Installing and Using MySQL with MAMP on Mac OS X

March 1, 2021

This is what I used when I taught CSE 305 (Database Systems) in 2021. I think the info on this page is still valid. If necessary, you may only need to adjust slightly, I think.
Install MAMP which will allow you to use MySQL and JDBC on your Mac OS X.

1. Installing MAMP on Mac OS X

Follow these steps to install MAMP:

1. Download MAMP from https://www.mamp.info/en/. Click on **Free Download**. You will have two options there as follows:
    - You may buy **MAMP & MAMP PRO 6.3**. Yes, you will have to pay for it after a 14-day free trial. Read on for a free version.
    - I will use a free version myself and you should be fine if you use a free version yourself for this class. Go to the bottom of that page and you will see **Older MAMP & MAMP PRO versions**. From there the latest free version is **MAMP & MAMP PRO 5.7** (i.e., **MAMP_MAMP_PRO_5.7.pkg**).
2. Install your choice accepting defaults.
3. You will end up with a folder named /Applications/MAMP and in it you will see MAMP.app that you can put on the dock if you want.
4. You can start the MySQL server (and optionally Apache server) using MAMP.app. After you open the app, you can "Start Servers" which will start both Apache Sever and MySQL Server although you will only use the MySQL Server initially. How do you know the server is running? You will see the dots change their colors on the interface window. Alternatively, you can start a Unix Terminal window and try the "top" command on the Unix prompt. You will then see a process named "mysqld" along with a bunch of "httpd" processes.
5. By now, you have installed the MySQL server (as a part of MAMP) and the *server* is running. Now let's open a MySQL *client* and try some commands as explained in the next section. So, you are using a client-server model of computing here.

2. Running a MySQL Client and Accessing the Server

Continuing from section 1 above, a MySQL server (MAMP) should be running at this point. Let us start a MySQL client shell and access the server. Try the following steps:

1. Open a Unix Terminal window and start a client using the command:
   ```
   shell> /Applications/MAMP/Library/bin/mysql -u root -p
   ```
   and use root as the password to continue.

2. If you are not comfortable with setting up a command line interface with MySQL, see this tutorial. If you set up your PATH variable as described in the tutorial, you can try the following simpler command instead:
   ```
   shell> mysql -u root -p
   ```

3. By now, you should be on a mysql prompt. To see a listing of all the existing databases, try:
   ```
   mysql> show databases;          // don't forget the ';'
   ```

4. You can create a new database named artlee like this:
   ```
   mysql> create database artlee;
   ```

5. You can create a user while granting privileges to that user. To grant a user access to the database from the localhost, use the command:
   ```
   mysql> grant all privileges on database.* to user@localhost
   identified by 'password';   // include the single quotes
   ```
   where database is the name of a database, user is the name of a new user, and password is a password. For example, I used artlee, art, and lee for database, user, and password, respectively. Don't forget the single quotes.

6. To grant the user rights to the database from other client machines, use the command:
   ```
   mysql> grant all privileges on database.* to user@"%"
   identified by 'password';
   ```
   where @"%" acts as a wildcard for access to the database from any client machine. Here again database is the name of a database, user is the name of a user, and password is a password. For example, I used artlee, art, and lee respectively.

7. If you want some help, try:
   ```
   mysql> help;
   ```

8. Assume that you have multiple databases in the system. To select a database (e.g., named 'artlee'), which you will need to do before you start using it among all the databases that you have created, try:
   ```
   mysql> use artlee;
   ```

9. You may even delete a database. To delete a database named 'artlee', try:
   ```
   mysql> drop database if exists artlee;
   ```

10. Running a script file. You may create a database and populate the database with some tables and records in the tables. It is more convenient to do it using a script file. If you

want to run the script file: db.sql located for example in /Users/alee/cse305/db/db.sql, try:

```
mysql> source /Users/alee/cse305/db/db.sql;
```

Take a look at the content of the script file db.sql. By the way, how would you create a script file? Use your favorite text editor.

