

Advanced SQL – Challenge 10

Jeremy Hooper

Introduction

I chose to holiday in Hawaii next year. To assist with my trip planning, I learned more about the climate by exploring and analysing Hawaiian climate data using Python, SQLAlchemy, and Matplotlib and designing a climate app using Python and Flask.

Part 1 – Analysis and Exploration of Hawaiian Climate Data

In this section, I used Python and SQLALCHEMY to perform a basic climate analysis and data exploration of the provided climate database. I used SQLAlchemy ORM queries, Pandas, and Matplotlib. I did the work in Jupyter Notebook.

- **Precipitation Analysis:**
 - I identified the most recent date in the dataset as 23 August 2017.
 - I extracted precipitation data from the recent date for the previous 12 months, focusing on the "date" and "prcp" values.
 - I loaded the retrieved data into a Pandas DataFrame and explicitly set column names. The DataFrame was sorted by date.
 - I visualised precipitation data using a line plot, showcasing the variations in precipitation over the selected period.
 - I computed and presented summary statistics for the precipitation data, including mean, median, and quartiles.
- **Station Analysis:**
 - I designed a query to calculate the total number of stations in the dataset, providing an overview of the available weather stations.
 - I ranked stations by observation counts in descending order, revealing the station with the highest number of observations as station USC00519281.
- **Temperature Analysis:**
 - For the most active station, I calculated the lowest, highest, and average temperatures.
 - I filtered data for the previous 12 months (TOBS) for the most active station and visualised it as a histogram with 12 bins, offering insights into temperature distribution.

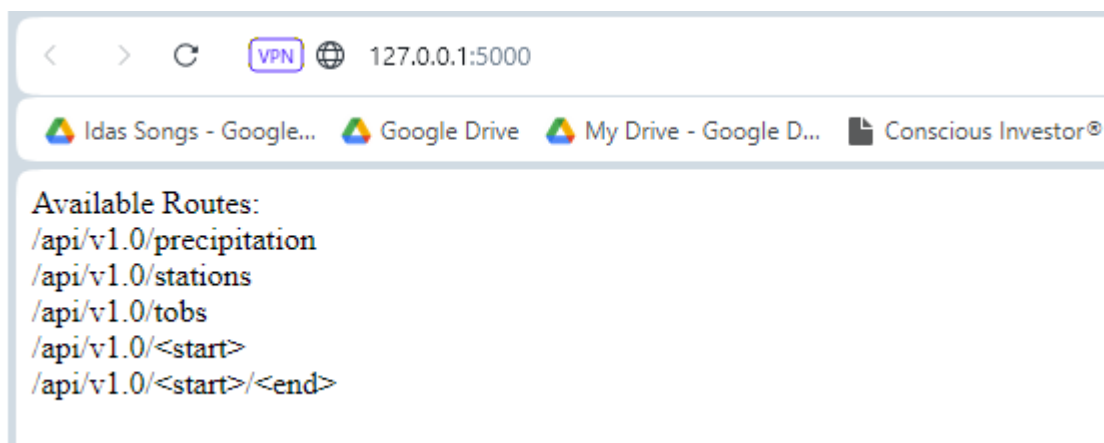
Part 2 – Design a Climate App

I designed a climate app using the instructions given in the Challenge instructions. Below are partial screenshots of each stage of the app.

I ran the program in Anaconda's Powershell Prompt window. When the program starts, the screen below appears (Figure 1), and the Chrome screen below shows the available routes (Figure 2).

```
(base) PS C:\Users\jerry\UWABootcamp\adv_sql-challenge-10> cd adv_sql-challenge10
(base) PS C:\Users\jerry\UWABootcamp\adv_sql-challenge-10\adv_sql-challenge10> python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with watchdog (windowsapi)
* Debugger is active!
* Debugger PIN: 119-840-502
```

