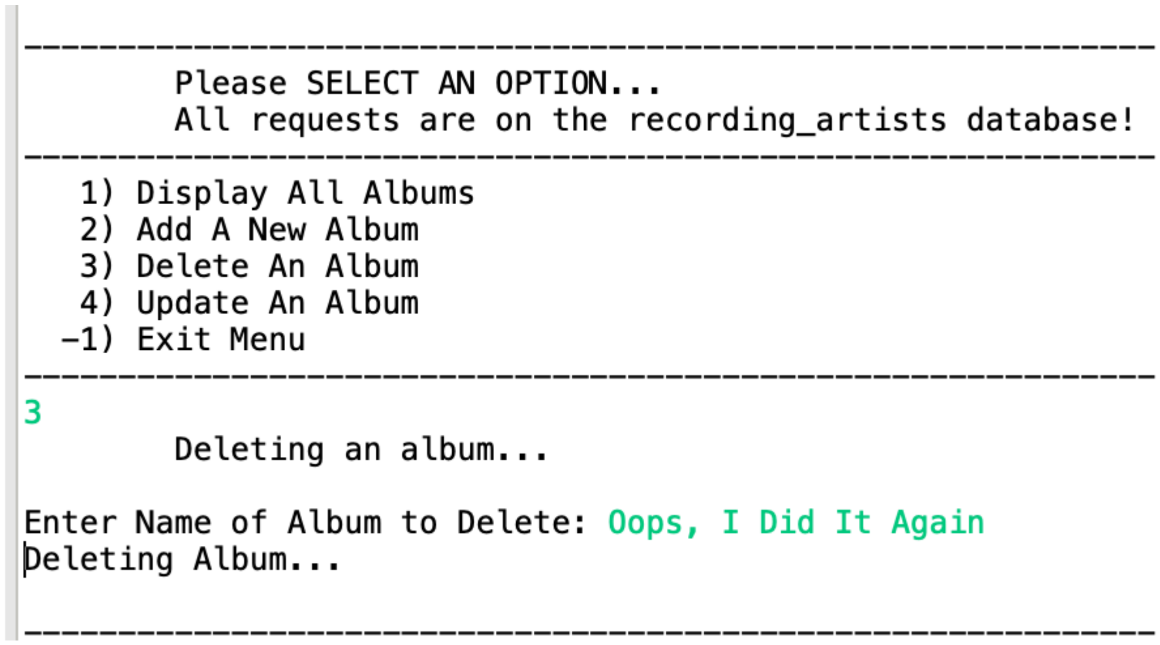
```
        Id: 20  Artist: Michael Jackson
        Label: Epic     Genre: PopGenre

Name: Invincible
        Id: 22  Artist: Michael Jackson, the Great
        Label: Epic     Genre: Pop

Name: Chase the Clouds Away
        Id: 24  Artist: Chuck Mangione
        Label: A&M      Genre: Jazz

Name: Baby One More Time
        Id: 33  Artist: Britney Spears
        Label: Interscope       Genre: Pop

Name: Oops, I Did It Again
        Id: 34  Artist: Britney Spears
        Label: Interscope       Genre: Pop
```



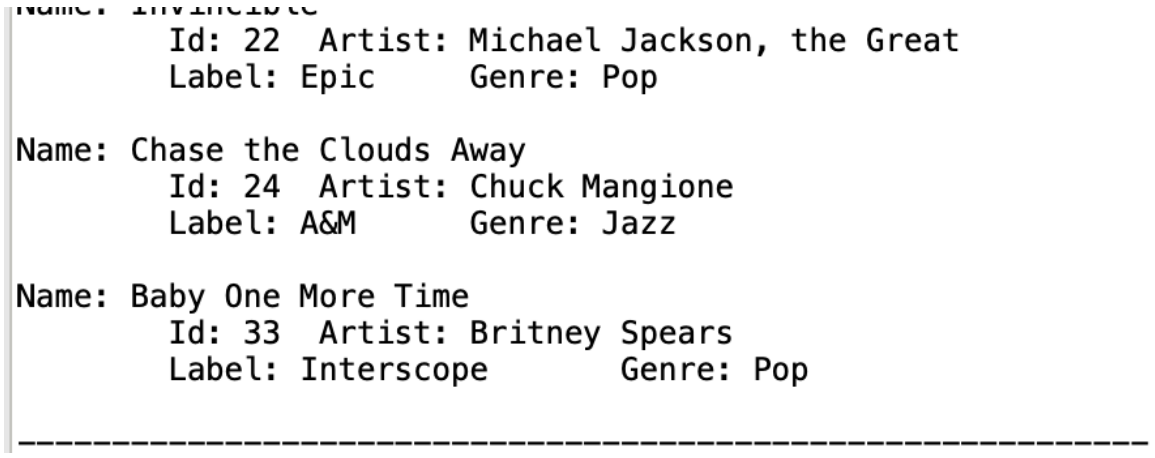
```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Albums
   2) Add A New Album
   3) Delete An Album
   4) Update An Album
  -1) Exit Menu
------------------------------------------------------------
4
        Updating an album...

