Piruz Alemi Jan 10th, 2020
Columbia University

A brief Note on SQLALCHEMY and the steps taken in the SQL_Alchemy_Challenge Repository follows:

In This first method, we read/reflect the database onto ORM, Create and Engine, Inspect the data base and Knowing the structure of the data base, establish a session, connect, and querry the data base. Given the retrieved data, we can convert the data into Pandas_Python Data Frames for further analysis or Using Functions conduct the Analysis. The necessary elements in the library follows:

1. import sqlalchemy from sqlalchemy ext.**automap**
2. import automap_base from sqlalchemy.**orm**
3. import Session from sqlalchemy import **create_engine**, **inspect, func**

Use the Base class to reflect the database tables

Base.prepare(engine, reflect=True)
"Views" all of the classes mapped to the Base via **Base.classes.keys()**
Then establishes a session to the Base to Querry the Base Tables

In this method we explore the database SQLITE, first thru **Inspect**. Based on the latter inspection, we "Reflect" and Query the database directly, through first establishing a Session connection and then Querying on the basis of Session.
Obviously Query is under the domain of Session

In step 2: I directed the terminal to where the Flask program resided.
In this case directory sqlalchemy_challenge. And gave the following command:
>>> Python AlemiFlask.py
Which gave me back:

```
README.md - Jupyter Text Editor.html.mhtml
README.md.mhtml
lemibio.html
lemiprcp.png
pp3.py
limate_alemi_flask.ipynb
awaii.sqlite
awaii_measurements.csv
awaii_stations.csv
ndex_Brackett.html
base) dyn-129-236-218-27:SQL_ALCHEMY_Flask_Challenge piruzalemi$ python AlemiFl
sk.py
* Serving Flask app "AlemiFlask" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployme
t.
  Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 153-827-043
```

Activating my Python program under a self-server IP Address I then introduced Flask and its Routers, creating various APIs

```
['measurement', 'station']
 * Debugger is active!
 * Debugger PIN: 153-827-043
 * Detected change in '/Users/piruzalemi/AF.py', reloading
 * Restarting with stat
['measurement', 'station']
 * Debugger is active!
 * Debugger PIN: 153-827-043
 * Detected change in '/Users/piruzalemi/AF.py', reloading
 * Restarting with stat
['measurement', 'station']
 * Debugger is active!
 * Debugger PIN: 153-827-043
 * Detected change in '/Users/piruzalemi/AF.py', reloading
 * Restarting with stat
['measurement', 'station']
 * Debugger is active!
 * Debugger PIN: 153-827-043
 * Detected change in '/Users/piruzalemi/AF.py', reloading
 * Restarting with stat
['measurement', 'station']
 * Debugger is active!
 * Debugger PIN: 153-827-043
```

The program was executed thru the command >>>Python AF.PY
While the debugger is active, we could test each modification to the program, interfacing between the program (Client) And the Server (APIs) that would establish connections, sessions and queries to the SQLITE data base.

A very rewarding process, that tests the ability of a programmer on both APIs, Python, Flask, Terminal commands, data visualization, debugging and more. Thank you. Piruz Alemi, Jan 16th, 2020.