1. Running app.py in Powershell Prompt. Running on <http://127.0.0.1:5000>



2. Display on URL [HTTP://127.0.0.1:5000](http://127.0.0.1:5000) – Display of available routes.

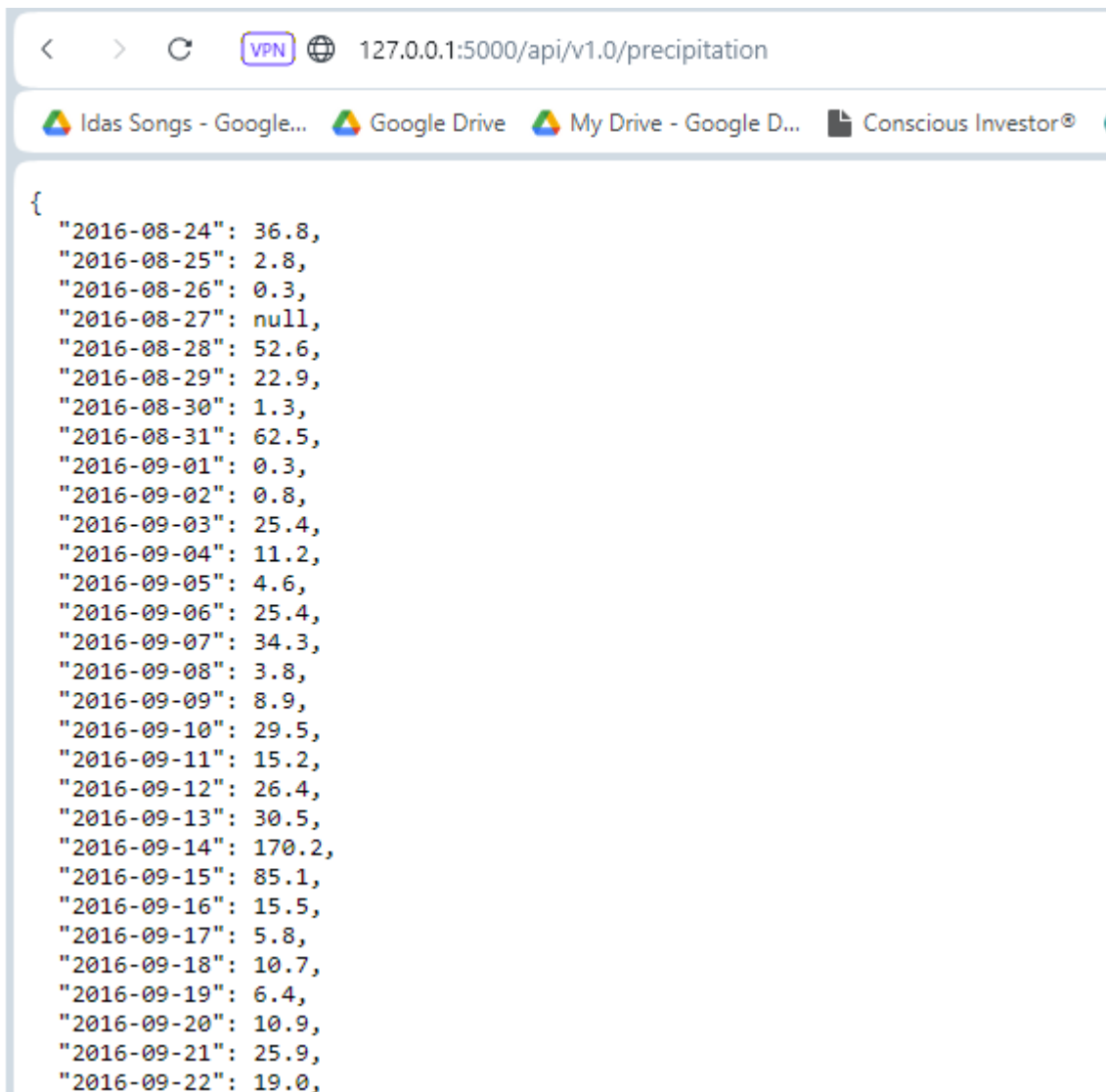
On completing the URL for each route, we get the appropriate outputs.

[HTTP://127.0.0.1:5000/api/v1.0/precipitation](http://127.0.0.1:5000/api/v1.0/precipitation) outputs the JSON representation of the dictionary created by the data and precipitation values for the last year of the year of readings for the most active station USC 00519281.