Enter Name of Album to Update: Dangerous
Enter Change to Album Name: 
Enter Name of Label to Update: Epic
Enter Genre to Update: Pop
Updating Album...
```



```
        Id: 18  Artist: Michael Jackson
        Label: Epic Records     Genre: Pop

Name: Bad
        Id: 19  Artist: Michael Jackson
        Label:  Genre: 

Name: Dangerous
        Id: 20  Artist: Michael Jackson
        Label: Epic     Genre: Pop

Name: Invincible
        Id: 22  Artist: Michael Jackson, the Great
        Label: Epic     Genre: Pop

Name: Chase the Clouds Away
        Id: 24  Artist: Chuck Mangione
        Label: A&M      Genre: Jazz
```

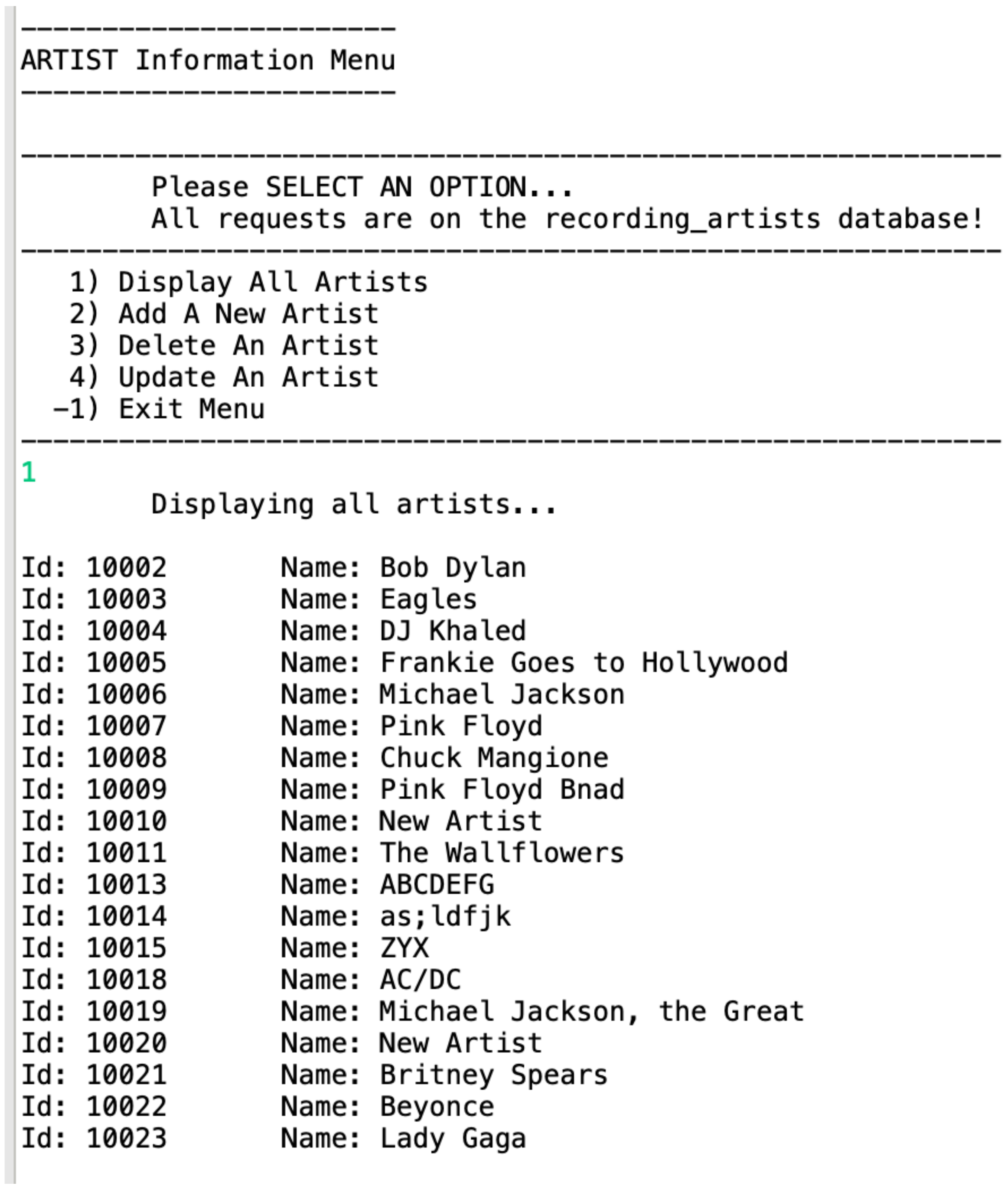


```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Albums
   2) Add A New Album
   3) Delete An Album
   4) Update An Album
  -1) Exit Menu
------------------------------------------------------------
3
        Deleting an album...

