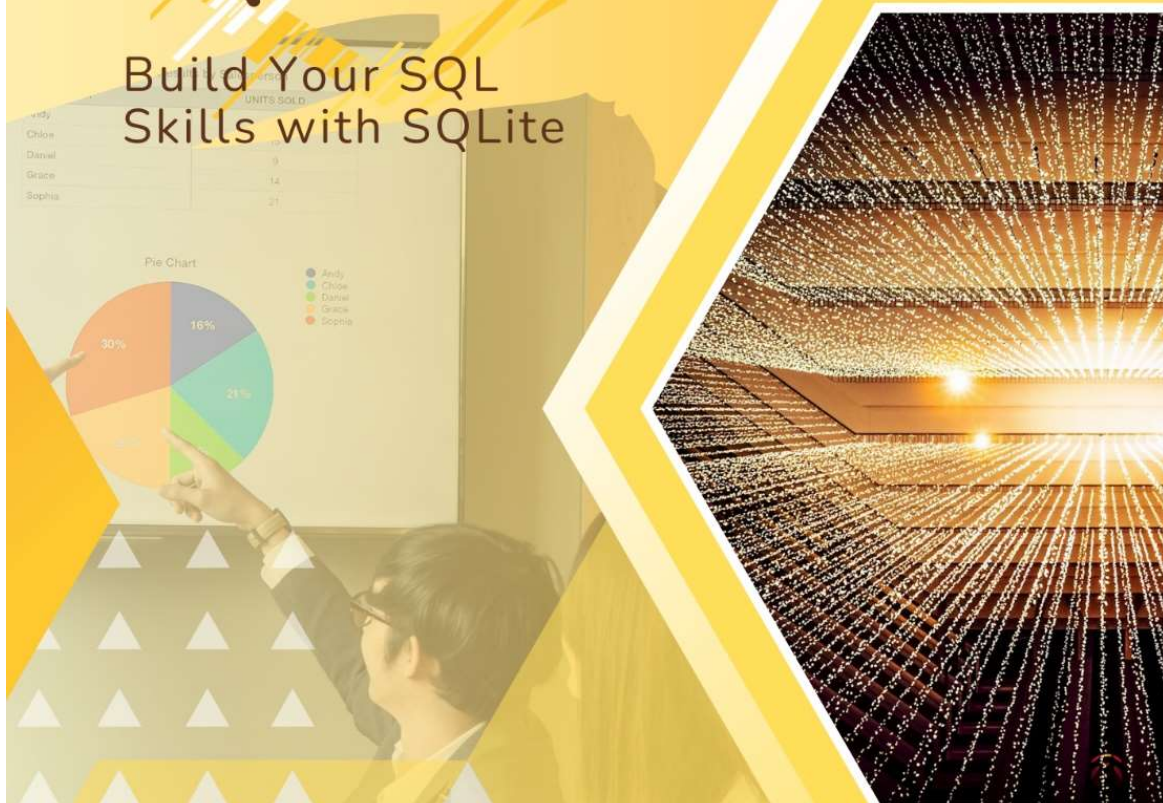# SELF-GUIDED SQL

Build Your SQL Skills with SQLite

Develop your skills and master the technology through hands-on lessons.

Andrew Comeau
Comeau Software Solutions

Self-Guided
SQL

Build Your SQL Skills with SQLite

Andrew Comeau

Comeau Software Solutions
Ocala, Florida

Sample Edition
Updated January 2025

This is a free sample of the upcoming book, "Self-Guided SQL".

For Moose –
You were loved.

Contents

Introduction

If you've picked up this book, you probably already know that you need to learn Structured Query Language (SQL) and don't need me to spend a lot of time on the reasons why. With data and databases a part of everyday work and life, being able to retrieve and analyze that data will be a valuable skill for many years to come and it's a great addition to your skillset. The real question is how this particular book can help you.

Some books present you with a lot of theory before setting you loose to work and, by the time you get to the exercises, they feel like an afterthought. This book does the opposite. In this book, you learn SQL by *doing*. Each lesson is made up of steps you will take to actually see SQL and database operations in action, with enough explanation in between so you'll understand what you are doing. Every lesson is designed to build on the last.

This is not a book that you can passively read through over a cup of coffee; it's a hands-on workshop that will encourage you to use and experiment with the skills you're learning at all times. Beyond simply teaching you SQL and SQLite, my intent is to show you how to actively explore a new area of technology. Hopefully, by the time you finish here, you will have a firm grasp on SQL *and* be better prepared to approach the next subject you learn, whatever that might be.

Having taught SQL in the classroom and being self-taught, I know there are three things that are essential to mastering SQL or any other subject - desire, focus and curiosity. The last one is especially important and makes the difference between someone who just passes tests and someone who excels and whose talent will eventually be noticed and sought out by others. Curiosity takes a person beyond any one book or resource and motivates them to spend time on that all-important experimentation and discovery.

In every lesson of this book, I will encourage you to ask questions and try things for yourself, even if ... especially if ... they result in error messages. They won't hurt you and you're going to see them at some

point so you might as well find out what causes them.

Using SQLite

I've worked with a variety of database software from Microsoft Access to MySQL. Every software has its place in the scheme of things, its advantages and its disadvantages. For this book, I chose SQLite as the software to teach for a few reasons:

- SQLite is in the public domain; it's completely free to use. The database browser program that you'll be using is also free and open source. You don't have to pay a cent for either.

- Neither of these programs require any installation beyond copying some files to a directory. SQLite itself is a single DLL file. You won't clutter up your computer with hidden files or registry changes. You can even run them from a flash drive if you want. It also works on both Windows and Linux.

- Despite the last two items, SQLite is actually a very popular database format that's used for standalone applications on smartphones, PCs and other devices. If you're going to learn about databases, it's a good one to know.

Additional Resources

Throughout this book, I'll be pointing you to additional resources such as websites and videos that will provide additional information on SQL and SQLite to help you in your mastery of these subjects. SQL is a large and interesting language, there is a world of information out there and exploring it will help you maximize your SQL skills.

Researching a subject on your own is also an integral part of self-learning and it's a good idea to have a library of resources on hand. The advantage of *this* book is that I've evaluated the resources so that I can filter out the noise and recommend the best ones that will give you the most value for your time.

A Note on Formatting

As I said earlier, this book centers around the actions you need to take to use and learn SQL. These actions are presented in numbered steps for emphasis and should be your first focus when moving through the book. Each chapter should be read as a list of instructions to be completed with some notes on theory here and there.

Because of this, many chapters have a *single* numbered list of steps that runs through the chapter, rather than restarting the numbering after section headings, etc.. This might make it appear that lists are starting in the middle but they are actually *continuing*. I also believe this will help readers maintain their position within a chapter and reference specific actions more easily.

I also don't show a lot of query results throughout the book, except where I feel it's necessary to demonstrate the performance of the SQL. Readers are strongly encouraged to try each one of the queries shown and view the results for themselves.

Statement on A.I. Use

The content in this book was not written by A.I.. It is the result of my research into SQL and SQLite and the words are my own.

I do use A.I. as an assistant. At times, during the writing process, I have consulted with ChatGPT and other tools for answers on specific technical questions as it's often easier than going down the rabbit hole of modern search engine results.

