DBI Lab 004: SQL CREATE, INSERT & SELECT

COMP1048 - Databases and Interfaces

Matthew Pike & Yuan Yao

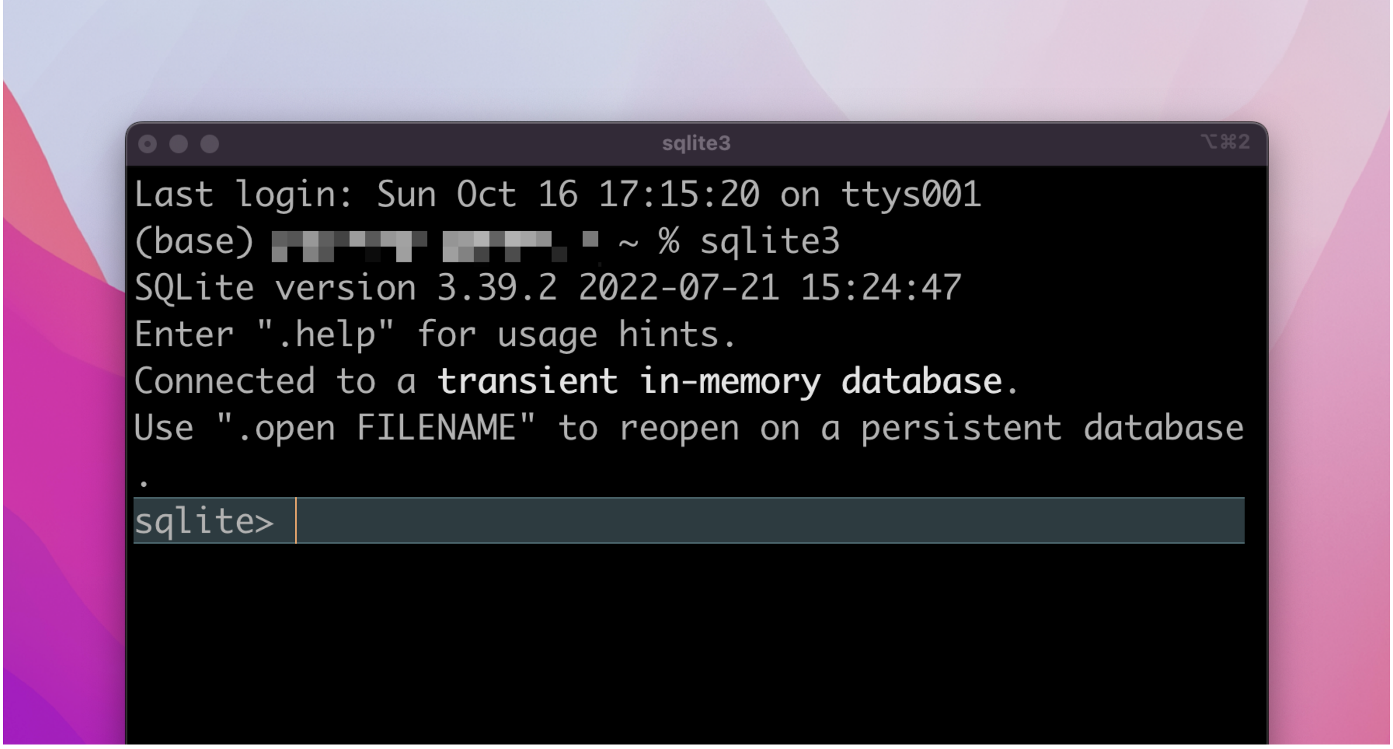
Lab Overview

In Lab 1, we prepared SQLite on your personal laptop, making sure it was working properly. In this lab, we will be using SQLite to complete the activities below.

If you are using Windows, you can start SQLite by double-clicking the `sqlite3.exe` file, which should show a Command-Line Interface (CLI).

If you are using a Mac or Linux computer, you will need to open a Terminal and enter the `sqlite3` command followed by the "Enter" key on your keyboard. If, you're on Mac, press `CMD` + `Spacebar` to open Spotlight, then type `Terminal` followed by “Enter”.

