Work with LabThreeSixFive.com MySQL client

MySQL

MySQL is a popular open-source DBMS originally developed under the auspices of Sun Microsystems, and currently developed and distributed by Oracle (after Oracle's acquisition of Sun).

MySQL server comes with a simple interactive client program, mysql. This program allows users to connect to a MySQL server of their choice, send commands to the server and observe output of the commands.

This handout is a brief "survival manual" that explains how to set up your mysql client to work with the course's MySQL server, and how to work with the client.

LabThreeSixFive.com

LabThreeSixFive.com is an interactive web-based application that is designed to facilitate learing SQL in CSC 365. It was developed by Andrew Migler, one of the Department's CSC 365 instructors as part of his effort to improve the feedback students get when working on CSC 365 SQL query assignments.

LabThreeSixFive.com works with a MySQL server located at mysql.labthreesixfive.com.

You need to be aware of the following information:

- Your LabThreeSixFive.com accounts (loginId/password) will let you log into the web UI of LabThreeSixFive.com **and** access mysql.labthreesixfive.com MySQL server directly.
- You cannot change your password via the LabThreeSixFive.com UI.
- You *can* change password on the mysql.labthreesixfive.com MySQL server directly via the MySQL client, **but doing so locks you out of LabThreeSixFive.com - you won't be able to log in on the web portal anymore**.
- LabThreeSixFive.com collects usage information. The system tracks all SQL commands run from various environments. The LabThreeSixFive.com lab assignment modules will allow you to test whether the output of your query is the same as the output of instructor's query. For some tasks, the system will also allow you to review the exact expected output (i.e., the output of instructor's solution). If you *opt to participate in our study*, the information will be used (anonymously) to analyze how students learn SQL. If you opt *not to participate in the study*, the collected information will be used for grading purposes in the course, but will not be analyzed for any other purpose.
- The mysql.labthreesixfive.com MySQL server does not collect usage statistics and any queries or commands you run directly via any MySQL client that is connected directly to that server will be neighther recorded for posterity (beyond the client's command buffer which is accessible to you but not to us) nor used in any research, nor passed to the LabThreeSixFive.com web application.

Setup Instructions

These setup instructions refer to running mysql client from the CSL linux machines, such as the ones found in 14-231A and 14-235 and the CSL servers `unix1.csc.calpoly.edu` through `unix6.csc.calpoly.edu`.

`mysql` client. The `mysql` client prgram is installed on all machines in CSL and on all CSL servers.

```
dekhtyar@unix1:~ $ which mysql
 /bin/mysql
```

There are two ways to access the mysql.labthreesixfive.com server. One way uses only command line options for invoking the mysql client. The other uses a cofiguration file.

Accessing our server through command-line options. In order to correctly access the CSC 365 MySQL server from any of the CSL machines or from the CSL servers, you shall type the following

```
$ mysql -h mysql.labthreesixfive.com -P 3306 -p
```

or

```
$ mysql -h mysql.labthreesixfive.com  -P 3306 -u <username> -p
```

where `<username>` is your Cal Poly login id.
Note: please note the captialization of the command-line options. Here:

Option	Explanation
`-h`	host name
`-P`	port number
`-u`	username
`-p`	password. If followed by a string, the string is treated as the password if no value given, mysql client will request password interactively

Accessing MySQL Server via a configuration file. On my CSL linux account, `mysql` is aliased as follows:

```
dekhtyar@londo:~/classes/365/lectures $ which mysql
alias mysql='mysql --defaults-extra-file=~/my.cnf'
        /bin/mysql
```

You can set **your** mysql command (or any other command, e.g., `mysql365` to be aliased this way by including in the bottom of the `.bashrc` file located in your CSL home directory, the following command[1]

```
alias mysql="mysql --defaults-extra-file=~/my.cnf"
```

This mysql invocation method uses a configuration file called `my.cnf` located in the home directory of the user account (note: you can override both the name of the file and its location in your home directory structure as you see fit. These instructions are based on the default and commonly acceptable setup.)

The `--defaults-extra-file` option overrides the behavior of the `mysql` client and makes it use the settings from the configuration file provided - in our case - `my.cnf`.