I have asked it to suggest SQL exercises based on the sample databases shown in this book as, for me, that's the most time consuming part of writing a book like this. I have personally tested and verified all SQL code shown here and, in most cases, I've written it myself or heavily modified it from that suggested by the A.I. tools.

I have also used tools such as Google's NotebookLM as part of the editing process to verify coverage of concepts, fact-check, check readability, etc..

I am including this statement because I believe that quality content created by humans is inherently more valuable than that generated by A.I. There is no shortcut to true quality or substitute for the willingness of an author to associate their work with their name and reputation. There are also serious issues of intellectual property, copyright and reliability in the training of A.I. tools. Authors and readers both have a responsibility to demand responsibility and transparency in the use of artificial intelligence.

Feedback and Updates

I'm very interested in hearing whether this book was of help to you and what you got out of it. You can contact me at books@comeausoftware.com .

Also, while I've done everything possible to avoid any errors in the book, it's always possible you might find a typo here or there. Please let me know if you do. For minor corrections and announcements of updates to the book, please check the official page on my website at ComeauSoftware.com.

https://www.comeausoftware.com/self-guided-sql-sqlite/

Andrew Comeau
books@comeausoftware.com
January 2025

Section 1 – Installation and Setup

1.1 - Installation

For this book, you'll be installing the SQLite utilities which include a text-only console program along with the **DB Browser for SQLite** graphical database browser which will enable you to see a lot more about the databases at a glance rather than typing in a lot of commands. I believe it's important to be familiar with both so that you're not dependent on one or the other and can be productive no matter which tool is available.

In Windows, these programs are available with installation routines and with no-install versions for which the files can simply be copied into a folder and run manually. I will be focusing on the no-install option for simplicity.

In Linux, it's best to install both through the Terminal program and let Linux handle the rest. I'll provide the instructions for this in the section on Linux.

Windows

If you would like a video guide to the Windows installation you can find a link to the demo video I made on this book's official page at:

https://www.comeausoftware.com/self-guided-sql-sqlite/

I recommend creating a new directory on your computer to hold all the files that you'll be downloading. You could even create it on your Windows desktop if that's easiest for you. I'd suggest the following names with 'SQLite' being the main directory and the others as subdirectories:
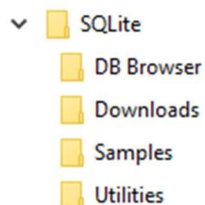
1. In your web browser, go to the site **https://sqlite.com**. Take a moment to read the short introduction "What is SQLite" and the About page to get some more facts about the software. Then click on the **Download** link at the top of the page.

2. Scroll down to the section titled "Precompiled Binaries for Windows".

3. At this point, you're probably using a 64-bit version of Windows but there is a 32-bit version if needed. The filename is based on the most recent version so it will change by the time you read this book.

4. Download either the 32-bit or 64-bit "sqlite-dll-winx ..." file containing the DLL and the "sqlite-tools-win..." files and save them to the **SQLite\Downloads** folder you created earlier.

You can download SQLite for 32-bit and 64-bit systems
along with command line tools from the official site at SQLite.com.

5. Both files are in ZIP format so they should open easily in any modern version of Windows. Extract the contents from both of the files to the **SQLite\Utilities** folder you created earlier.

That's it for the installation of the SQLite utilities. You should have five files in your Utilities folder.

*   **sqlite3.dll** – Main SQLite software library

*   **sqlite3.def** – Support file

*   **sqldiff.exe** – Program for finding differences between databases.

*   **sqlite3.exe** – Console program for querying and managing databases.

*   **sqlite3_analyzer.exe** – Analysis tool for database files.

Installing the graphical database browser is just as easy.

1. In your web browser, go to the site **https://sqlitebrowser.org/** and click on the **Download** link at the top of the page.

2. Download either the 32-bit or 64-bit version of the software depending on your system. Again, I'm recommending the version labeled "no installer". This is also a ZIP file.

3. Extract the ZIP file to the **SQLite\DB Browser** folder you created earlier.

4. In the extracted file, find the file **DB Browser for SQLite.exe**. This is the program file that you will run to start the software. You should probably create a shortcut to this file somewhere on your Windows Desktop or wherever else you can easily get to it.

That's all for the Windows installation. You now have SQLite on your system along with the utilities and a browser program. In the next lesson, we'll find some sample databases to work with.

For both programs, you can create shortcuts on your desktop or you can add the directories to your system PATH statement.

1. On the Windows taskbar search box, type "system environment". The best match will be a selection called **Edit the System Environment Variables**.

2. On the **System Properties** screen that appears, select **Environment Variables** at the bottom.

3. Under the **System Variables** section, double-click the **Path** setting. This will open the list of directories contained in the Path statement.

4. Click the **New** button and enter the path to the folder that contains the SQLite files you downloaded (i.e. C:\SQLite) and press **Enter**.

5. Click **OK** and close the System Properties panel.
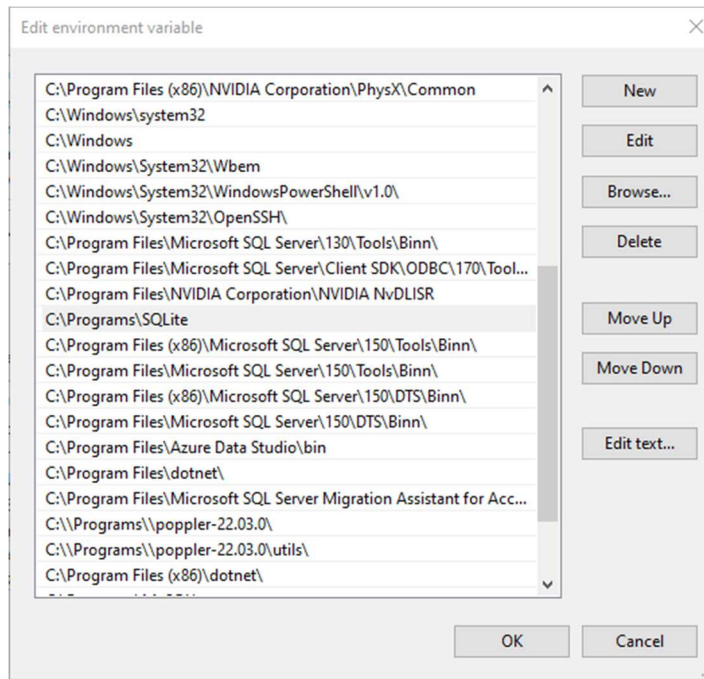
Linux

The following instructions should work in Ubuntu-based Linux distributions and others.

