SQLite

About SQLite

- A free relational database system
- Most widely-deployed DB in the world
- Included in Android distributions
- Serverless
- Much easier to set up than MySQL and other free relational database systems
- Includes support for most DB features, including transactions

Installation

1. From sqlite.org/download.html:
2. unzip the files
3. put the files in a folder such as C:\sqlite
4. optionally, create a shortcut to C:\sqlite\sqlite3.exe

resources:

codeproject.com/Articles/850834/Installing-and-Using-SQLite-on-Windows

tutorialspoint.com/sqlite/sqlite_installation.htm

Starting and ending a session

Start SQL from command prompt or by clicking on sqlite3.exe

```
SQLite version 3.8.11.1 2015-07-29 20:00:57
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .help
.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
.clone NEWDB           Clone data into NEWDB from the existing
database
.databases             List names and files of attached databases
...
sqlite> .exit
```

Reading an SQL file

emacs@COMP-ROOM-PC

File Edit Options Buffers Tools SQL Help

```
drop table if exists patient;

create table patient (
  patient_no integer primary key,
  last_name varchar(64) not null,
  first_name varchar(64) not null,
  sex varchar(1) not null,
  date_of_birth varchar(8) not null,
  ward integer not null
);

insert into patient values(454, "Smith",    "John",    "M", "14.08.78", 6);
insert into patient values(223, "Jones",    "Peter",   "M", "07.12.85", 8);
insert into patient values(597, "Brown",    "Brenda",  "F", "17.06.61", 3);
insert into patient values(234, "Jenkins",  "Alan",    "M", "29.01.72", 7);
insert into patient values(244, "Wells",    "Chris",   "F", "25.02.95", 6);
```

-(Unix)--- patients.sql All L6 (SQL[ANSI])

SQLite commands start with .

SQL statements end with ;

```
sqlite> .read hospital.sql

sqlite> select * from patient limit 2;
223|Jones|Peter|M|07.12.85|8
234|Jenkins|Alan|M|29.01.72|7
sqlite>
```

Tips

`.mode column` for left-aligned columns

`.headers on` to see column headers

```
sqlite> select * from patient limit 2;
223|Jones|Peter|M|07.12.85|8
234|Jenkins|Alan|M|29.01.72|7
sqlite> .mode col
sqlite> .headers on
sqlite> select * from patient limit 2;
patient_no  last_name   first_name  sex         date_of_birth  ward
----------  ----------  ----------  ----------  -------------  ----------
223         Jones       Peter       M           07.12.85       8
234         Jenkins     Alan        M           29.01.72       7
sqlite>
```

Use up and down-arrow keys for command history

In-memory versus persistent data

- in-memory database
  - high performance
  - any changes you make will be lost when you exit SQLite
  - when SQLite starts, or when you read from a .sql file, the data is in-memory
- persistent (disk) database
  - changes to the database are changed on disk
  - .save to create a disk database
  - .open to use a disk database

Persistent data

session 1:

```
sqlite> .read hospital.sql
sqlite> .save hospital.db
```

session 2:

```
sqlite> .open hospital.db
sqlite> select * from patient limit 2;
223|Jones|Peter|M|07.12.85|8
234|Jenkins|Alan|M|29.01.72|7
sqlite> delete from patient where last_name="Jones";
```

session 3:

```
sqlite> .open hospital.db
sqlite> select * from patient;
234|Jenkins|Alan|M|29.01.72|7
244|Wells|Chris|F|25.02.95|6
454|Smith|John|M|14.08.78|6
597|Brown|Brenda|F|17.06.61|3
```

Listing tables and schemas

```
sqlite> .tables
patient
sqlite>
sqlite> .schema
CREATE TABLE patient (
  patient_no integer primary key,
  last_name varchar(64) not null,
  first_name varchar(64) not null,
  sex varchar(1) not null,
  date_of_birth varchar(8) not null,
  ward integer not null
);
sqlite>
```

Deleting a table

```
sqlite> drop table patient;
sqlite>
```

Reading/Writing CSV files

reading a CSV file:

```
sqlite> CREATE TABLE campaign (
   ...>   cmte_id              varchar(12),
   (etc.)
   ...>   election_tp          varchar(20)
   ...> );
sqlite> .mode csv
sqlite> .import campaign-all.csv campaign
```

writing a CSV file:

```
sqlite> .mode csv
sqlite> .headers on
sqlite> .out cmpgn.csv
sqlite> select * from campaign;
sqlite> .out stdout
```

Summary

SQLite:

- is a free, easy to use, serverless relational DB
- has good performance and documentation

We learned:

- how to install and use SQLite
- about in-memory vs on-disk data
- how to deal with CSV files