

# CSE 414 Introduction to Data Management

Section 2: Introduction to SQLite

# SQLite: What is it

- SQLite is a C library that implements a relational database management system (DBMS).
  - Simple, lightweight: good for embedded software
  - But does not provide all of the functionalities that other DBMSs do
- `sqlite3`: a standalone program that can run queries and manage an SQLite database

References:

<http://www.sqlite.org/lang.html> (SQL Syntax)

<http://www.sqlite.org/datatype3.html> (SQL Data type)

<http://www.w3schools.com/sql/default.asp> (w3school SQL tutorial)

# SQLite: How to Run it (1/2)

- On the Linux machines, or Mac:
  - Open a terminal, then run the command:  
`sqlite3 [database]`  
where "database" is the name of the database file you want to use.
  - WARNING: If you don't specify a database file, sqlite3 won't complain, but your data will be lost!

# SQLite: How to Run it (2/2)

- On the Windows machines:
  - Open a Cygwin terminal, then proceed as if you were on Linux.
  - If that doesn't work, you may need to install the "sqlite3" Cygwin package from Cygwin Setup.
  - If \*that\* doesn't work, try downloading sqlite yourself.
- Download it yourself:
  - Get the "sqlite-shell" binary for your OS from:  
<http://www.sqlite.org/download.html>
  - Extract "sqlite3" or "sqlite3.exe" from the archive and run it from a command line.

# SQLite: . Commands (Not SQL)

- `.help` - list other `.` commands
- `.header(s) ON/OFF` - show/hide column headers in query results
- `.mode [mode type]`- change how to separate the columns in each row/tuple (for better formatting)
- `.read [file name]` - read and execute SQL code from the given file
- `.separator [string]` - change the separator for output mode or importing files, i.e. `.separator ,`
- `.nullvalue [string]` - print the given string in place of NULL values
- `.import [file name] [table name]` - load the file to the table
  - be careful to set the separator correctly!
- `.show` - see how we have set our parameters
- `.exit` - exit from `sqlite3`

# SQLite: Basic SQL statements

- **CREATE** - creates a new table

ex) `CREATE TABLE [table] ( ... );`

- **INSERT INTO** - inserts new data into a table

ex) `INSERT INTO [table] VALUES ([value1], [value2], ...);`

- **SELECT** - extracts data from a table

ex) `SELECT [column(s)] FROM [table_name];`

- **UPDATE** - updates data in a table

ex) `UPDATE FROM [table] SET ... WHERE ...;`

- **DELETE** - deletes data from a table

ex) `DELETE FROM [table] WHERE ...;`

\*Note: Queries are case-insensitive in SQLite

# SQLite: SQL keyword, operator, etc

- WHERE clause - filter records
- AND, OR operator - filter records based on more than one condition
- LIKE operator - used in a WHERE clause to search for a specified pattern in a column
- AS - give an alias name to a table or a column
- Relational operators: =, >, >=, <, <=
- Special functions: DATE(...), LENGTH(string), SUBSTR(string, start index, end index), etc

# SQLite: Example

**Class**

dept	number	title
CSE	378	Machine Organization and Assembly Language
CSE	451	Introduction to Operating Systems
CSE	461	Introduction to Computer Communication Networks

**Teaches**

username	dept	number
zahorjan	cse	378
tom	cse	451
tom	cse	461
zahorjan	cse	451
zahorjan	cse	461
djw	cse	461
levy	cse	451

**Instructor**

username	fname	lname	started_on
zahorjan	John	Zahorjan	1985-01-01
djw	David	Wetherall	1999-07-01
tom	Tom	Anderson	1997-10-01
levy	Hank	Levy	1988-04-01



# SQLite: things to watch out for

- SQLite allows a key to be null
- Older versions of sqlite do not enforce FOREIGN KEY constraints.
  - Newer versions are opt-in at both compile time and runtime (with `PRAGMA FOREIGN_KEYS = ON`)
- SQLite ignores string length maximums or fixed string lengths: N in `VARCHAR(N)` or `CHAR(N)`
- SQLite does not have a separate data type for dates, times, or combined date and time.
  - Instead, these are represented as specially formatted strings; dates are represented as `yyyy-mm-dd`
- And many more as you will discover!