The `my.cnf` file shall have the following contents:

```
[mysql]
host=mysql.labththreesixfive.com
port=3306
user=<your UserId>

password
```

Note, we are using the same commands as the command-line option list, only their verbose variants:

Line	Explanation
`[mysql]`	Config files are used to control different MySQL executables. This line declares a block of options for the `mysql` client.
`host=<hostname>`	specifies the server to which `mysql` client will connect.
`port=<port>`	specifies the port to `mysql` client will connect.
`user=<username>`	specifies the user under which name `mysql` client will connect.
`database=<dbName>`	specifies the database in which the `mysql` session will start.
`password`	specifies that `mysql` client shall prompt the user for a password.

Work with MySQL Client

MySQL client commands are available for use **both** when you are connected directly to the MySQL server via a mysql client, and when you are working using the LabThreeSixFive.com UI/UX in your browser.

[1] If you want to immediately test how this command works, save the `.bashrc` file and at the command prompt, issue the `source .bashrc` command to rerun the `.bashrc` script. On all subsequent Linux sessions the new alias will be available to you at the start.

Note on commands. mysql client processes some commands on the client side, while other commands are passed to the server. We will refer to the commands processed on the client side as *mysql commands*, and to the commands passed to the server as *SQL commands* or *SQL statements*.

The **key difference** between the *mysql commands* and the SQL commands is the termination.

- **SQL commands** are typically multi-line, they require a terminator. A typical terminator you see is `";"` (a semicolon).
- **mysql commands** do not have a terminator. Press <Enter> after typiing the command, and mysql client will execute it.

In LabThreeSixFive.com UI/UX, the ";" terminators are optional.
Below is a short summary of useful commands. Note that we show the terminators (when needed).

Command	Type	Explanation
`show databases;`	SQL	Output the list of all databases your account has access to
`use <Database>`	mysql	Change current database
`show tables;`	SQL	List all tables in the current database
`describe <Table>;`	SQL	show the schema of a given relational table
`show CREATE TABLE <Table>;`	SQL	show the CREATE TABLE statement for a given table
`source <filename>`	mysql	run a SQL script from a given file
`tee <filename>`	mysql	record in a given file a log of all subsequent commands
`notee`	mysql	stop logging

Database selection

All MySQL activity happens inside a selected database. With each student accout, we associate the database under the same name. Outside of some MySQL's test databases, this is the only database you have access to.

When you see the following `mysql` client prompt:

```
mysql[none]>
```

it means that you have not yet selected the database. You need to do so before continuing any other work. Note, sometimes, your prompt looks as follows:

```
mysql>
```

but the database has not yet been set. Therefore it is safe to always execute the commands below at the beginning of your session.

First, you can see what databases are available by issuing the `show databases;` command:

```
mysql[none]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| dekhtyar           |
+--------------------+
3 rows in set (0.00 sec)
```

Here, `dekhtyar` is the instructor's designated database, while `information_schema` is a MySQL's internal database with open access (ignore it).

To select the database you want to work with, issue the `use` command:

```
mysql[none]> use dekhtyar;
Database changed
mysql@dekhtyar>
```

(Note: your prompt may vary at this point.)

Requesting database info. `show tables` is a very useful command. It allows you to see the list of tables in your current database. Here is a sample interaction:

```
MySQL [dekhtyar]> show tables;
+--------------------+
| Tables_in_dekhtyar |
+--------------------+
| ArtistGenres       |
| Artists            |
| genres             |
| students           |
+--------------------+
4 rows in set (0.02 sec)
```

Here the database `dekhtyar` contains four tables that are listed in the output of the command.

Reviewing the schema of a table. Another very useful ability is to review the schema of a single database table. Use the `describe` command for that. The syntax is

```
describe <tableName>;
```

where `<tableName>` is a name of a table in the current database. The output is rather verbose, as seen in the example below.

```
MySQL [dekhtyar]> describe genres;
+-------+-------------+------+-----+---------+----------------+
| Field | Type        | Null | Key | Default | Extra          |
+-------+-------------+------+-----+---------+----------------+
| id    | int(11)     | NO   | PRI | NULL    | auto_increment |
| genre | varchar(24) | YES  | UNI | NULL    |                |
+-------+-------------+------+-----+---------+----------------+
2 rows in set (0.02 sec)
```

Each row is one attribute definition. Column specifications are as follows:

Column name	Explanation
`Field`	attribute name
`Type`	SQL data type of the attribute
`Null`	whether the attribute can take `NULL` values (Yes/No)
`Key`	whether the attribute belongs to a key (Primary, Unique/Candidate key, or none)
`Default`	Default value of an attribute (usually `NULL`
`Extra`	any additional constraints/specifications

Note, that the output of the `describe` command is a list of attributes of a given table, not a full table specification.

