Climate App- Flask

By: Suvrangshu Ghosh

Step 1 - Climate Analysis and Exploration:

Code - Suvclimate_starter.ipynb

Precipitation Analysis

Design a query to retrieve the last 12 months of precipitation data.

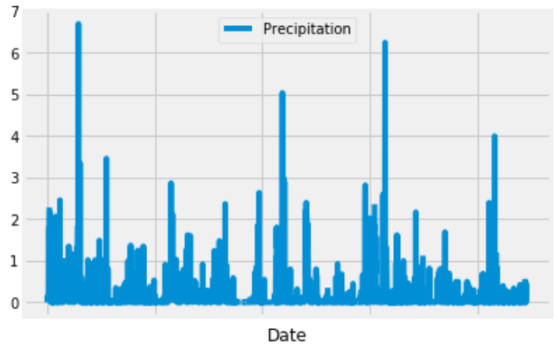
Select only the date and prcp values.

Load the query results into a Pandas DataFrame and set the index to the date column.

Sort the DataFrame values by date.

Plot the results using the DataFrame plot method.

```
2017-08-23 00:00:00
2016-08-22 00:00:00
```



In [14]:
```
# Use Pandas to calcualte the summary statistics for the precipitation data
df = pd.DataFrame(prcp1.Precipitation.describe())
df
```

Out[14]:

	Precipitation
count	2021.000000
mean	0.177279
std	0.461190
min	0.000000
25%	0.000000
50%	0.020000
75%	0.130000
max	6.700000

What are the most active stations? (i.e. what stations have the most rows)?

List the stations and the counts in descending order.

Out[18]:

	Station	Station Name	Count
0	USC00519281	WAIHEE 837.5, HI US	2772
1	USC00519397	WAIKIKI 717.2, HI US	2724
2	USC00513117	KANEOHE 838.1, HI US	2709
3	USC00519523	WAIMANALO EXPERIMENTAL FARM, HI US	2669
4	USC00516128	MANOA LYON ARBO 785.2, HI US	2612
5	USC00514830	KUALOA RANCH HEADQUARTERS 886.9, HI US	2202
6	USC00511918	HONOLULU OBSERVATORY 702.2, HI US	1979
7	USC00517948	PEARL CITY, HI US	1372
8	USC00518838	UPPER WAHIAWA 874.3, HI US	511

Using the station id from the previous query, calculate the lowest temperature recorded,

highest temperature recorded, and average temperature most active station?

In [19]:

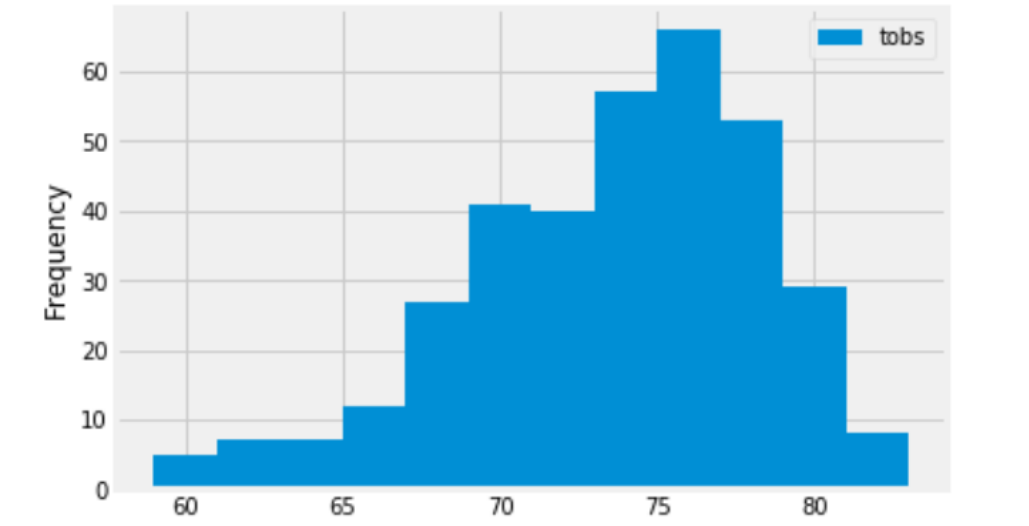
```
# Using the station id from the previous query, calculate the lowest temperature recorded,
# highest temperature recorded, and average temperature most active station?
temp_result = session.query(Measurement.station, func.min(Measurement.tobs), func.max(Measurement.tobs), func.avg(
        filter(Measurement.station == active_stations[0][0]).all()

temp_resultpd = pd.DataFrame(temp_result, columns=["StID", "MinTemp", "MaxTemp", "AvgTemp"])
temp_resultpd
```

Out[19]:

	StID	MinTemp	MaxTemp	AvgTemp
0	USC00519281	54.0	85.0	71.663781

Choose the station with the highest number of temperature observations.

