Advanced Programming

Lab Sheet 17 – Databases & JDBC

Introduction

In one of this week's lectures, we took a look at Java Database Connectivity (JDBC); the packages we use when we wish to connect to a database server to insert/update/delete data, or to run queries on it. JDBC abstracts over the exact database engine being used, with virtually identical code being used to connect to and query many common database engines including mySQL/mariaDB, Oracle, MSSQL, PostgresSQL, Amazon RedShift, Apache Derby and SQLite.

In this lab, we'll be using the SQLite database engine, given that its quick and simple, and that it stores its databases as simple files. It also gives you a chance to gain some familiarity with a commonly-used RDBMS that you may not yet have encountered in your studies.

Database Setup

You are going to need a database to work on. The easiest way to create an SQLite database is with one of the specialised GUI tools. There are several tools that you could use for this, but I recommend DB Browser for SQLite (available here). If you don't have it on your computer you can download the program as a simple zip file, without needing to install it. It is also available for Linux or even Mac.

Create the table structure depicted in Fig. 1, below. You should select appropriate data types for each column. Each of the ID fields can be set to auto increment, to save you from having to specify a unique ID for each entry.

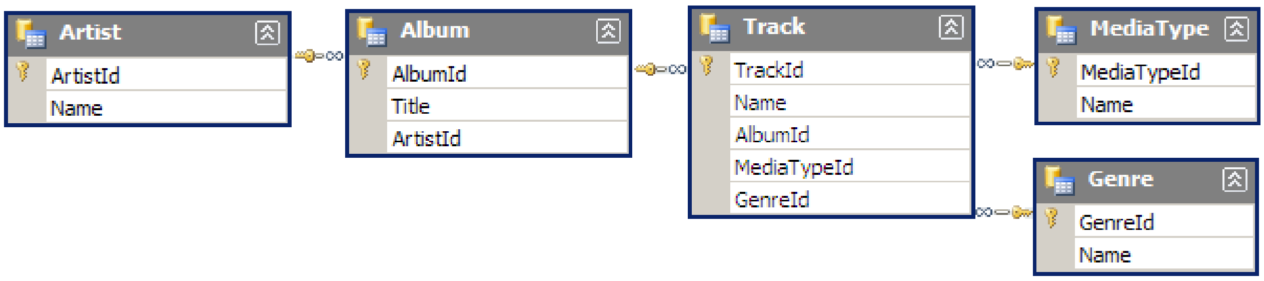

Figure 1: Digital Music Download Store Database Diagram

You will also need to add a few rows to each of the tables, so that your eventual programmes will produce output that you'll be able to check as correct. Add some music from artists you enjoy!

Eclipse Project Setup

Once you have created your database, you should create an eclipse project and drag-and-drop the database file (it typically has a .db extension) into the project folder, much as you did with the XML and JSON files in last week's lab.

Additionally, you will need a copy of the SQLite JDBC driver in order to be able to access the database using your Java code. You can download a copy of this jar file here. Once downloaded, this too should be dragged-and-dropped into your eclipse project. You will also need to add the jar to your build path, by right clicking on it, selecting "Build Path", and then "Add to Build Path".

Tasks

There is no need to use a separate section for each of this week's tasks: they'll be quite short because once you have written the code for one, it can easily be adapted to the others. It is very common for the code for different database queries to be substantially similar, so don't be concerned! This also means that it will be fairly straightforward to adapt the code from the lecture to these tasks.

1. Write a program that prints out the name of each artist in the database

2. Write a program that prints out how many albums there are (use an SQL COUNT)

3. Write a program that prints out all of the tracks in the database by a specific artist

4. As an extension task, you could try writing a program that prints out a table showing how many tracks each artist in the database has. This is probably more of a test of your skill with SQL than with Java programming, however.