Enter Name of Album to Delete: Oops, I Did It Again
Deleting Album...

------------------------------------------------------------
```



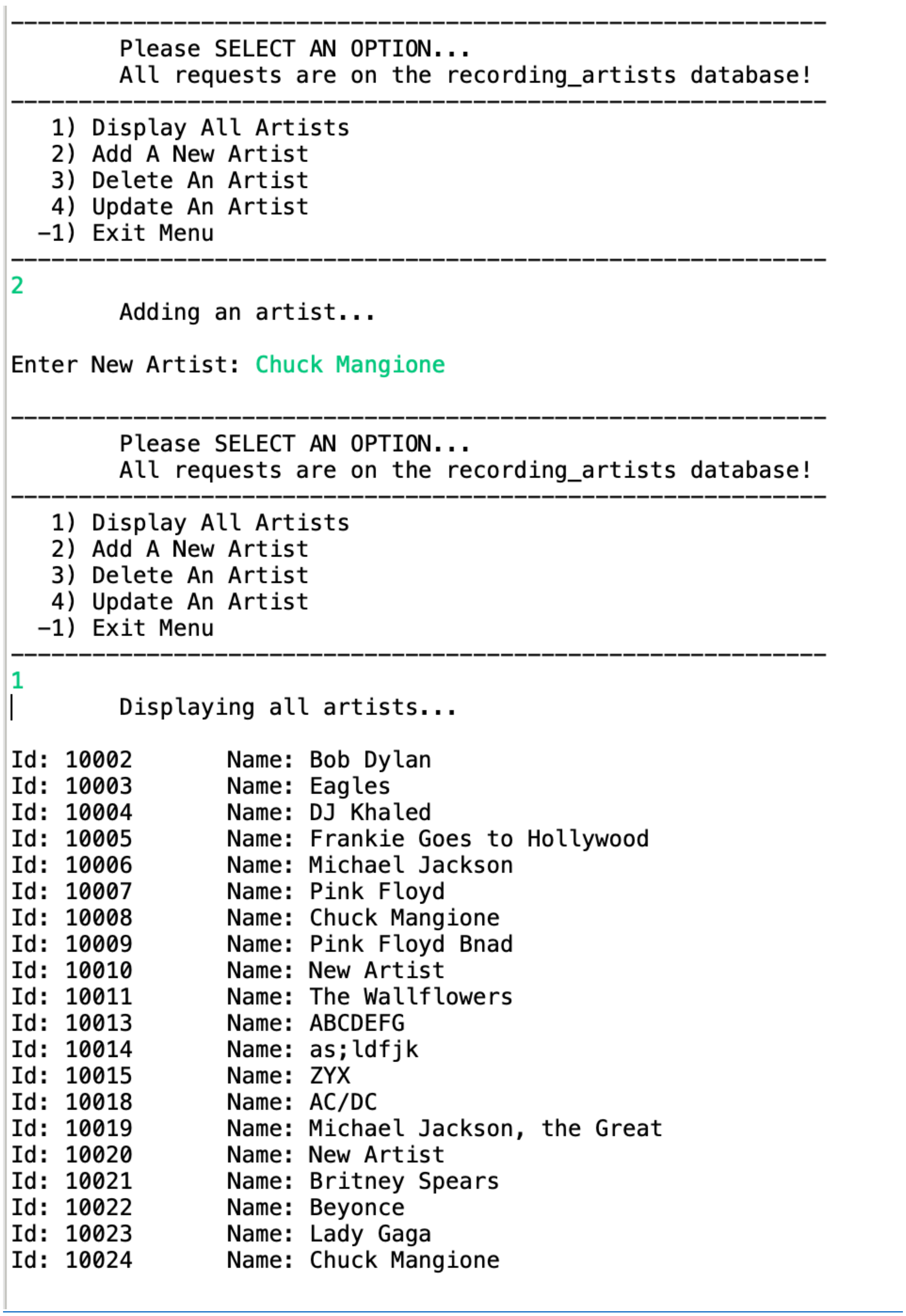
```
Name: Invincible
        Id: 22  Artist: Michael Jackson, the Great
        Label: Epic     Genre: Pop

Name: Chase the Clouds Away
        Id: 24  Artist: Chuck Mangione
        Label: A&M      Genre: Jazz

Name: Baby One More Time
        Id: 33  Artist: Britney Spears
        Label: Interscope       Genre: Pop

------------------------------------------------------------
```

```
---------------------------
MAIN recording_artists MENU
---------------------------


------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Album Menu
   2) Artist Menu
   3) Certification Menu
  -1) Exit Menu
------------------------------------------------------------
2
-----------------------
ARTIST Information Menu
-----------------------


------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Artists
   2) Add A New Artist
   3) Delete An Artist
   4) Update An Artist