Query the last 12 months of temperature observation data for this station and plot the results as a histogram



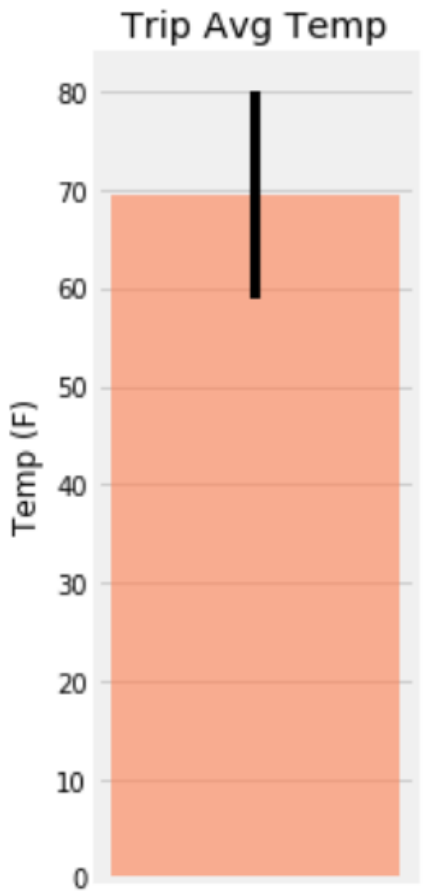
Plot the results from your previous query as a bar chart.

Use "Trip Avg Temp" as your Title

Use the average temperature for the y value

Use the peak-to-peak (tmax-tmin) value as the y error bar (yerr)

```
Out[23]: Text(0,0.5,'Temp (F)')
```



Calculate the rainfall per weather station for your trip dates using the previous year's matching dates.

Sort this in descending order by precipitation amount and list the station, name, latitude, longitude,

and elevation

trip_start = '2017-01-01'

trip_end = '2017-01-20'

:

	Station	Station Name	Latitude	Longitude	Elevation	PCount
0	USC00516128	MANOA LYON ARBO 785.2, HI US	21.33310	-157.80250	152.4	0.71
1	USC00514830	KUALOA RANCH HEADQUARTERS 886.9, HI US	21.52130	-157.83740	7.0	0.63
2	USC00519523	WAIMANALO EXPERIMENTAL FARM, HI US	21.33556	-157.71139	19.5	0.61
3	USC00513117	KANEOHE 838.1, HI US	21.42340	-157.80150	14.6	0.35
4	USC00519281	WAIHEE 837.5, HI US	21.45167	-157.84889	32.9	0.23
5	USC00517948	PEARL CITY, HI US	21.39340	-157.97510	11.9	0.00
6	USC00519397	WAIKIKI 717.2, HI US	21.27160	-157.81680	3.0	0.00

Optional Challenge Assignment

calculate the daily normals for your trip

push each tuple of calculations into a list called `normals`

Set the start and end date of the trip

Use the start and end date to create a range of dates

Stip off the year and save a list of %m-%d strings

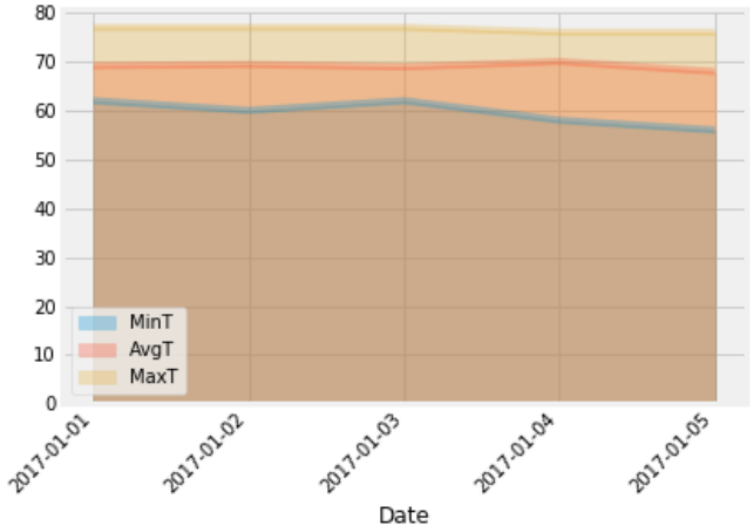
Loop through the list of %m-%d strings and calculate the normals for each date

Plot the daily normals as an area plot with `stacked=False`

trip_start = '2017-01-01'

trip_end = '2017-01-05'

	MinT	AvgT	MaxT
0	62.0	69.153846	77.0
1	60.0	69.396226	77.0
2	62.0	68.909091	77.0
3	58.0	70.000000	76.0
4	56.0	67.964286	76.0



Load the previous query results into a Pandas DataFrame and add the `trip_dates` range as the `date` index

#joinging two data frames

Out[27]:

Date	MinT	AvgT	MaxT
2017-01-01	62.0	69.153846	77.0
2017-01-02	60.0	69.396226	77.0
2017-01-03	62.0	68.909091	77.0
2017-01-04	58.0	70.000000	76.0
2017-01-05	56.0	67.964286	76.0

Step 2 - Climate App:

I have created the second part of the flask in two ways: **Easy Way**, **Tedious Way**.