View CREATE TABLE statement for a table. If you want to view the entire definition of a given relational table, you can use the `show CREATE TABLE` command. It has the following syntax:

```
show CREATE TABLE <tableName>;
```

where `<tableName>` is the name of the table of interest in the current database.

Here is an example run:

footnotesize

```
MySQL [dekhtyar]> show CREATE TABLE genres;
+--------+-----------------------------------------------------------------------------+
| Table  | Create Table
+--------+-----------------------------------------------------------------------------+
| genres | CREATE TABLE `genres` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `genre` varchar(24) DEFAULT NULL,
```

```
  PRIMARY KEY (`id`),
  UNIQUE KEY `genre` (`genre`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+--------+-----------------------
```

Note: the `CREATE TABLE` statement returned by the server contains more information that a `CREATE TABLE` statement what actually created it. What MySQL returns is the full *expanded* version of the `CREATE TABLE` statement. Some of the components of this statement are default behaviors that are omitted from the `CREATE TABLE` statement written by a human analyst. For example, the `CREATE TABLE` statement for the `genres` table above written by the instructor looks as follows:

```
CREATE TABLE genres(
id INT PRIMARY KEY AUTO_INCREMENT,
genre VARCHAR(24) UNIQUE
);
```

Work with SQL scripts.

Any file containing SQL comments, SQL commands and mysql client commands can be run by the `mysql` client.

There are two ways this can be done: from within the client (interactively) and in a batch mode.

Running SQL scripts interactively. To run SQL scripts interactively, use the source command:

```
mysql> source <filename>;
```

Running SQL scripts in batch mode. From the Linux prompt, enter the following command:

```
$ mysql [options]  < <filename>
```

Here, `options` are the command-line parameters you want `mysql` client to run with.

Comments. Comments in SQL scripts are any lines that start with a double dash `'--'`. Comments can also be put at the ends of the lines, as shown in the example below.

Consider the following simple SQL script `test.sql`.

```
-- Alex Dekhtyar
-- CSC 365

CREATE TABLE test(
   id int,
   name varchar(20),
   PRIMARY KEY(id)   -- primary key declaration on a separate line
)
;

INSERT INTO test VALUES(1,2),(3,4),(5,6);

select * from test;
```

Running the script interactively yields the following output:

```
mysql> source test01.sql
Query OK, 0 rows affected (0.02 sec)

Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0

+----+------+
| id | name |
+----+------+
|  1 | 2    |
|  3 | 4    |
|  5 | 6    |
```

```
+----+------+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Running the same script in the batch mode results in the following output:

```
 $ mysql <test01.sql
Enter password:
id      name
1       2
3       4
5       6
```

Verbose mode.

`mysql` client can be run in a `verbose` mode, in which all commands sent to the server as echoed on the terminal. To engage the mode, the `mysql` client needs to be started with the `--verbose` or `-v` flag.

For example:

```
 $ mysql -v <test01.sql
Enter password:
--------------
CREATE TABLE test(
   id int,
   name varchar(20),
   PRIMARY KEY(id)
)
--------------

--------------
INSERT INTO test VALUES(1,2),(3,4),(5,6)
--------------

--------------
select * from test
--------------

id      name
1       2
3       4
5       6
```

Log of **mysql** activity

You can create logs of your mysql activity using the `tee` command:

```
mysql> tee <filename>
```

where `<filename>` is the name of the file you want to log your commands to will result in logging of all your `mysql` activity to the named file.

When you want to stop logging you can issue the `notee` command:

```
mysql> notee
```

Here is a simple session:

```
mysql> tee test.out
Logging to file 'test.out'
mysql> source test01.sql
Query OK, 0 rows affected (0.01 sec)

Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

+----+------+
| id | name |
+----+------+
|  1 | 2    |
|  3 | 4    |
```

```
|  5 | 6    |
+----+------+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql> notee
Outfile disabled.
mysql> exit
Bye
dekhtyar@londo:~/classes/365/scripts $ cat test.out
mysql> source test01.sql
Query OK, 0 rows affected (0.01 sec)

Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

+----+------+
| id | name |
+----+------+
|  1 | 2    |
|  3 | 4    |
|  5 | 6    |
+----+------+
3 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql> notee
```