CSCE 156 – Computer Science II

Lab 6.0 - SQL+JDBC I

Sarah Roscoe*

Summer 2019

Prior to Lab

1. Review this laboratory handout prior to lab.
2. Create a MySQL account by logging in here: https://cse-apps.unl.edu/amu/amu/login
3. Change the account password using the following directions at: http://cse.unl.edu/faq-section/unix-linux#node-302
4. Review the supplemental SQL Cheat Sheet for the Album Database
5. This is a long lab but it can be completed if you prepare properly. Review the following materials:
   - Information About Databases and Tables
     http://dev.mysql.com/doc/refman/5.6/en/getting-information.html
   - Connecting to MySQL from the command line:
     http://dev.mysql.com/doc/refman/5.6/en/connecting-disconnecting.html
   - Retrieving data
     http://www.w3schools.com/sql/sql_select.asp
   - Conditional clause
     http://www.w3schools.com/sql/sql_where.asp
   - Inserting data
     http://www.w3schools.com/sql/sql_insert.asp

*Developed by Dr. Chris Bourke. Revamped by Yi Xia

- Deleting data
  http://www.w3schools.com/sql/sql_delete.asp
- Updating data
  http://www.w3schools.com/sql/sql_update.asp
- `count()`
  http://www.w3schools.com/sql/sql_func_count.asp
- `max()`
  http://www.w3schools.com/sql/sql_func_max.asp
- `min()`
  http://www.w3schools.com/sql/sql_func_min.asp
- Joining tables inner join
  http://www.w3schools.com/sql/sql_join_inner.asp
- left join
  http://www.w3schools.com/sql/sql_join_left.asp
- right join
  http://www.w3schools.com/sql/sql_join_right.asp

Lab Objectives & Topics

Following the lab, you should be able to:

- Connect to a database and execute queries
- Perform basic Create, retrieve, update, and delete (CRUD) operations
- Understand more complex queries using Joins and Aggregate functions

Pre Lab Activity

Material for this lab is available for download on the course website.

1 Querying a Database

You will be connecting to a remote MySQL database server on CSE and executing several queries. The queries you will be performing involve a database that contains data about various music albums, songs and the artists involved. The database structure is illustrated in the ER (Entity-Relation) diagram in Figure 1.

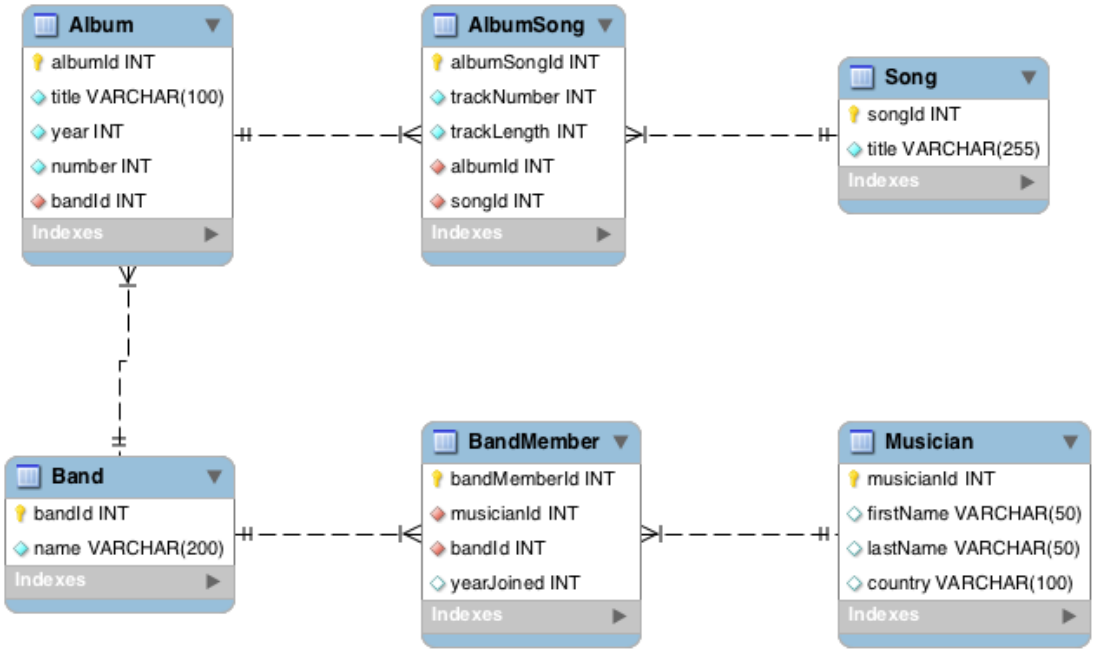

Figure 1: Albums Database

1.1 Importing the Database

You will need to "install" the Albums database and data into your own database on CSE. Note: database in this sense is just a collection of related tables; on CSE you only have access to one actual database–the database named after your CSE login. To import the Albums database, you can either a) simply run the `albums.sql` script (you may need to add a `use cselogin;` first) in MySQL Workbench; or b) from the command line:

1. Make sure the `albums.sql` DDL file (Data Description Language) is on your Z drive.
2. From the command line (via PuTTY), execute the following:

   ```
   mysql -u username -p username < albums.sql
   ```

   where `username` is replaced with your CSE login. Enter your MySQL password. This redirects the contents of the `albums.sql` file (a collection of SQL commands) to the mysql command line interface, creating all the tables and inserting all the data necessary.

