Project 6: Database Programming with JDBC

Objectives
Understand how a common high-level language database API works.
Learn how to write a simple Java program that uses JDBC to interact with a SQLite database.

Overview
Create a simple Java program that allows the user to

- display all the records in a database
- insert a new record in the database
- update a record in the database
- delete a record in the database

Here is a simple example interaction:

```
Welcome to the Albums Database! Enter a command to continue

1.Display all albums
2.Insert a new album
3.Update an existing album
4.Delete an album

>
```

If the user types 1, the program should display a list of album data from the database. Do not spend a lot of time formatting the output (messy is ok). After displaying the data, the program may then terminate.

If the user types 2, the program should then prompt the user for an album title, year and rank.

```
New Album
Title: Foobars and Sunsets
Year: 1976
Rank: 123

Album saved.
```

The program may then terminate.

You do not have to validate the Rank values before/during insertion. In other words, it is ok for there to be multiple albums with the same rank in the database. *Later, we will reflect on the additional logic and/or constraints necessary to handle such cases.*

If the user types 3, the program should then prompt the user for an album id, display it, and prompt the user for the new attribute values. For example:

```
Update Album
ID: 101
Foobars and Sunsets (1976) Ranked 123
New Title: Barfoos and Sunrises
New Year: 1976
New Rank: 123

Album updated.
```

Your program may then terminate.

You do not have to validate Rank or Title values before/during updating. *Later, we will reflect on the additional logic and/or constraints necessary to handle such cases.*

If the user types 4, the program should then prompt the user for an album id and delete the record from the database. For example:

```
Delete Album
ID: 101
Album deleted.
```

Your program may then terminate.

You do not have to ensure that a record with the provided ID exists in the database before deleting. *Later, we will reflect on the additional logic and/or constraints necessary to handle such cases.*

Instructions

Read SQLite chapter 8, Language Extensions, pages 236 - 243 “Java.”

In your environment of choice, ensure that the Java JDK (not just the JRE!) are installed and that both `java` and `javac` are on your path. *Ask on Piazza immediately if you do not know how to do either of these things.*

Remember, you can use the linux virtual machine (link on Piazza Course Page), campus computers, or your own option. Want to try an IDE? Use IntelliJ IDEA12.

Download and extract this zip file. Next, compile and run the existing Main.java.

```
javac Main.java
java -classpath sqlite-jdbc-3.7.2.jar:. Main
```

You should see that the program displays the album data on the screen.

Now, modify the code such that it meets the requirements described above.

Tip: Your code should involve the use of `java.sql.PreparedStatement`.
Tip: Define and use well-named functions!

Presentation

Submit your main code listing as an attachment on Blackboard.

Good writing is important, **especially when it comes to code**. *Writing with poor readability, lack of appropriate whitespace, misspellings, etc will be handed back for re-writing (no grade penalty).*

Submit a zip of your code via blackboard. I would prefer a zip containing only one source file (.java) rather than a bunch of project assets and the like.

Grading Criteria (90 points)

Complete implementation of four operations and the main menu (60 total points).
- writing quality (30)
- implementation (60)

Please submit your work via Blackboard by 7AM on the due date.