04 16 SQLite 1-7

SQLite

LILLEBAELT ACADEMY OF
PROFESSIONAL HIGHER EDUCATION

Author
Martin Grønholdt
mart80c7@edu.eal.dk

Sunday 16 April 2017

Table of Contents

1 Introduction

SQLite is a lightweight SQL database library. SQL is the Structured Query Language used as a standard way to talk to a wide range of SQL database implementations. SQLite is special in that SQLite is a library that you use by making function calls, most other SQL database implementations use a client/server architecture. SQLite saves the database to a file in a cross-platform compatible format, and locks this file during write access using the available OS methods. Because of the small footprint the library is well suited for use in embedded systems and other use cases where it may be unfeasible to run a full SQL server.

Places I have encountered SQLite.

- IOS as storage for massages.
- Android as data storage for applications.

Reasons for using SQLite.

- Using a database to store data removes the need to invent new file formats and readers.
- Free concurrency when multiply sources are sending you data to store.
- Good for structuring and searching for data.
- Good for keeping historical records.

The SQLite homepage includes a page called “Appropriate Uses For SQLite”[1].

2 Download the Code

On a Gentoo Linux system to install SQLite run the following command:

```
sudo emerge sqlite
```

Command to install SQLite.

After inputting you password emerge will compile and install SQLite:

```
Calculating dependencies... done!
>>> Verifying ebuild manifests
>>> Emerging (1 of 1) dev-db/sqlite-3.17.0::gentoo
>>> Installing (1 of 1) dev-db/sqlite-3.17.0::gentoo
>>> Jobs: 1 of 1 complete                           Load avg: 2.97, 1.86, 1.55
>>> Auto-cleaning packages...

>>> No outdated packages were found on your system.

* GNU info directory index is up-to-date.
```

Output when the installation has succeeded.

The “sqlite3“ program is now available.

1 https://sqlite.org/whentouse.html

2.1 Other platforms

2.1.1 Debian

On a Debian compatible Linux system the command to do this would be:

```
apt-get install sqlite
```

Command to install SQLite.

2.1.2 Windows

For windows a guide is available at this address: http://www.guru99.com/download-install-sqlite.html

2.1.3 Mac OS

A guide to installing SQLite on Mac OS using homebrew is available at this address: http://macappstore.org/sqlite/

3 Command Line Shell For SQLite

The SQLite program is the default command line interface to the SQLite library. This program is used to manually enter and execute SQL statements from the sqlite3's command prompt.

3.1 Getting Started

3.2 Starting sqlite3 and creating a database

The sqlite3 program takes a database file name as an argument, if this file does not exists a new database file is created containing a database called "main".

Example of creating a database called example.db:

```
$ sqlite3 example.db
SQLite version 3.17.0 2017-02-13 16:02:40
Enter ".help" for usage hints.
sqlite> .databases
main: /mnt/data/Documents/Skole/It-et/2016/programming/04 16 SQLite 1-
7/example.db
sqlite>
```

Creating a new database using the sqlite3 program.

Running sqlite3 without a file name as an argument keeps the database in memory until using the .save command described later:

```
SQLite version 3.17.0 2017-02-13 16:02:40
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
Sqlite>
```

Creating an database only in memory.

3.3 How to exit sqlite3

You can terminate the sqlite3 program by typing your system End-Of-File character (usually a Control-D). The .exit command will also exit the program from the command shell as illustrated here:

```
$ sqlite3 example.db
SQLite version 3.17.0 2017-02-13 16:02:40
Enter ".help" for usage hints.
sqlite> .exit
$
```

Exiting the sqlite3 program using the .exit command.

4 Entering SQL

SQL databases are split in to tables. A table is a scheme for storing associated data in the database, not unlike a dictionary in Python. Entries in the table are organised in rows, each row follows the data scheme of the table they belong to.