1.2 Executing Queries

You may use any interface to your MySQL database that you wish, but we recommend that you use MySQL Workbench. You can download and install it from here: https://www.mysql.com/products/workbench/. Otherwise, it is available on the CSE lab computers.

1. Launch MySQL Workbench
2. From the quick launch menu select "Open Connection to Start Querying"
3. Enter the host name (cse.unl.edu), username (your cse login) and enter your sql password; click "OK"
4. You can now enter queries and execute them (follow the menu options)

Execute the queries in the worksheet and demonstrate them to a lab instructor. Instead of writing the answers by hand, you may simply type them in the worksheet provided.

1.3 SQL Supplemental Cheat Sheet

For your benefit, we have created a supplemental SQL cheat sheet that you may reference. It contains many of the major types of queries along with a practical application using the Album database.

2 Database Design

Refer back to Figure 1, a careful examination of the DDL file provided (`albums.sql`) indicates how these tables were built and related to each other. New requirements may mean that the underlying data model must be modified to support new pieces of data. For example, if we wanted to keep track of the emails of each Musician we could modify the Musicians table to include an email address. SQL allows us to *alter* tables:

```
ALTER TABLE Musicians ADD EmailAddress VARCHAR(50);
```

A better solution would add support for multiple emails in which case we would need to add an entirely new table.

```
CREATE TABLE email (
    EmailID INT NOT NULL AUTO_INCREMENT,
    MusicianID INT NOT NULL,
    Address VARCHAR(100) NOT NULL,
    PRIMARY KEY(email_id),
    FOREIGN KEY `fk_email_to_musician` (MusicianID) REFERENCES
    Musician(MusicianID)
);
```

In this lab, you will build on the Albums database to add support for modeling venues (concert halls) at which bands are under contract to play select album songs. You will add tables and keys to this database to support this functionality.

2.1 Solution Outline

In this activity you will design entities and relation(s) to extend the Albums database such that it supports the following concert information:

1. The band playing at the concert
2. The band's select album songs played at the concert
3. The date the concert was held (use an appropriately formatted varchar, date time types are not cross-compatible)
4. The name of the hall where the concert was held
5. The number of seats in the concert hall
6. The number of concert tickets sold

The entities and relations illustrated in Figure 2 serves as a basis for the solution. You will add fields and another entity by following the first section of the worksheet.

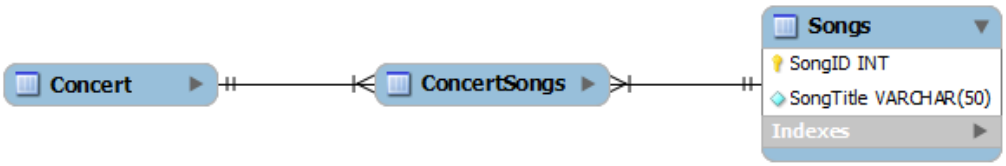

Figure 2: Albums Database

The Songs table is part of the original Albums database. Relationships between two entities are indicated by a line between the two entities and in general are either a one-to-one relationship or a one-to-many relationship. Figure 3 shows the standard way to draw the two types of relations.

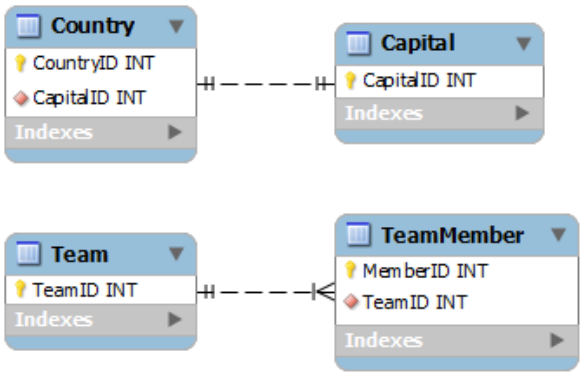

Figure 3: Albums Database

Complete the corresponding section of the worksheet.

2.2 SQL Script Modification

Now that you have properly designed the database modifications, you will realize them by writing SQL scripts (or modifying the original DDL file, `albums.sql`). Follow the directions in the worksheet in order to add the entities and relations you have designed in the previous activity to the tables and relations in the original *Albums* database.

Lab Activity

3 JDBC in a Web Application

In this lab you will familiarize yourself with the Java Database Connectivity API (JDBC) by finishing a simple, nearly complete retrieve-and-display web application and deploying it to an application server (Glassfish). The design of the webapp is simple: it consists of an index page that loads album data via Ajax (Asynchronous JavaScript and XML) and displays it in a table.