But first execute code app.py at command line:

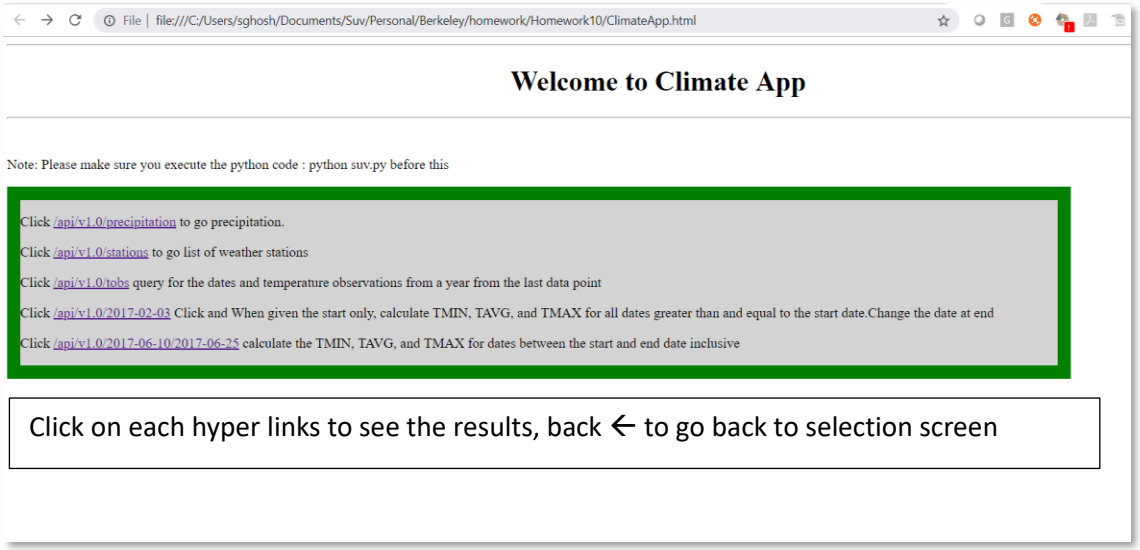
```
$ python suv.py
 * Serving Flask app "suv" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 167-151-802
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Easy Way:

I created a html file called ClimateApp.html. The ClimateApp.html displays all the hyperlinks to the flask and calls the app.py.

From the folder, please right click on ClimateApp.html and select open with Chrome

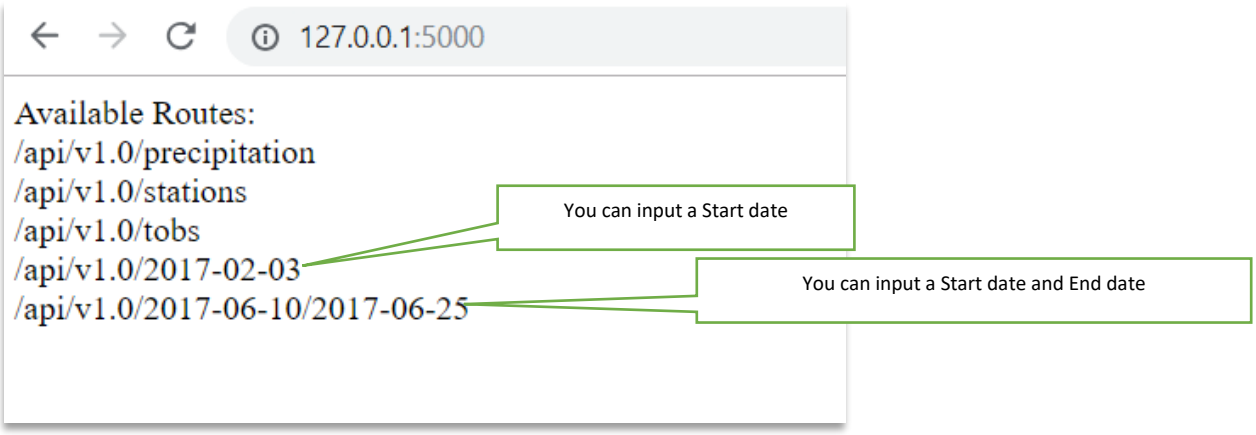
You will get this screen:



Tedious way:

You can ignore the ClimateApp.html and execute app.py at command line and put the url:

http://127.0.0.1:5000/ at the browser to get the screen below:



Copy each selection and paste next to the url like shown below:

http://127.0.0.1:5000/api/v1.0/precipitation