[HTTP://127.0.0.1:5000/api/v1.0/stations](http://127.0.0.1:5000/api/v1.0/stations) outputs the JSON list of stations from the data set.

[HTTP://127.0.0.1:5000/api/v1.0/tobs](http://127.0.0.1:5000/api/v1.0/tobs) outputs the JSON list of temperature observations for the last year of operations.

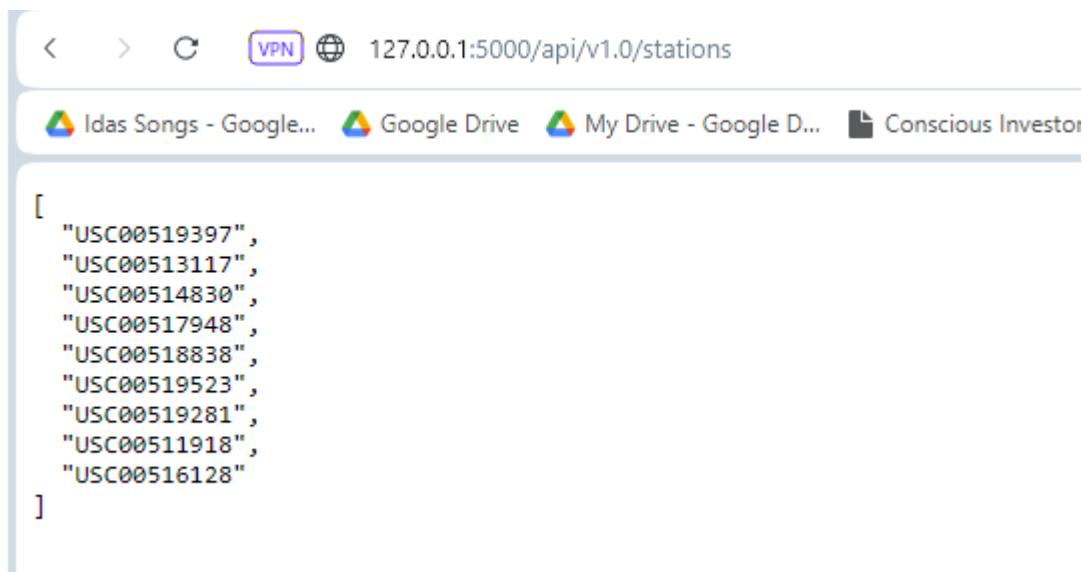
The last two items output the minimum, average, and maximum temperatures for selected date ranges.



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:5000/api/v1.0/precipitation`. The browser tabs include "Idas Songs - Google...", "Google Drive", "My Drive - Google D...", and "Conscious Investor®". The main content area displays a JSON object with precipitation data for various dates from 2016-08-24 to 2016-09-22. The data is as follows:

```
{
  "2016-08-24": 36.8,
  "2016-08-25": 2.8,
  "2016-08-26": 0.3,
  "2016-08-27": null,
  "2016-08-28": 52.6,
  "2016-08-29": 22.9,
  "2016-08-30": 1.3,
  "2016-08-31": 62.5,
  "2016-09-01": 0.3,
  "2016-09-02": 0.8,
  "2016-09-03": 25.4,
  "2016-09-04": 11.2,
  "2016-09-05": 4.6,
  "2016-09-06": 25.4,
  "2016-09-07": 34.3,
  "2016-09-08": 3.8,
  "2016-09-09": 8.9,
  "2016-09-10": 29.5,
  "2016-09-11": 15.2,
  "2016-09-12": 26.4,
  "2016-09-13": 30.5,
  "2016-09-14": 170.2,
  "2016-09-15": 85.1,
  "2016-09-16": 15.5,
  "2016-09-17": 5.8,
  "2016-09-18": 10.7,
  "2016-09-19": 6.4,
  "2016-09-20": 10.9,
  "2016-09-21": 25.9,
  "2016-09-22": 19.0,
```

3. [HTTP://127.0.0.1:5000/api/v1.0/precipitation](http://127.0.0.1:5000/api/v1.0/precipitation) - showing dates and precipitation between 24/08/2016 and 22/09/2016 (complete output to 23/08/2017).