1. Open the Linux Terminal program and enter the following commands. You will likely be asked for your password and to verify that you wish to install the software.
    ```
    sudo apt-get update
    sudo apt-get install sqlite3 libsqlite3-dev
    ```

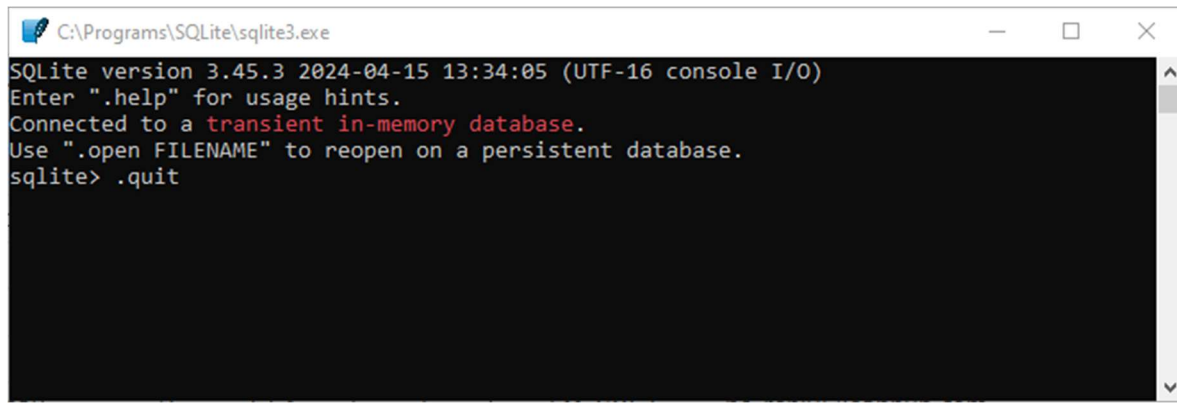2. After the installation finishes, enter the following command:
    ```
    sqlite3
    ```

3. If SQLite has been installed, this will open the SQLite console. At the console prompt, you can type `.quit` to leave the console.

4. To install the DB Browser for SQLite software, type the following command in the Terminal.
    ```
    sudo apt-get install sqlitebrowser
    ```

If you are running a Linux distribution that does not support apt-get commands, see the **Downloads** page for the browser program at the following address:

https://sqlitebrowser.org/dl

Once the installation is finished, there should be a new shortcut on your Linux menu, maybe under the **Programming** section, that will open the program.

The .quit command will close down the SQLite console.

Exploring Further

This section has shown the basics of SQLite and its installation which is enough to get you started in using the software. If you're really interested in its place in the bigger picture of database design and operations, I encourage you to bookmark the following links for reference and explore them at your leisure. Some of the information is written for experienced database professionals so don't worry if it seems fairly technical at this point.

SQLite Official Site - https://sqlite.com/

SQLite on Wikipedia - https://en.wikipedia.org/wiki/SQLite

1.2 – Downloading the Sample Databases and Code Samples

Before we can start exploring these new tools and the SQL language, we need some sample data to play around with. Fortunately, there are some excellent sample databases available for free on the web.

- The **Chinook** database is based on a fictional music company and has a dozen or so tables for albums and artists as well as employees and customers.

- The **Northwind** database is a classic order management database based on a company that sells specialty and gourmet foods. It has many of the tables you would expect including a product listing, employees, customers and customer orders.

- The **Sakila** database was originally developed for MySQL and is based on a movie rental store although the data is obfuscated with actor names like "Cuba Olivier" and fake movie titles. It's a very large database so there's plenty of room to explore.

Together, these databases will provide you with a variety of data on which you can develop your SQL skills. You can find these databases around the web but, for to keep things simple, I'm also hosting them on this book's GitHub page along with all the example queries shown in this book.

1. In your web browser, go to the GitHub page at **https://github.com/ajcomeau/SelfGuidedSQL** and click on the green **Code** button.

2. The pulldown menu will give you the option of downloading a ZIP file. Select this option and save the file to your computer in the **SQLite\Samples** directory you created earlier.

3. Extract the file to the **Samples** directory. You should now have three database files – Northwind.db, Chinook.db and sqlite-Sakila.db, along with diagrams for each database in PNG format.

Licensing

Most of the time, when downloading free software and resources from the web, you shouldn't have to

worry about licensing issues, especially if it's just for your own training use. Still, it's a good idea to be aware of the licensing terms under which you're using a piece of software. The SQLite software and sample databases each have their own licensing terms and I'm including a few notes on them below.

- The main **SQLite** software is Public Domain. According to the official site, "*Anyone is free to copy, modify, publish, use, compile, sell, or distribute the original SQLite code, either in source code form or as a compiled binary, for any purpose, commercial or non-commercial, and by any means.*" See more at https://sqlite.com/copyright.html.

- **DB Browser for SQLite** is an open-source software distributed under the Mozilla Public License, version 2 and GNU General Public License, version 3. This essentially gives you the right to freely use and distribute the software provided that, if you distribute it, you do so under the same license terms as you received it and communicate those terms as part of the distribution. See the following link for more information: https://github.com/sqlitebrowser/sqlitebrowser?tab=License-1-ov-file

- The **Chinook sample database** is copyrighted by Luis Rocha with a custom license that permits free use and distribution of the database. It can be found at https://github.com/lerocha/chinook-database?tab=License-1-ov-file.

- The **Northwind database** was originally created by Microsoft for use with Microsoft Access. Multiple SQLite versions might be available online. The primary version I found is available under the MIT license allowing for free use and distribution. See https://github.com/jpwhite3/northwind-SQLite3?tab=MIT-1-ov-file for more information. The script for the database is also available on Wikiversity at https://en.wikiversity.org/wiki/Database_Examples/Northwind/SQLite.

- The **Sakila movie database** was developed for MySQL. The SQLite version is copyrighted by the DB Software Laboratory (ETL-Tools.com) under the BSD license which permits use and redistribution on the condition that the copyright notice and disclaimers are maintained and that the copyright holder name is not used to endorse any redistribution. My copy of the database was obtained from: https://www.kaggle.com/datasets/atanaskanev/sqlite-sakila-sample-database

These links and the licenses are also included in the sample databases download file on ComeauSoftware.com.

Section 2 – Software Overview

2.1 – What is a database?

Storing the Data

The word *database* has been used in a lot of ways over the years other than the electronic databases shown in this book. Generally, a database can be any collection of information that is organized for easy retrieval and analysis. An address book is technically a database because it's a collection of contacts that has been organized into similar records that can be quickly accessed.

The electronic databases that Structured Query Language (SQL) and SQLite work with are called *relational databases* because they contain multiple tables that are referred to as *relations* in database theory. Each table is intended to hold information *related* to a specific subject.

These tables are further organized into records that might hold, for example, a specific customer's contact information and the records are divided into fields that hold specific pieces of information such as the city name in a customer's address. Records are often referred to as rows and fields are often called columns because that is the way they are represented to the user. I will use these different terms interchangeably in this book.

Retrieving the Data

SQL was originally designed in the 1970s to retrieve and manipulate data in the new IBM System R databases of the time. Another company that would eventually become Oracle saw the potential in these systems and developed their own database system with its own implementation of the language. A standard definition of the language was developed by ANSI and ISO in 1986 and has been updated every few years since then.