4.1 CREATE a table, INSERT and SELECT

To create tables and input data the following SQL commands are used:

- CREATE – Create a new table
- INSERT INTO – Insert data in to a table in the database.
- SELECT - Returns rows from one or more tables.

customerTable				
idCust	name	email	address	city
1	Per	pda@eal.dk	Mystreet 1	Odense
2	Artur	at@hotmail.com	Allstreet 741	Vilnius
3	Alice	ab@gmail.com	Topstreet 56	London

Example data to enter into the new database.

```
$ sqlite3 example.db
SQLite version 3.17.0 2017-02-13 16:02:40
Enter ".help" for usage hints.
sqlite> create table customerTable(idCust int NOT NULL UNIQUE, name
varchar(100), email varchar(50), address varchar(50), city varchar(30));
sqlite> insert into customerTable values(1, 'Per', 'pda@eal.dk', 'Mystreet1',
'Odense');
sqlite> insert into customerTable values(2, 'Artur', 'at@hotmail.com',
'Allstreet 741', 'Vilnius');
sqlite> insert into customerTable values(3, 'Alice', 'ab@gmail.com', 'Topstreet
56', 'London');
sqlite> select * from customerTable;
1|Per|pda@eal.dk|Mystreet1|Odense
2|Artur|at@hotmail.com|Allstreet 741|Vilnius
3|Alice|ab@gmail.com|Topstreet 56|London
```

Creating a table and entering example data into the database.

Breakdown of the commands:

- `create table customerTable(idCust int NOT NULL UNIQUE, name varchar(100), email varchar(50), address varchar(50), city varchar(30));`
  Create a table named customerTable with the following fields:
  - idCust – *An integer that cannot be empty and must be unique.*
  - name – *A string with a maximum length of 100 characters.*
  - email – *A string with a maximum length of 100 characters.*
  - address – *A string with a maximum length of 100 characters.*
  - city – *A string with a maximum length of 100 characters.*
- `insert into customerTable values(1, 'Per', 'pda@eal.dk', 'Mystreet1', 'Odense');`
  Insert a row of data into the customerTable table with these values:
  - idCust – 1.
  - name – 'Per'.
  - email – 'pda@eal.dk'.
  - address – 'Mystreet1'.
  - city – 'Odense'.
- `select * from customerTable;`
  Return/show all rows in the customerTable table.

5 Special dot commands to sqlite3

When entering command in the sqlite3 shell, lines that start with a dot are intercepted by the program, whereas other commands are send to the SQLite library. Intercepting the dot commands make it possible to adjust output settings, do control functions, run shell commands, and much more. The sqlite3 program includes help for the dot commands, just type ".help".

```
sqlite> .help
.auth ON|OFF           Show authorizer callbacks
.backup ?DB? FILE      Backup DB (default "main") to FILE
.bail on|off           Stop after hitting an error.  Default OFF
.binary on|off         Turn binary output on or off.  Default OFF
.changes on|off        Show number of rows changed by SQL
.check GLOB            Fail if output since .testcase does not match
.clone NEWDB           Clone data into NEWDB from the existing database
.databases             List names and files of attached databases
.dbinfo ?DB?           Show status information about the database
.dump ?TABLE? ...      Dump the database in an SQL text format
                        If TABLE specified, only dump tables matching
                        LIKE pattern TABLE.
.echo on|off           Turn command echo on or off
.eqp on|off|full       Enable or disable automatic EXPLAIN QUERY PLAN
.exit                  Exit this program
.explain ?on|off|auto? Turn EXPLAIN output mode on or off or to automatic
.fullschema ?--indent? Show schema and the content of sqlite_stat tables
.headers on|off        Turn display of headers on or off
.help                  Show this message
.import FILE TABLE     Import data from FILE into TABLE
.imposter INDEX TABLE  Create imposter table TABLE on index INDEX
.indexes ?TABLE?       Show names of all indexes
```

```
                          If TABLE specified, only show indexes for tables
                          matching LIKE pattern TABLE.
.limit ?LIMIT? ?VAL?    Display or change the value of an SQLITE_LIMIT
.lint OPTIONS           Report potential schema issues. Options:
                          fkey-indexes     Find missing foreign key indexes
.load FILE ?ENTRY?      Load an extension library
.log FILE|off           Turn logging on or off.  FILE can be stderr/stdout
.mode MODE ?TABLE?      Set output mode where MODE is one of:
                          ascii    Columns/rows delimited by 0x1F and 0x1E
                          csv      Comma-separated values
                          column   Left-aligned columns.  (See .width)
                          html     HTML <table> code
                          insert   SQL insert statements for TABLE
                          line     One value per line
                          list     Values delimited by .separator strings
                          quote    Escape answers as for SQL
                          tabs     Tab-separated values
                          tcl      TCL list elements
.nullvalue STRING       Use STRING in place of NULL values
.once FILENAME          Output for the next SQL command only to FILENAME
.open ?--new? ?FILE?    Close existing database and reopen FILE
                          The --new starts with an empty file
.output ?FILENAME?      Send output to FILENAME or stdout
.print STRING...        Print literal STRING
.prompt MAIN CONTINUE   Replace the standard prompts
.quit                   Exit this program
.read FILENAME          Execute SQL in FILENAME
.restore ?DB? FILE      Restore content of DB (default "main") from FILE
.save FILE              Write in-memory database into FILE
.scanstats on|off       Turn sqlite3_stmt_scanstatus() metrics on or off
.schema ?PATTERN?       Show the CREATE statements matching PATTERN
                            Add --indent for pretty-printing
.separator COL ?ROW?    Change the column separator and optionally the row
                          separator for both the output mode and .import
.session CMD ...        Create or control sessions
.shell CMD ARGS...      Run CMD ARGS... in a system shell
.show                   Show the current values for various settings
.stats ?on|off?         Show stats or turn stats on or off
.system CMD ARGS...     Run CMD ARGS... in a system shell
.tables ?TABLE?         List names of tables
                          If TABLE specified, only list tables matching
                          LIKE pattern TABLE.
.testcase NAME          Begin redirecting output to 'testcase-out.txt'
.timeout MS             Try opening locked tables for MS milliseconds
.timer on|off           Turn SQL timer on or off
.trace FILE|off         Output each SQL statement as it is run
.vfsinfo ?AUX?          Information about the top-level VFS
.vfslist                List all available VFSes
.vfsname ?AUX?          Print the name of the VFS stack
.width NUM1 NUM2 ...    Set column widths for "column" mode
                          Negative values right-justify
```