  -1) Exit Menu
------------------------------------------------------------
```



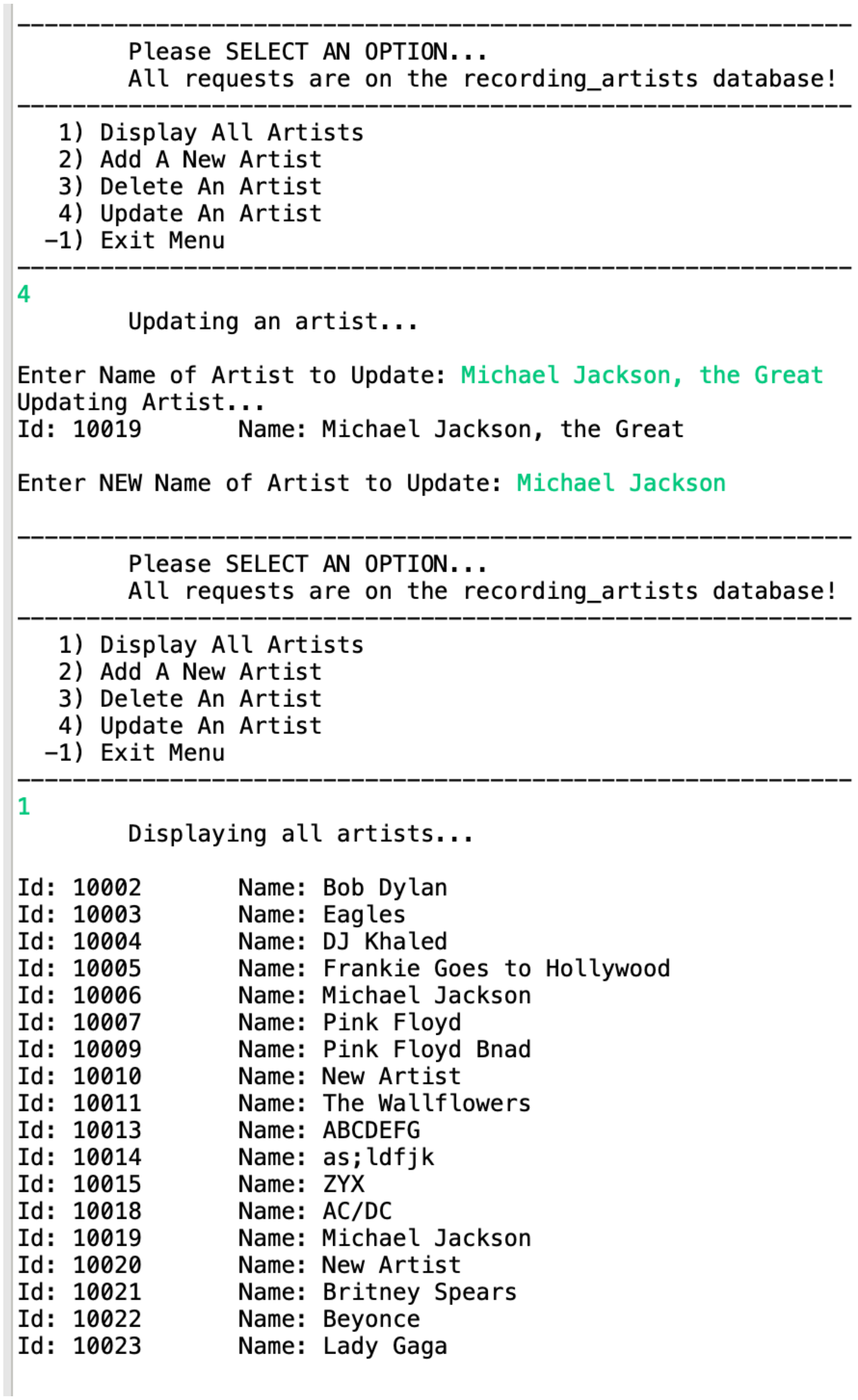
```
-----------------------
ARTIST Information Menu
-----------------------

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Artists
   2) Add A New Artist
   3) Delete An Artist
   4) Update An Artist
  -1) Exit Menu
------------------------------------------------------------
1
        Displaying all artists...

Id: 10002       Name: Bob Dylan
Id: 10003       Name: Eagles
Id: 10004       Name: DJ Khaled
Id: 10005       Name: Frankie Goes to Hollywood
Id: 10006       Name: Michael Jackson
Id: 10007       Name: Pink Floyd
Id: 10008       Name: Chuck Mangione
Id: 10009       Name: Pink Floyd Bnad
Id: 10010       Name: New Artist
Id: 10011       Name: The Wallflowers
Id: 10013       Name: ABCDEFG
Id: 10014       Name: as;ldfjk
Id: 10015       Name: ZYX
Id: 10018       Name: AC/DC
Id: 10019       Name: Michael Jackson, the Great
Id: 10020       Name: New Artist
Id: 10021       Name: Britney Spears
Id: 10022       Name: Beyonce