The result should look like this:



```
Last login: Sun Oct 16 17:15:20 on ttys001
(base) [redacted] ~ % sqlite3
SQLite version 3.39.2 2022-07-21 15:24:47
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database
.
sqlite>
```

We can state where we want our database to be stored when opening the database:

```
sqlite3 FilmHouse.db
```

This will create a file name `FilmHouse.db` in the directory where you ran the `sqlite3` command. All future commands run during this SQLite session will take effect on the data stored in `FilmHouse.db`.

> **Note**: SQLite does not require us to use any particular file extension. For this course, we will use `.db` for all SQLite databases.

Alternatively, if you are already running the `sqlite3` executable, you can use the `.open` dot command built into `sqlite3`:

```
.open MyDBFile.db
```

For a complete list of the available dot commands in `sqlite3`, use the `.help` command. A selection of relevant dot commands are shown below (Note: This is not an complete list of commands).

```
sqlite> .help

.cd DIRECTORY            Change the working directory to DIRECTORY
.databases               List names and files of attached databases
.dump ?OBJECTS?          Render database content as SQL
.excel                   Display the output of next command in spreadsheet
.headers on|off          Turn display of headers on or off
.help ?-all? ?PATTERN?   Show help text for PATTERN
.import FILE TABLE       Import data from FILE into TABLE
.mode MODE ?OPTIONS?     Set output mode
.open ?OPTIONS? ?FILE?   Close existing database and reopen FILE
.output ?FILE?           Send output to FILE or stdout if FILE is omitted
.quit                    Exit this program
.read FILE               Read input from FILE or command output
.save ?OPTIONS? FILE     Write database to FILE (an alias for .backup ...)
.schema ?PATTERN?        Show the CREATE statements matching PATTERN
.separator COL ?ROW?     Change the column and row separators
.show                    Show the current values for various settings
.tables ?TABLE?          List names of tables matching LIKE pattern TABLE
```

Exercise 1: `CREATE`

Given the follow problem description, write the `SQL` code necessary to represent the tables needed to store this data.

> A film house wants to create a database to store the details of its collections. Information to be stored about each movie includes their price, title, year and genre. Each movie will have a leading actor, and each leading actor may appear in several movies. Actors have names associated with them, and it should be possible to search the database with the actor's name.

Exercise 2: `INSERT`

Using the database tables you made in Exercise 1, write the SQL needed to `INSERT` the following data in your database tables:

title	price	year	genre	leading actor
Die Hard with a Vengeance	12.24	1999	Action	Barry Nelson
Black Snake Moan	9.99	2007	Adventure	Barry Nelson
Snakes on a Plane	9.99	2011	Comedy	Arethan Franklin
Freeway of Love	9.99	2018	Drama	Bullet Prakash
I knew you were waiting for me	12.25	1997	Comedy	Daniel Craig
The Black Panther	10.99	2018	Action	James Bond
The Jungle Book	9.99	2015	Adventure	Jonny Walker
Infinity War	8.5	1975	Horror	Laura Dern
Coming to Europe	12.99	2001	Adventure	Laura Dern
The Midnight	10.99	2019	Drama	Ryan Reynolds

Exercise 3: `SELECT`

Using the database tables you made in Exercise 2, write the SQL needed to `SELECT` the rows from your database tables that meet the following requirements:

1. List the `title` and `year` of all movies in the database
2. List the `title` of movies that have a `price` greater than 10
3. List the names of all leading actors.
4. List the `title` of movies that have a price less than or equal to 9.99 and are in the "Adventure" `genre`
5. List the `title` and `genre` of all movies in the `Action` or `Comedy` genres.
6. List all lead actors. Duplicates should be removed, and results presented in descending order.

Submission

Please submit a PDF document containing your `SQL` solutions to the above tasks. **You do not need to include the output of your queries.**

Submitting this assignment will contribute 1% to your overall Module grade.

Your submission should demonstrate reasonable effort and fulfil the specified requirements set out in this lab sheet to receive the available marks. There is no granularity to the marking, the marking is on a pass-or-fail basis.

The submission point is available on Moodle.

Submission Deadline - Monday, 31 October 2022 @ 15:00