11. If you want to leave the server, try:

```
mysql> quit;
or
mysql> exit;
```

3. Some More Examples of SQL Queries

Let's assume that you have a server running, and you just started a client. Try the following on the client, which are similar to the commands that you saw in db.sql:

- Delete the database named artlee first if it exists so that you know you are starting with a clean state:

```
mysql> drop database if exists artlee;
```

- Now, create a new database named artlee:

```
mysql> create database artlee;
```

- To select a database named artlee:

```
mysql> use artlee;
```

- Creating a table named Accounts in a database named artlee:

```
mysql> create table Accounts (ID INTEGER, Balance INTEGER);
```

- To list the tables in a database named artlee:

```
mysql> show tables;
```

- See the contents of the table just created (of course, it would be empty at this point):

```
mysql> SELECT * FROM Accounts;
```

  Use this again below to see if insert worked or not when you try inserts below.

- Creating a relation (also called tuple or a row) in a table (three times):

```
mysql> insert into Accounts VALUES (1, 11);
mysql> insert into Accounts VALUES (2, 22);
mysql> insert into Accounts VALUES (3, 33);
```

- Add up the balances in the Accounts table:

```
mysql> SELECT Sum(Balance) FROM Accounts;
```

- Add another table and some tuples into the table (I intentionally used lower case letters here to show that it is not case insensitive, but try to use a good style in naming):

```
mysql> create table Names (id integer, dob date);
mysql> insert into Names values (1, "2021-03-02");
mysql> insert into Names values (2, "2019-03-20");
mysql> insert into Names values (3, "2017-01-20");
```

- The cartesian product of the Names and Accounts relations if both exist:
  ```
  mysql> SELECT * FROM Names, Accounts;          -- (+)
  ```
  Use '--' to add a comment.

- Keep only those records from (+) that agree on ID:
  ```
  mysql> SELECT * FROM Names, Accounts WHERE Names.ID =
  Accounts.ID;
  ```

- Keep only those records from (+) that have a DOB prior to 1/1/2021:
  ```
  mysql> SELECT * FROM Names, Accounts WHERE Names.ID =
  Accounts.ID
               AND DOB < Date("2021-01-01");
  ```

- Keep only the Names.id, Balance, and DOB columns:
  ```
  mysql> SELECT Names.id, Balance, DOB from Names, Accounts
  WHERE
               Names.ID = Accounts.ID AND DOB < Date("2021-01-01");
  ```

- Try more that you can think of. Find a reference if you need.

4. Using MySQL with Java (JDBC)

Up till now, we saw how a client communicates with a database server through command line interface. In this section, I describe how a Java program as a client communicates with a database server. Java does it through a connector called JDBC. I assume that you have MySQL (in MAMP) installed by now.
Find a Java connector, MySQL Connector/J, from http://dev.mysql.com/downloads/connector/j/.
I downloaded mysql-connector-java-8.0.23.zip (the latest at the time of this writing). When you unpack it, you will see a file that looks like this:

```
  mysql-connector-java-8.0.23-bin.jar
```

To make compilation simpler, add this jar file to the CLASSPATH in .profile (adjust it if you are not using a Bourne-compatible shell) in your home directory. My CLASSPATH looks like this:

```
CLASSPATH=".:/Users/alee/cse305/mysql-connector-java-8.0.23/mysql-conn
ector-java-8.0.23-bin.jar:${CLASSPATH}"
  export CLASSPATH
```

Now, download, compile, and run Bank1.java which will bring in DB.java. They use JDBC. Be sure to read the comments at the top of Bank1.java to see how to run it.

- jdbc.sql (Run this script from a command line prompt to set up the database first. Read the file. It creates a simple bank database.)
- DB.java. In DB.java you will notice that I am using the port number 8889. That is the port

number that you will see in the MySQL section on the "Welcome to MAMP" page when you "Start Servers" with MAMP.

- Bank1.java (A Java program that uses JDBC to talk to a database.)

If This Does NOT Work and nothing else could be tried to make it work, uninstall MAMP and anything else that could have installed MySQL, and then follow the instructions *very carefully* from the very beginning. Some students had to that and it worked for them.
That's it!