Id: 10023       Name: Lady Gaga
```

URL to GitHub Repository: **https://github.com/kendallcodes/JavaSQLWeek6Final.git**



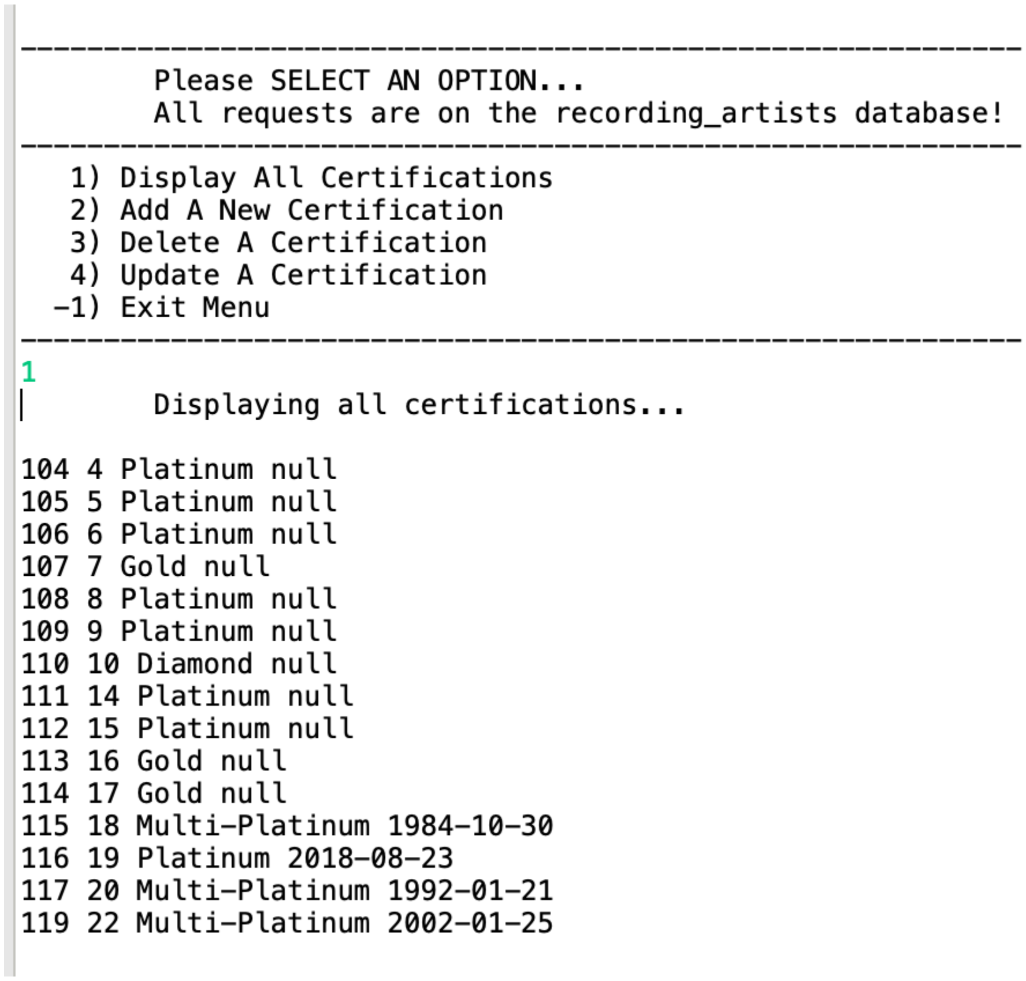
```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Artists
   2) Add A New Artist
   3) Delete An Artist
   4) Update An Artist
  -1) Exit Menu
------------------------------------------------------------
2
        Adding an artist...

Enter New Artist: Chuck Mangione

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Artists
   2) Add A New Artist
   3) Delete An Artist
   4) Update An Artist
  -1) Exit Menu
------------------------------------------------------------
1
        Displaying all artists...

Id: 10002       Name: Bob Dylan
Id: 10003       Name: Eagles
Id: 10004       Name: DJ Khaled
Id: 10005       Name: Frankie Goes to Hollywood
Id: 10006       Name: Michael Jackson
Id: 10007       Name: Pink Floyd
Id: 10008       Name: Chuck Mangione
Id: 10009       Name: Pink Floyd Bnad
Id: 10010       Name: New Artist
Id: 10011       Name: The Wallflowers
Id: 10013       Name: ABCDEFG
Id: 10014       Name: as;ldfjk
Id: 10015       Name: ZYX
Id: 10018       Name: AC/DC
Id: 10019       Name: Michael Jackson, the Great
Id: 10020       Name: New Artist
Id: 10021       Name: Britney Spears
Id: 10022       Name: Beyonce
Id: 10023       Name: Lady Gaga
Id: 10024       Name: Chuck Mangione
```

```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Artists
   2) Add A New Artist
   3) Delete An Artist
   4) Update An Artist
  -1) Exit Menu
------------------------------------------------------------
3
        Deleting an artist...

Enter Name of Artist to Delete: Chuck Mangione
Deleting Artist...
Id: 10008       Name: Chuck Mangione
        **  This will also delete all of the associated data in the recording_artists database!  **

Would you like to proceed, yes or no?  yes

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Artists
   2) Add A New Artist
   3) Delete An Artist
   4) Update An Artist
  -1) Exit Menu
------------------------------------------------------------
1
        Displaying all artists...

