A202/A598 and A290/A590 Lecture

Feb 22, 2010 11:45am-12:30pm

I announced on the website (late, though) that printing was holding me, and so I was late. I came at about the time Nick and Jared were getting out of the building (and I don’t blame them since I ended up being late more than anticipated). In any event we still had 45 minutes worth of time and so we proceeded as follows. We organized ourselves in three teams:

<table>
<thead>
<tr>
<th>Team 1</th>
<th>Team 2</th>
<th>Team 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dmitriy, Andrew and Grant</td>
<td>Navin, Linwood and Karys</td>
<td>Josh, Marie and Allen</td>
</tr>
</tbody>
</table>

Recall the document that was distributed started with some information about Python access to MySQL. It started like this:

```
Today we will discuss Python database access.

These are the facts:

1. Every Python program we wrote starts with the line below, also needs to be made executable:
```

```shell
#!/usr/bin/python
```
2. The routines that allow your Python program to access databases are in this module:

```python
import MySQLdb
```

3. To connect to a database one needs to invoke the right function, and pass the right params:

```python
Con = MySQLdb.connect(host="silo.cs.indiana.edu", \
                      port=57862, user="dgerman", \
                      passwd="sp00n", db="demoOne")
```

What you see above is a connection to a fictitious MySQL server on silo. Adapt accordingly. The thing you get back is an object, we think of it as being “the connection.”

4. A connection can give you a cursor, with which you can issue SQL statements. To obtain it:

```python
Cursor = Con.cursor()
```

5. To issue an SQL statement we first write it down. We need it as a string:

```python
sql = "SELECT * FROM judges"
```

Question: I see we’re working with a table called judges. What database are we in?

6. With that string we can invoke a special cursor method that makes things happen:

```python
Cursor.execute(sql)
```

This silently sends the query and collects the answer (from the database server) inside.

7. You can invoke a second magical method of the cursor to extract those results:

```python
Results = Cursor.fetchall()
```

8. The easiest thing to do at this point would be to just print it, see what you got.

```python
print Results
```
9. It’s always thoughtful to close the connection to the database (like hanging up):

```python
Con.close()
```

It also contained the task(s) for the day:

Now this is what we need to do:

- let’s get in our database server as root and create a new database
- let’s give to our already existing regular user access to it
- let’s get out as root and come back as the regular user
- let’s select the database, create a table, put some data in it
- let’s get out, then go to cgi-bin and create a Python script that extracts and prints the data from the table we just created

We decided to work in teams. The teams had 2-3 minutes to think and then they had to each ask one question, to get us started. I was supposed to answer the question. In the first team Andrew and Grant delegated Dmitriy to ask about step 6. As we discussed what the answer would entail, in the second team Karys and Linwood delegated Navin to ask “What is a cursor?” to which we answered: “A broker.” This emphasizes the role it plays, as an intermediary – establishing or enabling communication between a script and the database server (this was Grant’s answer). Finally the third team asked about step 4 above. It looked clear that we need to read carefully all the steps and start discussing and understanding them all. So we organized ourselves as follows:
Overall task: Create a database and make it available on the web.

Progression of events (plan, sequence of necessary actions):

Team 1:
Log into silo, log into mysql, create a table in a database (homeworkThree)

Team 2:
Puts data in table, starts a CGI script that says "Howdy!" and tests it.

Team 3:
Make Python script print result of query select * from ... in database set by Team 1.

Grant came from team 1. Turns out his MySQL wasn't running and wasn't even completely set up (he said). So we had to check it and performed the following steps:

(a) worked on his step008 script:

```
/nobackup/gchouck/mysql/bin/mysqld_safe \
  --user=gchouck \
  --pid-file=/nobackup/gchouck/mysql/mysqld.pid \
  --log=/nobackup/gchouck/mysql/mysqld.log \
  --socket=/nobackup/gchouck/mysql/mysql.sock \
  --basedir=/nobackup/gchouck/mysql \
  --log-error=/nobackup/gchouck/mysqld-error.log \
  --datadir=/nobackup/gchouck/mysql \
  --port=17212 &
```
(b) started his server then went in by running `connect_as_root`:

```bash
call $-socket=/nobackup/gchouck/mysql/mysql.sock \
    --port=17212 --u root -p
```

(c) as root created new user and database, granted permission for user to that database:

```bash
mysql> create database homeworkThree;
mysql> create user 'gchouck'@'silo.cs.indiana.edu' IDENTIFIED BY 's0mething';
mysql> grant all on homeworkThree.* to 'gchouck'@'silo.cs.indiana.edu';
```

(d) then went out and created a `connect_as_gchouck` script

```bash
call $-socket=/nobackup/gchouck/mysql/mysql.sock \
    --port=17212 --host=silo.cs.indiana.edu --u gchouck -p
```