TrackId	Name	AlbumId	MediaTypeId	GenreId	Co
Filter	Filter	Filter	Filter	Filter	Filter
19	19 Problem Child	4	1	1	AC/DC
20	20 Overdose	4	1	1	AC/DC
21	21 Hell Ain't A Bad Place To Be	4	1	1	AC/DC
22	22 Whole Lotta Rosie	4	1	1	AC/DC
23	23 Walk On Water	5	1	1	Steven Tyler, Joe F
24	24 Love In An Elevator	5	1	1	Steven Tyler, Joe F
25	25 Rag Doll	5	1	1	Steven Tyler, Joe F
26	26 What It Takes	5	1	1	Steven Tyler, Joe F
27	27 Dude (Looks Like A Lady)	5	1	1	Steven Tyler, Joe F
28	28 Janie's Got A Gun	5	1	1	Steven Tyler, Tom
29	29 Cryin'	5	1	1	Steven Tyler, Joe F
30	30 Amazing	5	1	1	Steven Tyler, Richi
31	31 Blind Man	5	1	1	Steven Tyler, Joe F
32	32 Deuces Are Wild	5	1	1	Steven Tyler, Jim \
33	33 The Other Side	5	1	1	Steven Tyler, Jim \
34	34 Crazy	5	1	1	Steven Tyler, Joe F

19 - 34 of 3503   Go to: 1

Relational databases contain one or more tables related to specific subjects,
expressed in multiple fields over a series of records.

Managing the Data

Once you have a way to store the data and a language like SQL that can be used to manipulate it, you need a software that will manage all the processes involved. There are many different database software titles with their own database formats based on the relational model. This includes enterprise systems such as Oracle and Microsoft SQL Server, desktop database software for the office user like Microsoft Access and OpenOffice Base and databases intended for use with application development like SQLite.

Like other database software, SQLite has its own implementation of SQL with variations on syntax here and there and ways of handling different types of data. These are not major differences and the SQL you learn in this book is largely the same as that in other database software but it is important to know the quirks of any technology that you use often. Throughout the book, I will call out where SQLite's behavior differs from what you'll see elsewhere.

Takeaways

- At this point, it's important to remember the difference between the database itself which is the collection of information, the database *software* that maintains it such as Oracle or SQLite and SQL, the *language* used by the software to retrieve and update information in the database.

- You have a basic definition of the relational database model which uses tables of rows and columns to store the data.

- Structured Query Language (SQL) is used to manipulate the data and the SQLite software has its own implementation of the language which might differ slightly from the SQL used in other database software.

- There are many different database software titles intended for different environments and uses such as networked environments and application development.

2.2 – Navigating the SQLite Console

Database software often comes with different types of interfaces including graphical, menu-based browsers and text-only console programs. While browser programs are easier to navigate, it's important to understand how to work with a text-based console program. It can even be faster for certain tasks, once you're familiar with the commands and have some practice. They also tend to be more standardized in operation, as opposed to the graphical programs that can vary in layout and features. For this reason, we'll start out with an overview of basic operations in the SQLite console.
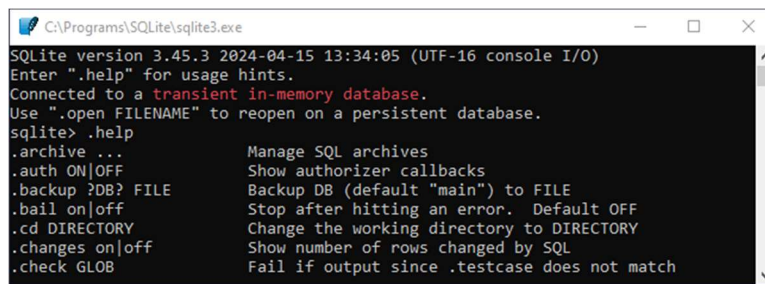
Opening a database

1. If you are running Linux or you added your SQLite directory to your Windows path statement, you should be able to open the Linux Terminal or the Command window in Windows and simply type `SQLite3` at the prompt.

The console will load with some introductory text, including the version information and instructions for a couple of commands. See the next page for an example.

2. Type "`.help`" to get a list of all console commands. Don't forget the period at the beginning. Then type `.help open` to get specific help on the .open command.

Take a few moments to look through the commands and their descriptions but don't worry if you don't understand many of them at this point. Some of them perform immediate actions while others change settings within the console. You can get additional help on any of these commands by typing `.help` followed by the command name.

The SQLite console offers extensive help on the various commands.

The first task is to open one of the sample databases that you downloaded and installed. Since you're working in the console, you can't just point and click to the file location. You need to enter the proper command with the location of the database. You can type in the full path and name of the database or you can navigate to the right directory and open the file there.

Let's look at some navigation commands. The `.shell` command gives you access to the terminal commands for whatever operating system you're using.

3. In the console, type `.shell cd` for Windows or `.shell pwd` for Linux to see the current working directory. This should be the directory where your main SQLite files are stored.

4. Use the `.cd` command to change the current working directory to the location of your sample files, i.e.:
   (Windows)   `.cd c:\SQLite\Samples`
   (Linux)   `.cd /home/user/Documents/Databases`

5. Use the `.open` command to open a sample database.
   `.open chinook.db`

It's not strictly necessary to change the working directory; you could include the entire path in the `.open` command. Still, it's good to know how to move around between directories.
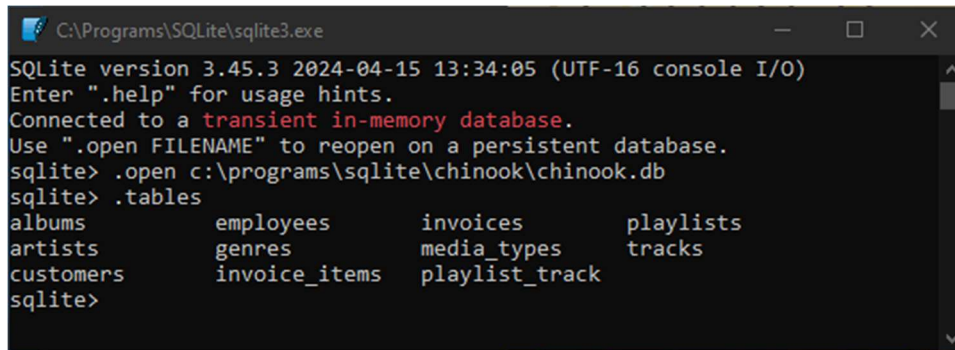
> *If there are any spaces in the directory names involved (i.e. /Documents/Sample Databases), you will need to add quotes around the path name. On Windows, you'll need to use double backslashes between the directory names. (i.e. "c:\\SQLite\\Sample Databases". Also, remember that Linux path statements are case-sensitive.*

Examining the tables

6. Use the `.tables` command to show a list of the tables in the database.

Every database has one or more tables in it that holds information on specific subjects. Many of these tables are also *related to each other* within the database so that, for example, you can retrieve lists of all

albums for a specific artist or all tracks on a specific album. The list of artists and the list of albums are stored in separate tables that have the specific fields those subjects need.

The .tables console command shows all available database tables.

These tables are related by common fields so that, as you'll see later, you can issue queries on both of them at the same time and get the necessary information.