Id: 10002       Name: Bob Dylan
Id: 10003       Name: Eagles
Id: 10004       Name: DJ Khaled
Id: 10005       Name: Frankie Goes to Hollywood
Id: 10006       Name: Michael Jackson
Id: 10007       Name: Pink Floyd
Id: 10009       Name: Pink Floyd Bnad
Id: 10010       Name: New Artist
Id: 10011       Name: The Wallflowers
Id: 10013       Name: ABCDEFG
Id: 10014       Name: as;ldfjk
Id: 10015       Name: ZYX
Id: 10018       Name: AC/DC
Id: 10019       Name: Michael Jackson, the Great
Id: 10020       Name: New Artist
Id: 10021       Name: Britney Spears
Id: 10022       Name: Beyonce
Id: 10023       Name: Lady Gaga
```

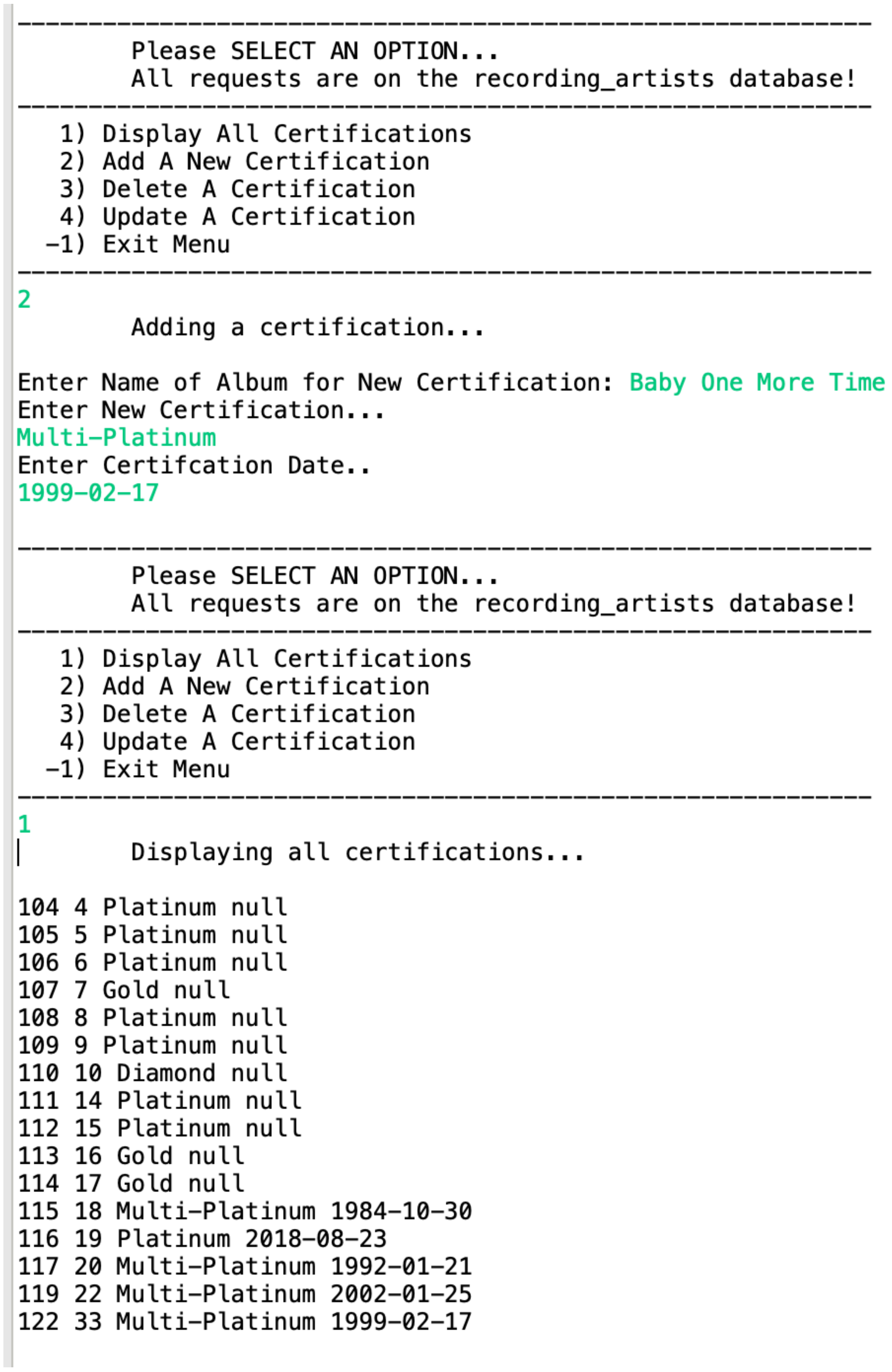


```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Artists
   2) Add A New Artist
   3) Delete An Artist
   4) Update An Artist
  -1) Exit Menu
------------------------------------------------------------
4
        Updating an artist...

Enter Name of Artist to Update: Michael Jackson, the Great
Updating Artist...