(e) we used it to connect as `gchouck'@'silo.cs.indiana.edu'` and create a table

```sql
create table example {
    username varchar(8) primary key,
    name varchar(45)
}
```

(f) we populated it with some data, and verified that

```bash
mysql> show tables;
+-------------------------+
| Tables_in_homeworkThree |
+-------------------------+
| example                 |
+-------------------------+
1 row in set (0.00 sec)
mysql> describe example;
+----------+-------------------+--------+-------+----------+---------+
| Field    | Type              | Null   | Key   | Default  | Extra   |
+----------+-------------------+--------+-------+----------+---------+
| username | varchar(8)        | NO     | PRI   |          |         |
| name     | varchar(45)       | YES    | NULL  | NULL     |         |
+----------+-------------------+--------+-------+----------+---------+
2 rows in set (0.00 sec)

mysql> select * from example;
Empty set (0.00 sec)

mysql> insert into example
-> (username, name)
-> values
-> ('kbryant', 'Kobe Bryant'),
-> ('mjordan', 'Michael Jordan'),
-> ('mjackson', 'Michael Jackson')
-> ;
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from example;
+----------+--------+
| username | name   |
+----------+--------+
| kbryant  | Kobe Bryant |
| mjordan  | Michael Jordan |
| mjackson | Michael Jackson |
+----------+--------+
3 rows in set (0.00 sec)

mysql> exit
Bye
(g) wrote a script to do the same (extract all data from the table). We started with:

```
#!/usr/bin/python
print "Content-type: text/html\n\n"
print "Howdy!"
```

We made it executable, tested it, then modified it to:

```
#!/usr/bin/python
import MySQLdb
Con = MySQLdb.Connect(host="silo.cs.indiana.edu", port=17212, user="gchouck", passwd="s0mething", db="homeworkThree")
print "Howdy!"
```

We tried this from the prompt and on-line and not seeing an error we concluded it was making the connection.

```
#!/usr/bin/python
import MySQLdb
```

Con = MySQLdb.Connect(host="silo.cs.indiana.edu", \
    port=17212, user="gchouck", \n    passwd="s0mething", db="homeworkThree")

Cursor = Con.cursor()

cursor.execute("select * from example")

print Cursor.fetchall()

print "Howdy!"

(h) checked script online, it works. From the command line/prompt it works like this:

[gchouck@silo cgi-bin]$ ./what
Content-type: text/html

(("kbryant", 'Kobe Bryant'), ('mjordan', 'Michael Jordan'), ('mjackson', 'Michael Jackson'))
Howdy!

(i) went back in and added one more record

[gchouck@silo cgi-bin]$ cd /nobackup/gchouck/mysql-5.0.22/
[gchouck@silo mysql-5.0.22]$ ./connect_as_gchouck
Enter password: *********
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.0.22-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql> use homeworkThree
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> insert into example values ('lbird', 'Larry Bird');
Query OK, 1 row affected (0.00 sec)

mysql> exit
Bye
[gchouck@silo mysql-5.0.22]$  

(j) then we went back online and we verified that the new record shows.

For Wednesday we have the following reading assignment:

- What does the following program do?
- How does it do it?
- Do you see any of the above in it?

The code is somewhat annotated below. Wednesday try to come ready to discuss this code:

#!/usr/bin/python
import cgi, random, MySQLdb, sys, os
q = cgi.FieldStorage()

(message, session_id, balance, action, id) = ("", ",", ",", ",")

if q.has_key("action"):
    action = q["action"].value
    print "Content-type: text/html\n\n"
Con = MySQLdb.Connect(host="silo.cs.indiana.edu", \
port=8974, \
user="lbird", \
passwd="dribl", \
db="awards")

Cursor = Con.cursor()

if q.has_key("session_id") and action != "Reset":
    id = q["session_id"].value
    Cursor.execute("SELECT message, balance \
                   FROM example \
                   WHERE session_id = '%s' \
                   " % q["session_id"].value)
    Results = Cursor.fetchall()
    (message, balance) = Results[0]
    balance = str(int(balance) + 1)
    message = " You have clicked me %s times. " % balance
    Cursor.execute("update example \
                   set message = '%s', balance = '%s' \
                   where session_id = '%s' \
                   " % (message, balance, q["session_id"].value))
else:
    balance = 0
    message = "Welcome."
    id += str(random.randrange(10))
if not Cursor.execute("insert into example \
                     (session_id, balance, message) \
                     values \
                     ('%s', '%s', '%s') \
                     " % (id, balance, message)):
    print "Content-type: text/html\n\nError. Please reload."
    sys.exit()

print """"
<form>
  <input type=submit name=action value=Proceed> <input type=submit name=action value=Reset>
  <input type=hidden name=session_id value=%s>
</form>
"""" % (id, message)