7. Enter the following statements to get information on a couple of the tables. Don't forget the semi-colon at the end.
    ```
    pragma table_info('artists');
    pragma table_info('albums');
    ```

These are not console *commands* like `.table` or `.open`. The *pragma* keyword is an extension of SQL unique to SQLite so it doesn't start with the period and the statements do require the semi-colon at the end which tells the console that the statement is complete and can be run. If you leave the semi-colon off, the console will simply move down a line and let you keep typing until it sees a semi-colon.
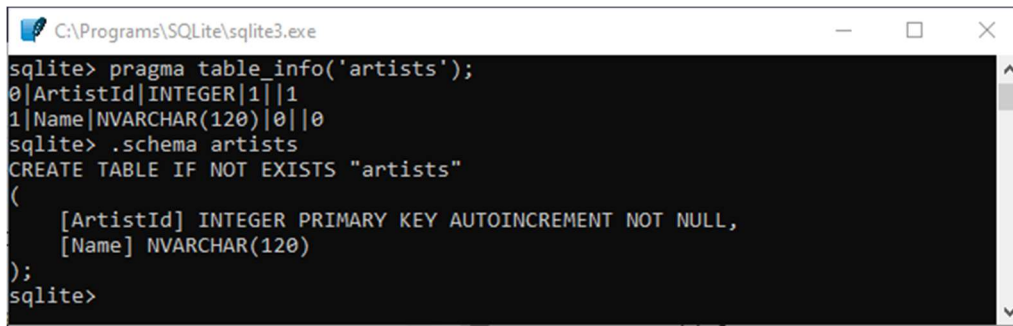
The statements give you information about the fields in the tables and their settings. In this case, you'll notice that both tables have an ArtistID field. This is the common field on which these two tables are linked. You can get the same information with the following console commands.

```
.schema artists
.schema albums
```

A schema refers to the physical structure of a database and these commands show the structure of specific tables by displaying the statements used to create them.

```
sqlite> pragma table_info('artists');
0|ArtistId|INTEGER|1||1
1|Name|NVARCHAR(120)|0||0
sqlite> .schema artists
CREATE TABLE IF NOT EXISTS "artists"
(
    [ArtistId] INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
    [Name] NVARCHAR(120)
);
sqlite>
```

Output of the .schema console command

Practice Exercise:

Try both the pragma statements and `.schema` commands on different tables within the database to see the results and help you remember the syntax. You can always use the `.tables` command to see the list of tables again.
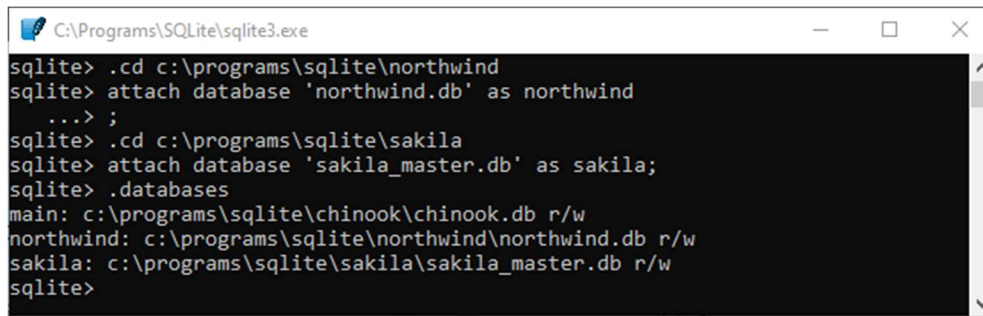
Attaching another database

You can have multiple databases open at the same time within the SQLite console. This can be handy if you have related databases or simply want to combine information from more than one database. In future chapters, you'll see examples from all three of the sample databases so you'll need to have one open and the other two attached.

8. Enter the following SQL statement in the console. I'm using the Sakila database as an example.
   ```
   ATTACH DATABASE 'sqlite-sakila.db' AS sakila;
   ```

9. List the tables again with the `.tables` command.

You now have access to all the tables in both databases with the second database using the alias 'sakila' before the table names. This helps if both databases have tables with common names. The alias itself can be anything you want it to be.

10. If you have all three sample databases downloaded, repeat the above procedure with the Northwind database.
    ```
    ATTACH DATABASE 'northwind.db' AS northwind;
    ```

You can use the UP arrow key to recall previous commands that you entered into the console. In the above example, you could recall the `ATTACH` command for the Sakila database and simply edit it to attach the Northwind database.

Output of the .databases console command showing open and attached databases in the console.

11. Use the `.databases` command to see the list of databases that are currently attached to the console.

When you're done with an attached database, you can detach it using the `DETACH` statement with the alias that you've assigned to the database.

12. `DETACH DATABASE sakila;`

Querying a table

Once you have your tables loaded, you'll want to be able to get the information out of them. Most of this book will be about the SQL statements that you use to query exactly the data you need but we can start with a basic statement just to see how they look in the console.

13. With the Chinook database loaded, enter the following command and statement.
```
.mode column
SELECT FirstName, LastName, Company, Phone
FROM Customers LIMIT 10;
```

14. If you're querying an attached database, you will need to put the database alias before the table name.
```
SELECT first_name, last_name
FROM sakila.customer LIMIT 10;
```

In SQLite, the alias for the database that you open without specifying an alias is "main". You do not need to specify it and, if your main database has a table with the same name as an attached database, a SQL query will default to it. In this book, I will specify the database in the instructions rather than the queries themselves.

Remember that you can get the information on table column names using the `.schema` command.

The first SELECT statement above instructs the program to display information from four of the fields from the Customers table. The `LIMIT 10` clause limits the output to the first 10 records it finds. This is

a helpful instruction when you just want to get a sense of what the data looks like without dumping the entire table to your screen. Some tables get really big. You'll see a lot more helpful keywords in future chapters.
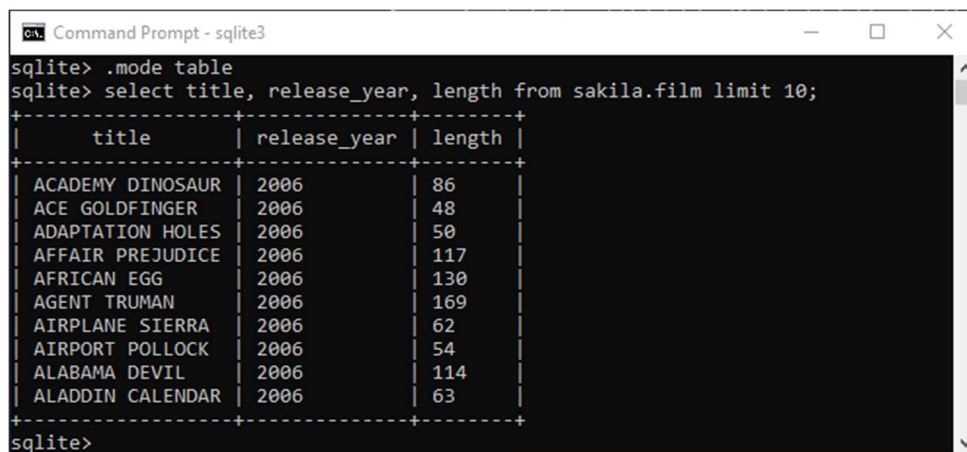
The `.mode column` command forces the console to display results of SQL statements in columns for easier reading and you should probably experiment with it at this point. This command actually has 14 different options including `.mode table` and `.mode html` which will generate HTML that can be pasted into a web page to display the query results. You can get additional information about the options by typing:

```
.help .mode
```

As a side note, SQL is usually *not* case-sensitive but the statements can get very long and complex and it's a common convention to capitalize the keywords such as SELECT and FROM to make it more readable. SQLite and other database software also doesn't care about whitespace or line breaks so SQL statements can (and should) be formatted for better readability by humans.

