Week 11 SQLAlchemy: Day 2

What you will learn

- Use the SQLAlchemy ORM to create classes that model tables.
- Perform database CRUD operations using the SQLAlchemy ORM.
- Reflect existing databases.
- Use the SQLAlchemy Inspector to view table names in the database.
- Plot the query results from the ORM

Instructor Activity: SQLAlchemy Queries

1. Basic Querying[1]
    a. 1000 rows
    b. **from sqlalchemy.orm import** Session
        i. session.query()
        ii. SQLAlchemy Conjuctions or_() and_() and not_()[2]

Students Activity: Shark.sql

***Instructions**:*

- *Create a new table within your database called "SharkSearch" and run the SQL code provided within SQLPro or MySQL Workbench*
- *Within a Python script, create a "Sharks" class that will be able to read all of the columns in from the table you created*
- *Using SQLAlchemy, perform the following queries...*

[1]https://github.com/coding-boot-camp/DataViz-Lesson-Plans/blob/master/01-Lesson-Plans/11-Advanced-Data-Storage-and-Retrieval/2/Activities/01-Ins_Basic_Querying/Solved/Ins_Basic_Querying.ipynb
[2]http://docs.sqlalchemy.org/en/latest/core/tutorial.html

- *Print all locations of shark attacks*
- *Find the number of provoked attacks*
- *Find the number of attacks in the USA*
- *Find the number of attacks in 2017*
- *Find the number of attacks while surfing*
- *Find the number of fatal shark attacks in 2017 in Australia*

Instructor Activity: Updating and Deleting Rows

Basic Updating: .first() .delete() .session.commit()

Partners Up: What a Cruddy Database

Instructions:

- Within a Python file, create new SQLAlchemy class called Garbage that holds the following values...
  - __tablename__: Should be "garbage_collection"
  - id: The primary key for the table that is an integer and automatically increments
  - item: A string that describes what kind of item was collected
  - weight: A double that explains how heavy the item is
  - collector: A string that lets users know which garbage man collected the item
- Create a connection and a session before adding a few items into the SQLite database crafte d.
- Update the values within at least two of the rows added to the table.
- Delete the row with the lowest weight from the table.
- Print out all of the data within the database.

Bonus:

- Modify the application so that items can be added, updated, queried, or removed according to user inputs.

BREAK

Instructor Activity: Reflections

- automap_base
- Then, engine = create_engine("")
- Next, create a Base = automap_base()
- Finally, call Base.prepare(engine, reflect=True)

Students Activity: Reflecting on SQL

Instructions:

- Create engine using the demographics.sqlite database file
- Declare a Base using automap_base() and use this new Base class to reflect the database's tables
- Assign the demographics table/class to a variable called Demographics
- Create a session and use this session to query the Demographics table and display the first five locations

Bonus:

- Query and print the number of unique locations in the table.

Hint:

- For the bonus, look into counting and grouping operations in SQLAlchemy

Instructor Activity: SQLAlchemy Exploration

Exploration:

- from **sqlalchemy** import **create_engine**, **inspect**
- inspect(engine)
- inspector.get_table_names()
- inspector.get_columns(<Table Name>)

Students Activity: Salary Exploration

Instructions:

- Using the attached SQLite file, use an inspector to collect the following information...
- The names of all of the tables within the database.
- The column names and data types for the Salaries table.
- Reflect the database, create a session, and query the Salaries table to collect the number of salaries that are over 50k per year.

Group Activity: Emoji Plotting

Instructions:

- Use the inspector to explore the database and print out the table names stored within it.
- Using the inspector, print out the column names and types for each of the tables contained within the SQLite file.
- Reflect the database into a SQLAlchemy class and start a session that can be used to query the database.
- Using Matplotlib, create a horizontal bar chart and plot the emoji score in descending order. Use emoji_char as the y-axis labels and plot only the top 10 emojis ranked by score
- Create the same kind of chart using Pandas to plot the data instead of Matplotlib.

Homework Cheat Sheet

Review these libraries import datetime import numpy import pandas import sqlalchemy from sqlalchemy.ext.automap import automap_base from sqlalchemy.orm import Session from sqlalchemy import create_engine, func from flask import Flask, jsonify Base = automap_base() # Save references to each table using Base Python Flask[3] **Day 3 (From StudentGuide.md import Flask Mega-Tutorial Video[4])** app = Flask(__name__)

[3] http://flask.pocoo.org/

[4] https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world