# Quick Guide to SQLite, MySQL, and PostgreSQL

To try some SQL queries yourself, and to do the exercises included with this course, you'll need to install a SQL-based database management system on your computer. There are three main free implementations: SQLite, MySQL, and PostgreSQL (usually called Postgres). Here we'll walk you through the installation procedures for these three systems, with instructions for both Windows and Mac OS X users.
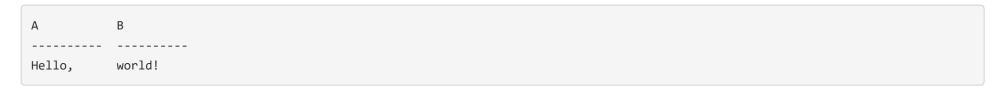
## SQLite

We encourage students just getting started with SQL experimentation to install and use the SQLite command-line shell first. Since it is a stand-alone executable file, there is no need to set up a server and client to connect.

**Windows:** Go to the SQLite download page and find the heading for "Precompiled Binaries for Windows." Download the .zip file for the command-line shell (not DLL or analysis). Extract the sqlite3.exe file and place it wherever you like; the desktop is a good start. Double-click the extracted sqlite3 executable to run.

**Mac OS X:** Go to the SQLite download page and find the heading for "Precompiled Binaries for Mac OS X." Download the .zip file for the command-line shell (not analysis). Double-click to extract the sqlite3 file, and place it wherever you like; the desktop is a good start. Double-click the extracted sqlite3 executable to run.

Once you're in the SQLite command-line shell, you have three options to run queries: you can type them by hand each time, copy them from somewhere else and paste them into the command-line shell window, or execute a file as a script. To try the latter two options, take a look at our SQLite starter script. If you copy-paste it into the SQLite window, it should generate the following output:

```
A           B
----------  ----------
Hello,      world!
```

To try executing a SQL file from the command-line shell, download the starter script, place it in the same directory as the SQLite executable, and type the following at the prompt:

```
sqlite> .read sqlite-start.sql
```

The output should be the same as with the copy-paste method. To see the full list of SQLite-specific commands like '.read', type '.help' or consult the special commands list in the SQLite command-shell documentation.

When you're finished, type '.exit' to exit the program. Note that the database state is not saved between sessions, so you will have to reload any data from scratch for each session. If you want to avoid this and you're familiar with the command line in your OS, you can execute SQLite from there with a single

argument:

```
sqlite database-filename
```

This will cause SQLite to open and use the file called 'database-filename', or create it if it does not already exist. The database state will be saved to that file as you work, and you can use it again later.

## MySQL

For more detailed instructions, see the official MySQL installation notes.

**Windows:** Go to the MySQL Community Server download page and select the MSI installer for your architecture (if in doubt, 32-bit is a safe bet). You may choose to register as a new user or proceed with the download without registering. Run the installer (a 'Typical' installation is fine). The installer will then run the MySQL Server Instance Configuration Wizard. Install MySQL as a Windows service and include the bin directory in Windows PATH. Create a root account with appropriately secure password (and one that you'll remember!), and make sure to disable root access from remote machines. When completed, MySQL will be running as a service on your computer. To connect and use it, select 'MySQL Test User' in the MySQL directory of the Start Menu.

**Mac OS X:** Go to the MySQL Community Server download page and select the DMG archive for your architecture (if in doubt, 32-bit is a safe bet). You may choose to register as a new user or proceed with the download without registering. Open the disk image file, and run both of the package installers (MySQL and MySQLStartupItem). All standard settings and installation locations should be fine. To get to the MySQL shell, you will need to open a terminal window and run the following commands:

```
sudo /Library/StartupItems/MySQLCOM/MySQLCOM start
/usr/local/mysql/bin/mysql
```

The first command starts the MySQL server (it will likely require that you enter your login password to proceed), and the second connects to the server and opens the MySQL shell. Note that the default setup uses no passwords; see the official documentation on securing the initial MySQL accounts to set your own.

Once you're in the MySQL command-line shell, you have three options to run queries: you can type them by hand each time, copy them from somewhere else and paste them into the command-line shell window, or execute a file as a script. To try the latter two options, take a look at our MySQL starter script. If you copy and paste its contents into the MySQL window, it should generate the following output (you may get a warning that the table does not exist on the first time you run it; this is OK):

```
+--------+--------+
| A      | B      |
+--------+--------+
| Hello, | world! |
```

```
+--------+--------+
1 row in set (0.00 sec)
```

To try executing a SQL file from the command-line shell, download the starter script, place it somewhere that you can easily remember the file path, and type the following at the prompt (where ~PATH~ represents the path to the file):

```
mysql> source '~PATH~/mysql-start.sql'
```

The output should be the same as with the copy-paste method. To see the full list of MySQL-specific commands like 'source', type 'help' or consult the commands list in the MySQL documentation.

When you're finished, type 'quit' to close the connection. The database state will be saved, so all your tables will still be there when you connect again later.

## PostgreSQL

For more detailed instructions, see the PostgreSQL installation guides.

**Windows:** Go the the PostgreSQL download page and select the version 9 installer for your architecture (if in doubt, 32-bit is a safe bet). All default settings are fine; adjust as you like. Make sure to set a secure password (and one that you'll remember!) for the database superuser account. You will not need Stack Builder for the exercises in this course. After installation, you can run the Postgres SQL shell psql, linked from the Postgres 9.0 folder in the Start Menu. Press Enter to use the default server, database, port, and username; then type the password you entered earlier.

**Mac OS X:** The install procedure for Mac OS X is not as simple, and requires some familiarity with the command line, willingness to work with system settings, and more space to describe than we have here. See this guide using the Homebrew package manager, or alternatively, the Postgres download page and Apple developer notes, to get started.

Once you're in the Postgres SQL shell, you have three options to run queries: you can type them by hand each time, copy them from somewhere else and paste them into the SQL shell window, or execute a file as a script. To try the latter two options, take a look at our Postgres starter script. If you copy and paste its contents into the Postgres window, it should generate the following output (you may get a notice that the table does not exist on the first time you run it; this is OK):

```
   a    |   b
--------+--------
 Hello, | world!
(1 row)
```

To try executing a SQL file from the SQL shell, download the starter script, place it somewhere that you can easily remember the file path, and type the following at the prompt (where ~PATH~ represents the path to the file):

```
postgres=# \i '~PATH~/postgres-start.sql'
```

The output should be the same as with the the the copy-paste method. To see the full list of Postgres-specific output commands like '\i', type '\?' or consult the meta-commands list in the Postgres psql documentation.

When you're finished, type '\q' to close the connection. The database state will be saved, so all your tables will still be there when you connect again later.