Quitting the console

When you want to close the console, either the `.quit` or `.exit` command will close everything down and exit.

Results of a basic SELECT statement from the Sakila movie database.

There's a lot more that you can do within the console. Throughout the book, I'll show you some examples alongside the instructions for the graphical browser environment.

Takeaways

In just these past few pages, you've seen the basics of working in the SQLite console and are off to a great start!

You should be able to:

- Open the SQLite console from the Windows or Linux command line.

- Get help within the console using the `.help` command.

- See and change the current working directory within the console.

- Open a database using the `.open` command.

- Use the `.tables` and `.databases` commands to get information on currently accessible objects.

- Use `pragma` and `.schema` to get design information on specific tables.

- Attach and detach multiple databases within the same console session.

- Issue a basic SQL statement in the console.

You have also learned about:

- Basic relational database theory concerning the use and linking of tables.

- The use of aliases to refer to databases in SQLite.

- The difference between console commands and SQL statements.

If any of these things seem unfamiliar, take some time to look back through the section and work through some of the tasks again. Experiment with different databases, tables and options so you can commit these things to memory.

If you're still a little fuzzy on some of it, don't worry - you'll gain more experience in these and other concepts in future lessons. You can also find more information online at https://sqlite.org/ and on many other websites by copying any of these items into your favorite search engine.

2.3 – DB Browser for SQLite

Now that you've gotten a look at operations in the console program, you might be looking forward to a point-and-click environment. Most database software has at least one browser program that enables you to see the content in the database at a glance and easily manage objects like tables and indexes. The one I'm using in this book is called **DB Browser for SQLite**.

1. If you downloaded the browser program as shown in section 1.1 under **Installation and Setup**, it should be already to use.

    a. On Windows, start the program by locating the file **DB Browser for SQLite.exe** under the files that you downloaded and extracted. It's best to create a shortcut to this file where you can easily get to it.

    b. On Linux, the program should be available through your main menu once it's installed. It will probably be under the **Programming** section.

    *You might notice another EXE, **DB Browser for SQLCipher.exe**. This is a separate version of SQLite that adds security features including 256-bit encryption to databases. For most of this book, we'll be working with the original, non-encrypted version.*

2. Open the program and use the **File** -> **Open** menu to open one of the sample databases. For now, I'll use the Chinook database as an example.

On the left side of the screen, under the **Database Structure** tab, you'll see a list of tables in the database. You can collapse the list using the dropdown control to the left of the Tables heading or by double-clicking the heading.

3. Select any one of the tables in the database and double-click on the name or use the arrow next to it to expand the details.
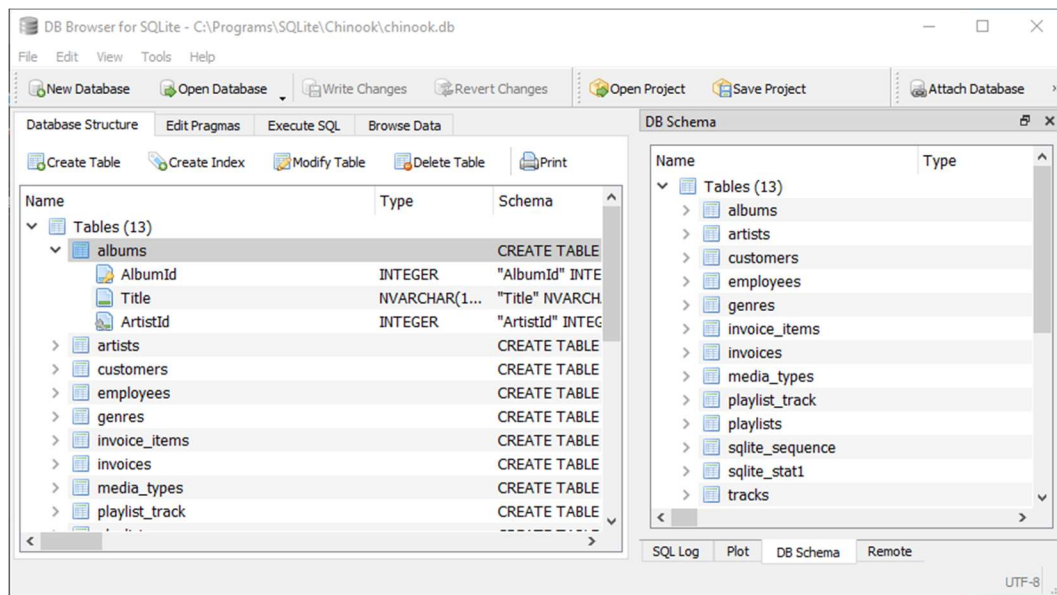
The DB Browser for SQLite interface
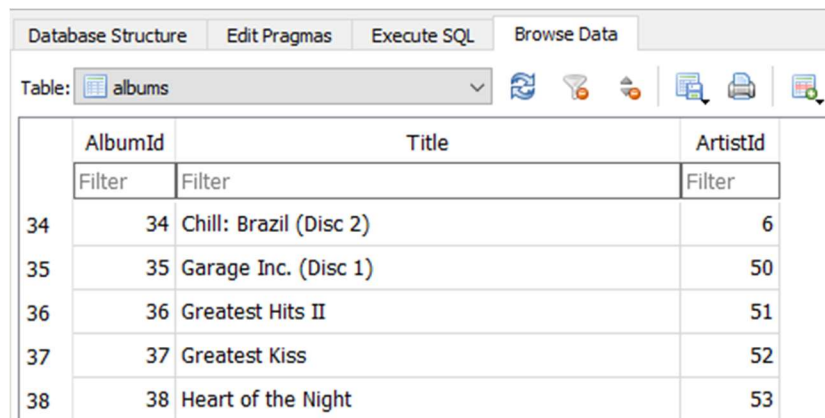
4. On the right side of the program's toolbar, you should see the **Attach Database** button which enables you to quickly attach one or more databases to the session just as you did under the console. Go ahead and attach the other two sample databases if you have them downloaded.

5. Back on the left side of the program window, select the **Browse Data** tab. This tab enables you to quickly scroll through and edit the contents of any of the tables.

   a. One of the tables should already be displayed in this tab. You can change the table shown by selecting another from the **Table** dropdown on the left side of the screen.
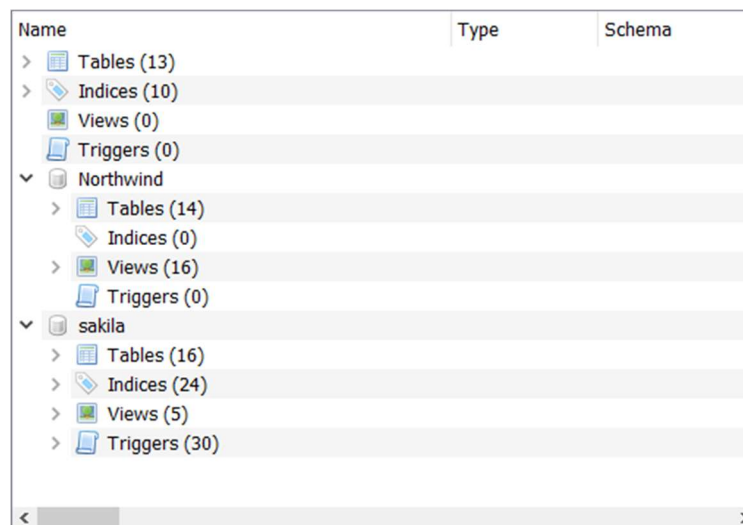
The three sample databases open or attached in the Database Structure tab

b. Click on any of the column headings in the data display to sort the table data by that column.

c. Under the column headings, there is a row of filter boxes where you can enter values to filter the rows shown.

d. Double-click on any cell in one of the tables and you'll notice a new tab appears on the right side of the program screen called **Edit Database Cell**. It will contain the contents of the cell you just double-clicked.

e. Select the **albums** table and edit one of the album titles. On the program's toolbar, you'll notice the Write **Changes** and **Revert Changes** buttons on the program's toolbar are now clickable where they were grayed out before.