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:5000/api/v1.0/stations`. The browser tabs are the same as in the previous screenshot. The main content area displays a JSON array of station IDs:

```
[
  "USC00519397",
  "USC00513117",
  "USC00514830",
  "USC00517948",
  "USC00518838",
  "USC00519523",
  "USC00519281",
  "USC00511918",
  "USC00516128"
]
```

4. [HTTP://127.0.0.1:5000/api/v1.0/stations](http://127.0.0.1:5000/api/v1.0/stations) - list of stations

The screenshot shows a web browser with the address bar displaying `127.0.0.1:5000/api/v1.0/tobs`. The browser tabs include "Idas Songs - Google...", "Google Drive", "My Drive - Google D...", and "Conscious Inv". The main content area displays a JSON array of temperature observations:

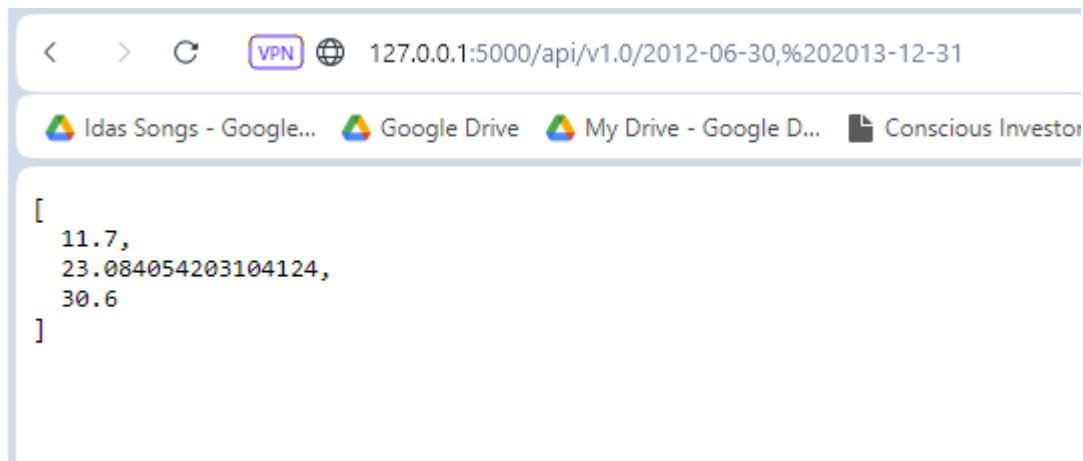
```
[
  25.0,
  26.7,
  26.7,
  23.9,
  22.8,
  25.6,
  25.0,
  25.6,
  26.7,
  26.7,
  25.6,
  25.6,
  25.6,
  25.6,
  22.8,
  23.3,
  26.7,
  26.1,
  25.0,
  26.7,
  24.4,
  26.1,
  23.9,
  26.1,
  25.6,
  26.1,
  25.6,
  25.6
```

5. [HTTP://127.0.0.1:5000/api/v1.0/tobs](http://127.0.0.1:5000/api/v1.0/tobs) - list of temperature observations for the last year.

The screenshot shows a web browser with the address bar displaying `127.0.0.1:5000/api/v1.0/2016-06-30`. The browser tabs are the same as in the previous screenshot. The main content area displays a JSON array with three values:

```
[
  14.4,
  23.89376218323582,
  30.6
]
```

5. [HTTP://127.0.0.1:5000/api/v1.0/<start>](http://127.0.0.1:5000/api/v1.0/<start>) - Minimum temperature, Average temperature, Maximum temperature on 30/06/20-16.



5. [HTTP://127.0.0.1:5000/api/v1.0/<start>](http://127.0.0.1:5000/api/v1.0/<start>) - Minimum temperature, Average temperature, Maximum temperature on between 30/06/2012 and 31/12/2013.

Conclusion

This climate analysis I conducted provides valuable insights from trip planning. I now clearly understand precipitation trends and temperature observations in the region. The identified station with the most data will be a valuable resource for tracking weather patterns during my vacation.