Id: 10019       Name: Michael Jackson, the Great

Enter NEW Name of Artist to Update: Michael Jackson

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Artists
   2) Add A New Artist
   3) Delete An Artist
   4) Update An Artist
  -1) Exit Menu
------------------------------------------------------------
1
        Displaying all artists...

Id: 10002       Name: Bob Dylan
Id: 10003       Name: Eagles
Id: 10004       Name: DJ Khaled
Id: 10005       Name: Frankie Goes to Hollywood
Id: 10006       Name: Michael Jackson
Id: 10007       Name: Pink Floyd
Id: 10009       Name: Pink Floyd Bnad
Id: 10010       Name: New Artist
Id: 10011       Name: The Wallflowers
Id: 10013       Name: ABCDEFG
Id: 10014       Name: as;ldfjk
Id: 10015       Name: ZYX
Id: 10018       Name: AC/DC
Id: 10019       Name: Michael Jackson
Id: 10020       Name: New Artist
Id: 10021       Name: Britney Spears
Id: 10022       Name: Beyonce
Id: 10023       Name: Lady Gaga
```

```
---------------------------
MAIN recording_artists MENU
---------------------------


------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Album Menu
   2) Artist Menu
   3) Certification Menu
  -1) Exit Menu
------------------------------------------------------------
3
------------------------------
CERTIFICATION Information Menu
------------------------------


------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Certifications
   2) Add A New Certification
   3) Delete A Certification
   4) Update A Certification
  -1) Exit Menu
------------------------------------------------------------
```



```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Certifications
   2) Add A New Certification
   3) Delete A Certification
   4) Update A Certification
  -1) Exit Menu
------------------------------------------------------------
1
        Displaying all certifications...

104 4 Platinum null
105 5 Platinum null
106 6 Platinum null
107 7 Gold null
108 8 Platinum null
109 9 Platinum null
110 10 Diamond null
111 14 Platinum null
112 15 Platinum null
113 16 Gold null
114 17 Gold null
115 18 Multi-Platinum 1984-10-30
116 19 Platinum 2018-08-23
117 20 Multi-Platinum 1992-01-21
119 22 Multi-Platinum 2002-01-25
```



```
------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Certifications
   2) Add A New Certification
   3) Delete A Certification
   4) Update A Certification
  -1) Exit Menu
------------------------------------------------------------
2
        Adding a certification...

Enter Name of Album for New Certification: Baby One More Time
Enter New Certification...
Multi-Platinum
Enter Certifcation Date..
1999-02-17

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Certifications
   2) Add A New Certification
   3) Delete A Certification
   4) Update A Certification
  -1) Exit Menu
------------------------------------------------------------
1
        Displaying all certifications...

104 4 Platinum null
105 5 Platinum null
106 6 Platinum null
107 7 Gold null
108 8 Platinum null
109 9 Platinum null
110 10 Diamond null
111 14 Platinum null
112 15 Platinum null
113 16 Gold null
114 17 Gold null
115 18 Multi-Platinum 1984-10-30
116 19 Platinum 2018-08-23
117 20 Multi-Platinum 1992-01-21
119 22 Multi-Platinum 2002-01-25
122 33 Multi-Platinum 1999-02-17
```

```
119 22 Multi-Platinum 2002-01-25
122 33 Platinum 1999-02-10

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Certifications
   2) Add A New Certification
   3) Delete A Certification
   4) Update A Certification
  -1) Exit Menu
------------------------------------------------------------
3
        Deleting a certification...

Enter Certification Id to delete....
122

------------------------------------------------------------
        Please SELECT AN OPTION...
        All requests are on the recording_artists database!
------------------------------------------------------------
   1) Display All Certifications
   2) Add A New Certification
   3) Delete A Certification
   4) Update A Certification
  -1) Exit Menu
------------------------------------------------------------
1
        Displaying all certifications...

104 4 Platinum null
105 5 Platinum null
106 6 Platinum null
107 7 Gold null
108 8 Platinum null
109 9 Platinum null
110 10 Diamond null
111 14 Platinum null
112 15 Platinum null
113 16 Gold null
114 17 Gold null
115 18 Multi-Platinum 1984-10-30
116 19 Platinum 2018-08-23
117 20 Multi-Platinum 1992-01-21
119 22 Multi-Platinum 2002-01-25
```