It is not necessary to understand the details of the application (the HTML, JavaScript, Servlets, or application server). The main goal of this lab is to give you some familiarity with JDBC and exposure to a multi-tiered application and web application server environment.

4 Getting Started

Note: Unless you are familiar with Java Applications Servers and have one installed on your laptop, you will need to use the lab computers for this lab. In addition, it would be a good idea to reset your Albums database by rerunning the SQL script from a prior lab.

For this lab, you will need to use the JEE (Java Enterprise Edition) version of Eclipse, not the JSE (Java Standard Edition). In Windows, click the start menu and enter "Eclipse", the "Java EE Eclipse" should show up, select this version. You may use the same workspace as with the JSE (Java Standard Edition) version of Eclipse.

Alternatively (and recommended) you can use the Eclipse version in the Linux partition, which will enable you to deploy locally to your lab machine instead of the csce server. To use this version of Eclipse:

1. Restart your lab computer and choose openSUSE. Login with your usual CSE credentials.
2. Once logged into Linux, go to "Applications" → "Search" and open up a "Terminal"
3. Launch eclipse by executing

   ```
   /usr/local/bin/eclipse &
   ```

   (the ampersand launches Eclipse in the background so you can still use the terminal session)

Once you've got Eclipse running, clone the project code for this lab from GitHub using

the URL, https://github.com/yi-xia/Lab06-SQL_JDBC.git. Refer to Lab 01 for instructions on how to clone a project from GitHub.

5 Modifying Your Application

1. You will first need to make changes to the `unl.cse.albums.DatabaseInfo` source file. In particular, change the login and password information to your MySQL credentials. You can reset these by going to http://cse.unl.edu/check.
2. The HTML, JavaScript, etc. has been provided for you. Feel free to make modifications these files, but you should know what you are doing as changes can break functionality in other parts of the application.
3. The application will not display any album data until you have completed the methods in the `Album` class.
   - `public static List<Album> getAlbumSummaries()` – This method will query the database and get a complete list of all albums in the database. It will create and populate `Album` objects and put them in a list which will then be returned. This method will be used to generate the album table, so it doesn't need all information, just a subset (see the documentation as to what is required). You should optimize your queries to only select the relevant columns.
   - `public static Album getDetailedAlbum(int albumId)` – this method will query the database for the specific album with the given primary key and return an `Album` instance with *all* relevant data (band and its members, songs, etc.) specified.
   - Important: do not forget to close your database resources (especially connections) after you are finished using them.
   - A `Test` class has been provided for you to test your `getAlbumSummaries()` method which you can also adapt to test your `getDetailedAlbum()` method. You should use it to debug your methods before deploying your application.

6 Deploying Your Application

You have two options for deploying your application as a Web Archive (WAR) file to an application server.

Linux: Deploying Locally

If you chose to do this lab in the Linux version, then follow these instructions.

1. Right click your project and select "Export...", select "Web", "WAR file", "Next"
2. Click Browse and select a directory (your home directory is recommended) and file to export to; name the file `loginLab06.war` where `login` is replaced by your cse login.
3. Return to your terminal and copy the WAR file to glassfish's autodeploy directory:

   ```
   cp loginLab06.war ~glassfish/glassfish/domains/domain1/autodeploy
   ```

4. Open a web browser ("Applications" → "Firefox") and go to the following url: http://localhost:8080/loginLab06 where login is replaced with your CSE login.

Windows: Deploying to CSCE

If you chose to do this lab in the Windows version, then follow these instructions.

1. Export your application in a Web Archive File (WAR): right click your project and choose "Export" → "Export..." → "Web" → "WAR file"
2. Click "Browse" and select a directory to export it to, name the file `loginLab06.war` where `login` is replaced by your cse login. The file should now be in the directory you selected.
3. Copy the WAR file to the root of your CSE directory (copy it to your Z: drive)
4. SSH (PuTTY) into `csce.unl.edu` (*not* `cse.unl.edu`) using your cse login
5. Copy the war file to the server's glassfish auto deploy directory with the command

   ```
   cp loginLab06.war ~glassfish/glassfish/domains/domain1/autodeploy/
   ```

6. Make sure that the file has proper permissions by executing the following from the command line:

   ```
   chmod 755 ~glassfish/glassfish/domains/domain1/autodeploy/loginLab06.war
   ```

   where, `login` is replaced by your cse login. If necessary, you may need to give glassfish a kick in the butt by executing:

   ```
   touch ~glassfish/glassfish/domains/domain1/autodeploy/loginLab06.war
   ```

7. Open a web browser and go to the following URL: http://csce.unl.edu:8080/loginLab06/ where `login` is replaced by your CSE login. Unless there were problems with your code, the webapp should now work, you can click on album

titles or bands to be taken to another page that gives further details.

7 Completing Your Lab

Complete the worksheet and have your lab instructor sign off on it. Before you leave, make sure you "undeploy" your web app by deleting the WAR file from the glassfish autodeploy directory:

```
rm ~glassfish/glassfish/domains/domain1/autodeploy/loginLab06.war
```