Lab for Week 6

Mysql Server & Flask Session Variables

Chaufournier & Wood
CSCI 2541

Previously we would do something like this:

Import the sqlite library

Create a new connection to the database

Create a cursor. It’s a pointer to the database and tracks the location of operations.

The query we want to execute on the database.

Fetch results from the cursor

Close the connection to save memory

```
import sqlite3

conn = sqlite3.connect(‘example.db')


c = conn.cursor()


c.execute(‘‘‘Select * from users’’’)

results = c.fetchone()

conn.close()
```

With mysql connector: Select

Import the mysql.connector library

```
import mysql.connector
```

Create a new connection to the database

```
mydb = mysql.connector.connect(
     host="ec2-54-152-12-119.compute-1.amazonaws.com",
     user="student",
     password="seas",
     database="dev"
  )
```

Create a cursor. It’s a pointer to the database and tracks the location of operations.

```
c = mydb.cursor()
```

The query we want to execute on the database.

```
c.execute('''Select * from users''')
```

Fetch the results from the cursor

```
results = c.fetchone()
```

Close the cursor to save memory

```
c.close()
```

With mysql connector: Insert

Import the mysql.connector library

```
import mysql.connector
```

Create a new connection to the database

```
mydb = mysql.connector.connect(
    host="ec2-54-152-12-119.compute-1.amazonaws.com",
    user="student",
    password="seas",
    database="dev"
  )
```

Create a cursor. It's a pointer to the database and tracks the location of operations.

```
C = mydb.cursor()
```

The query we want to execute on the database.

```
c.execute('''insert into users(name)
values(?)''',["Lucas"])
```

Commit our changes to the database, so they are persisted.

```
mydb.commit()
```

Close the cursor to save memory

```
c.close()
```

Mysql Server and Flask

Import the library

```
from flask import Flask
import mysql.connector

app = Flask('app')
```

Establish a connection outside of the route

```
mydb = mysql.connector.connect(
    host="ec2-54-152-12-119.compute-1.amazonaws.com",
    user="student",
    password="seas",
    database="dev"
  )
```

Inside each route create a cursor and run your commands

Close the cursor before exiting

```
@app.route('/')
def hello_world():
  c= mydb.cursor()
  c.execute("select * from users");
  result = c.fetchone()
  c.close()
  return result[0]
```

SQLITE vs Mysql Server

SQLITE

```
from flask import Flask
import sqlite3

app = Flask('app')

mydb = sqlite3.connect('example.db')




@app.route('/')
def hello_world():
  c = mydb.cursor()
  c.execute("select * from users");
  result = c.fetchone()
  c.close()
  return result[0]
```

Mysql

```
from flask import Flask
import mysql.connector

app = Flask('app')

mydb = mysql.connector.connect(
    host="ec2-54-152-12-119.amazonaws.com",
    user="student",
    password="seas",
    database="dev"
  )

@app.route('/')
def hello_world():
  c= mydb.cursor()
  c.execute("select * from users");
  result = c.fetchone()
  c.close()
  return result[0]
```

Do I need to run a database query every time I need data?

Session Variables

A way to store data per client session

- Session data is stored on the server as opposed to cookies which store data on the client
- Session data is encrypted in a secure fashion on the server
- Session variables can be used to store information about a client, data used for authorization, etc
- Session variables allow us to maintain small amounts of data fetched from the database for use throughout a website.

Session Variables and Flask

Import the session package from Flask

Assign a predetermined secret key

Using a secret key insures that your session variables are encrypted. Never share this key.

```
from flask import Flask, session
import mysql.connector

app = Flask('app')
app.secret_key = b'mysecretkey'

mydb = mysql.connector.connect(...)
```

Session Variables and Flask

Use a session variable like a dictionary.

You can store key value pairs with details about the client.

```
@app.route('/')
def hello_world():
  cursor= mydb.cursor()
  cursor.execute("select * from users");
  result = cursor.fetchone()
  cursor.close()
  session['username'] =  result[1]
  return 'Username is %s.' % result[1]
```

Session Variables and Flask

Fetch details from session variable by using the same key

```
@app.route('/user')
def user_handler():
  return "Hello %s" % session['username']
```

Session Variables and Flask

A full flask application using session variables.

```
from flask import Flask, session
import mysql.connector

app = Flask('app')
app.secret_key = b'mysecretkey'

mydb = mysql.connector.connect(...)

@app.route('/')
def hello_world():
  cursor= mydb.cursor()
  cursor.execute("select * from users");
  result = cursor.fetchone()
  cursor.close()
  session['username'] =  result[1]
  return 'Username is %s.' % result[1]

@app.route('/user')
def user_handler():
  return "Hello %s" % session['username']

app.run(host='0.0.0.0', port=8080)
```

Activity 1

Extending the login application

- Extend your login application lab 5 to connect to your aws mysql server
- Add a session variable that stores the username
- Add a third route that returns a welcome message to the user with their username from the session variable.
- Hint1: You will have to use a create sql query that creates the table from lab 5 in your database.
- Hint 2: We will be extending this activity in a later homework, make sure you understand all the parts involved.

Create a repository for your group here:
`http://bit.ly/DB21-6`