The browser program requires you to actively write any changes made during the session to the database file. You can use **Revert Changes** to undo any changes you've made during the session or you can simply close the program and choose **Discard** when the program asks if you want to save the changes.

The Browse Data table lets you scroll through the contents of a table and filter the displayed rows.

6. Click on the **Execute SQL** tab on the left side of the program screen. This will display the environment where you can enter SQL statements.

7. In the SQL editing block, enter the following statement and click the F5 key or by using the start button (▶) just above the edit box :
```
SELECT * FROM albums
WHERE title LIKE '%greatest hits%';
```

You can see the results of this query statement in the screenshot above. In SQL, the LIKE keyword enables you to find values that match a certain pattern. The percent character acts as a wildcard so, in this statement, it's going to find every album title containing the phrase "greatest hits".

Also notice that the semi-colon was left off the end. The main purpose of the semi-colon is to separate SQL statements. In the console, it's really required because that's the only way the console has to know that the statement is finished. In graphical environments like this, you can sometimes get away without it.
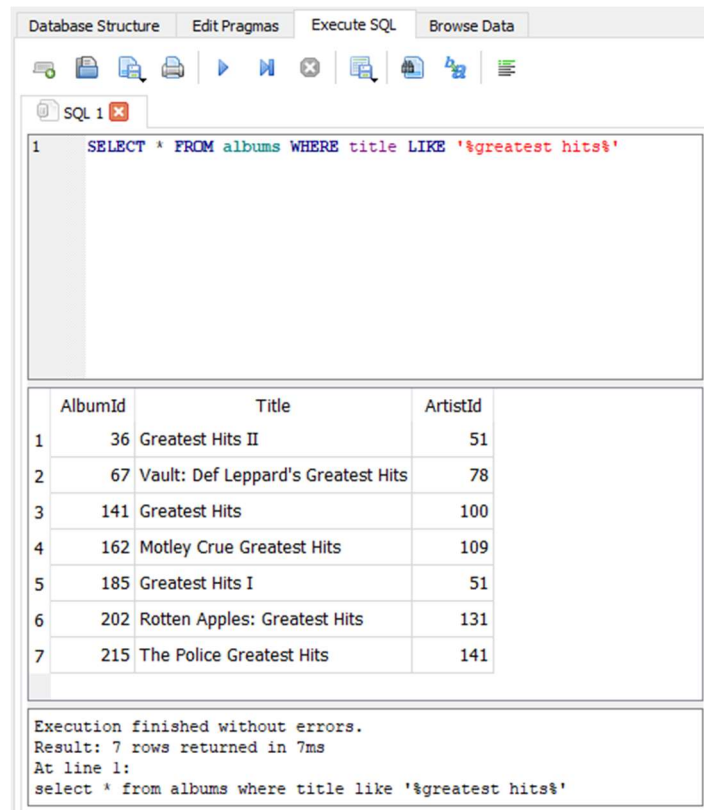
The browser program makes it easier to enter and execute SQL statements as shown here.

8. There is no interface control to detach a database when you're done working with it but you can issue the command just as you did in the console. Under the **Execute SQL** tab, enter the following statement, select it with your mouse and press F5 to run it or use the start button.
   `DETACH DATABASE Sakila;`
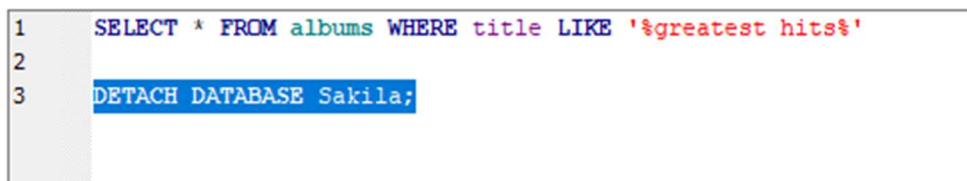
In the SQL editor, you can select specific statements to be run when you press F5.

Another advantage of the graphical environment is that you can more easily retain SQL statements you've already run for future use and select specific statements to run. If you just press F5 without selecting anything, it will run both statements, with an error in this case because of the lack of a semi-

colon at the end of the first one.

Go ahead, try it and see for yourself. We're just working with SELECT statements now but it's something to be aware of later when you start working with statements that change data.

9. After you've run the DETACH statement, try running it a second time to see what happens.

10. Use the **File** -> **Close** Database menu command or CTRL-F4 to close the current database. Click the Database Structure tab to verify that there are no tables open.

11. Use **File** -> **Exit** or CTRL-Q to exit the program.

Saving your work

The DB Browser for SQLite program has its own file extension that you can use to save your work in a session. In addition to saving your SQL queries, it saves the list of open and attached databases so that, when you reload the file, you will get the session back as you left it without having to reopen and reattach the databases.

When you're ready to save your work, you can use the **File** -> **Save Project** menu to save an *.sqbpro file to the location of your choice. This file is actually a plain text file with a combination of XML and SQL that stores all your work during the session. You can re-open it with the **File** -> **Open Project** menu options when you're ready to continue working.

Takeaways

There's a lot more that you can do in DB Browser for SQLite and you should take some time to look around the program. In these few pages, you've seen the basics and how much quicker it can be than using the console.

You should be able to:

- Find and open the DB Browser for SQLite program.

- Open a database and attach or detach secondary databases.

- View the objects within databases and browse the data in specific tables.

- Execute SQL statements within the browser's SQL editing area.

- Edit individual pieces of data within the database.

- Save or revert changes to the database during the current session.

You have also learned about:

- The SQL LIKE keyword and how to use it with wildcards.

- Separation of statements within the SQL editor.

If any of these things seem unfamiliar, take some time to look back through the section and work through some of the tasks again. Remember, you're only working with sample data that can easily be restored so now is the time to experiment. You can also find more information online on the official site for the program at https://sqlitebrowser.org/ .

Section 3 – Retrieving Data: The SELECT Statement

3.1 – Introduction

Structured Query Language (SQL) is divided into three sections:

- **Data <u>Manipulation</u> Language** (DML), the statements used to retrieve and change data.

- **Data <u>Definition</u> Language** (DDL), statements used to define and change the structures of tables and other objects in the database.

- **Data <u>Control</u> Language** (DCL), statements used to grant and revoke rights to objects within the database.