The sqlite3 dot command help.

6 Writing results to a file

When a database file name is given at the command line, sqlite3 saves the database to that file. After entering the data into the database using the commands above this is illustrated by using .exit to exit sqlite3. Then starting it over and showing that the data is still there:

```
sqlite> .exit
$ ls -l
-rw-rw-rw- 1 oblivion root 12288 apr 14 19:25 example.db
-rw-rw-rw- 1 oblivion root 12288 apr 14 20:28 my.db
$ sqlite3 example.db
SQLite version 3.17.0 2017-02-13 16:02:40
Enter ".help" for usage hints.
sqlite> select * from customerTable;
1|Per|pda@eal.dk|Mystreet1|Odense
2|Artur|at@hotmail.com|Allstreet 741|Vilnius
3|Alice|ab@gmail.com|Topstreet 56|London
sqlite>
```

Illustrating that the database is saved to the database file given at the command line.

Saving to a file is also possible using the .save command followed by the desired file name:

```
sqlite> .save my.db
sqlite> .shell ls -l
-rw-rw-rw- 1 oblivion root 12288 apr 14 19:25 example.db
-rw-rw-rw- 1 oblivion root 12288 apr 14 20:28 my.db
sqlite>
```

Saving the database to a file.

The above also shows how to run a system shell command, in this case “ls -l” from inside the sqlite3 shell. You cannot however save to the file that you have given on the command line when the sqlite3 shell was invoked, as this file is locked by sqlite3, that has the file open already;

```
sqlite> .save example.db
Error: database is locked
sqlite>
```

You cannot save to the database file given on the command line.

7 Recovery

To create a backup of the database use the .backup command followed by an optional database name and the backup file name. By default uses “main” as the database name when creating new ones as well as for both backup and restore.

```
$ sqlite3 example.db
SQLite version 3.17.0 2017-02-13 16:02:40
Enter ".help" for usage hints.
sqlite> .backup main example.db.bak
sqlite> .exit
$ ls -l example.db.bak
-rw-rw-rw- 1 oblivion root 12288 apr 14 23:10 example.db.bak
```

Saving a backup copy of the database.

Recovery is done using the .recovery command followed by the optional database name and the backup file name.

```
$ sqlite3
SQLite version 3.17.0 2017-02-13 16:02:40
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .restore example.db.bak
sqlite> .databases
main:
sqlite> select * from customerTable;
1|Per|pda@eal.dk|Mystreet1|Odense
2|Artur|at@hotmail.com|Allstreet 741|Vilnius
3|Alice|ab@gmail.com|Topstreet 56|London
sqlite>
```

Restoring the database.