This book primarily deals with the DML portion of SQL, the statements that let you access and update data. Within that part of the language, the SELECT statement and all its keywords are what you'll be using most of the time. If you are writing statements that add, change or delete data, the ability to select the correct data for the operations is still crucial.

Throughout the rest of the book, I recommend that you have all three sample databases from Section 1.2 open or attached in either the SQLite console or the DB Browser program. The individual databases offer different opportunities for demonstrating various functions and the examples I present will switch between them as needed.

I will be providing the text for SQL statements and I recommend that you get as much practice as possible *typing the statements in* rather than copying and pasting them. In addition to the statements I provide, I will specify statements to write on your own. You will often need to determine the names of available tables and fields in the different databases. If you're using the DB Browser program, the DB Schema view on the right of the program window works well in combination with the SQL editing environment.
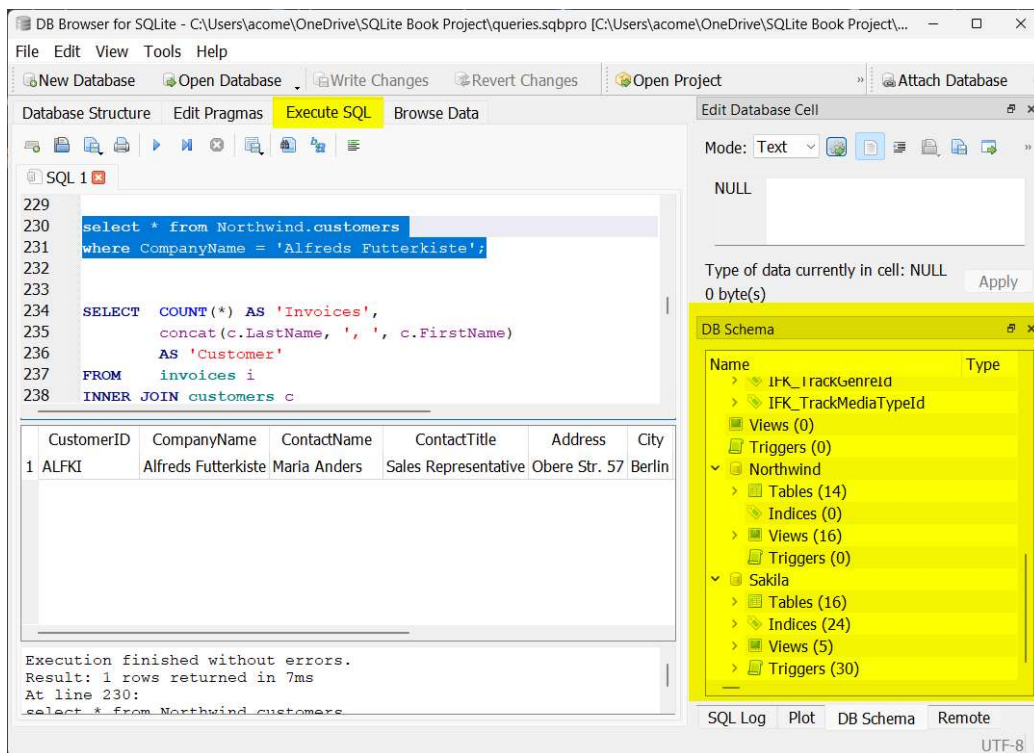
The DB Browser program provides a flexible interface for querying and managing databases.

If you're using the console, you should review the commands to examine tables including `.schema` and the `pragma table_info` statement in Section 2.

3.2 – SELECT Basics

Let's start with the most basic form of the SELECT statement.

1. Enter the following statement to query the Chinook database:

```
SELECT Name FROM Genres;
```

Name
Rock
Jazz
Metal
Alternative & Punk
Rock And Roll
Blues
Latin
Reggae
Pop

The partial results of a basic SQL statement entered in the SQLite console.

This shows just how simple SQL really can be; a four-word sentence that says, "Select this column from this table." and you've retrieved information from the database. It doesn't even have the LIMIT keyword to limit the number of rows returned as you saw earlier.

This collection of records is referred to as a *set*. Whenever you're querying a database, you are returning a set of records, even if that set only contains one record or no records at all, either by chance or design. It is a *set* of data and the number of records in it can change dramatically with small changes in the query or changes within the table itself.

This is an important point to remember, especially when you get into the parts of SQL that change or delete data. SQL can be used to target individual records but it is *designed* to work on sets and the data should always be viewed in this way.

2. Try both of the following statements:
    ```
    SELECT GenreId, Name FROM Genres;
    SELECT * FROM Genres;
    ```

Both of these statements return the same data, the GenreID and Name fields from the Genres table. As I mentioned earlier, the asterisk wildcard returns all fields in the table and there are only two fields in this table anyway.

Also notice that the genre names are in no particular order. Another aspect to relational database design is that information in a table *does not have a specific order* until it's imposed by whatever SQL statement you write. In this case, the GenreID field is a special field that automatically increments its value in order to provide a unique, identifying value for each row.

That identifier is referred to as the table's *primary key* which does provide a default order for the *display* of the table. Otherwise, data exists within a table in the order in which it is entered. You'll learn about how to sort records in your queries in an upcoming chapter.

3. Try entering this statement:
    ```
    SELECT LastName, FirstName, Title, HireDate
    FROM Employees;
    ```

This seems to be a small business since there are only eight employees and you can now see their names, their titles and when they were hired.

Just for fun, try reversing the first and last names in that query statement or putting the HireDate column first. You'll see that it doesn't matter what order you show the columns in.

	LastName	FirstName	Title	HireDate
1	Adams	Andrew	General Manager	2002-08-14 00:00:00
2	Edwards	Nancy	Sales Manager	2002-05-01 00:00:00
3	Peacock	Jane	Sales Support Agent	2002-04-01 00:00:00
4	Park	Margaret	Sales Support Agent	2003-05-03 00:00:00
5	Johnson	Steve	Sales Support Agent	2003-10-17 00:00:00
6	Mitchell	Michael	IT Manager	2003-10-17 00:00:00
7	King	Robert	IT Staff	2004-01-02 00:00:00
8	Callahan	Laura	IT Staff	2004-03-04 00:00:00

A basic SELECT from the Chinook Employees table.

Finally, remember that you can limit the number of records returned.

4. Try these two statements on the Chinook database:
    ```
    SELECT Title FROM albums LIMIT 10;
    SELECT Title FROM albums LIMIT 10 OFFSET 10;
    ```

The LIMIT and OFFSET keywords enable you to select groups of records with the OFFSET keyword skipping over the specified number of records from the start of the table. Both of the values can be replaced with formulas that will specify the number of records. It would actually be possible to get a percentage of records based on the number in the table or the values in one of the table fields. You'll see this and more about calculations in SQL in a later chapter.

Sample Edition

You've been reading a sample of "Self-Guided SQL", the newest book from Andrew Comeau and Comeau Software Solutions. This book is due to be released in 2025 and this sample edition is available for your review.

If you're interested in learning more about SQL and the SQLite database management software and have found this book helpful, please bookmark the following pages to stay up to date on the status of this release. Your feedback is also very important and you can send whatever questions or comments you have on this book to me through the site or to books@comeausoftware.com

Official page: https://www.comeausoftware.com/self-guided-sql-sqlite/

Leanpub: https://leanpub.com/self-guided-sql