SQL Alchemy Project

November 21, 2021

[1]:
```
%matplotlib inline
from matplotlib import style
style.use('fivethirtyeight')
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
import numpy as np
import pandas as pd
```

[3]:
```
import datetime as dt
```

1 Reflect Tables into SQLAlchemy ORM

[4]:
```
# Python SQL toolkit and Object Relational Mapper
import sqlalchemy
from sqlalchemy.ext.automap import automap_base
from sqlalchemy.orm import Session
from sqlalchemy import create_engine, func
```

[5]:
```
engine = create_engine("sqlite:///Resources/hawaii.sqlite")
```

[6]:
```
Base = automap_base()
Base.prepare(engine, reflect=True)
```

[7]:
```
Base.classes.keys()
```

[7]:
```
['measurement', 'station']
```

[8]:
```
Measurement = Base.classes.measurement
Station = Base.classes.station
# print(session.query(Station))
# print(session.query(Measurement))
```

[9]:
```
session = Session(engine)
```

2 Exploratory Climate Analysis

[10]:

```python
recent = session.query(func.max(Measurement.date))
lastdate = dt.datetime.strptime(recent[0][0], '%Y-%m-%d')
firstdate = lastdate - dt.timedelta(days=365)
print(lastdate)
print(firstdate)
```

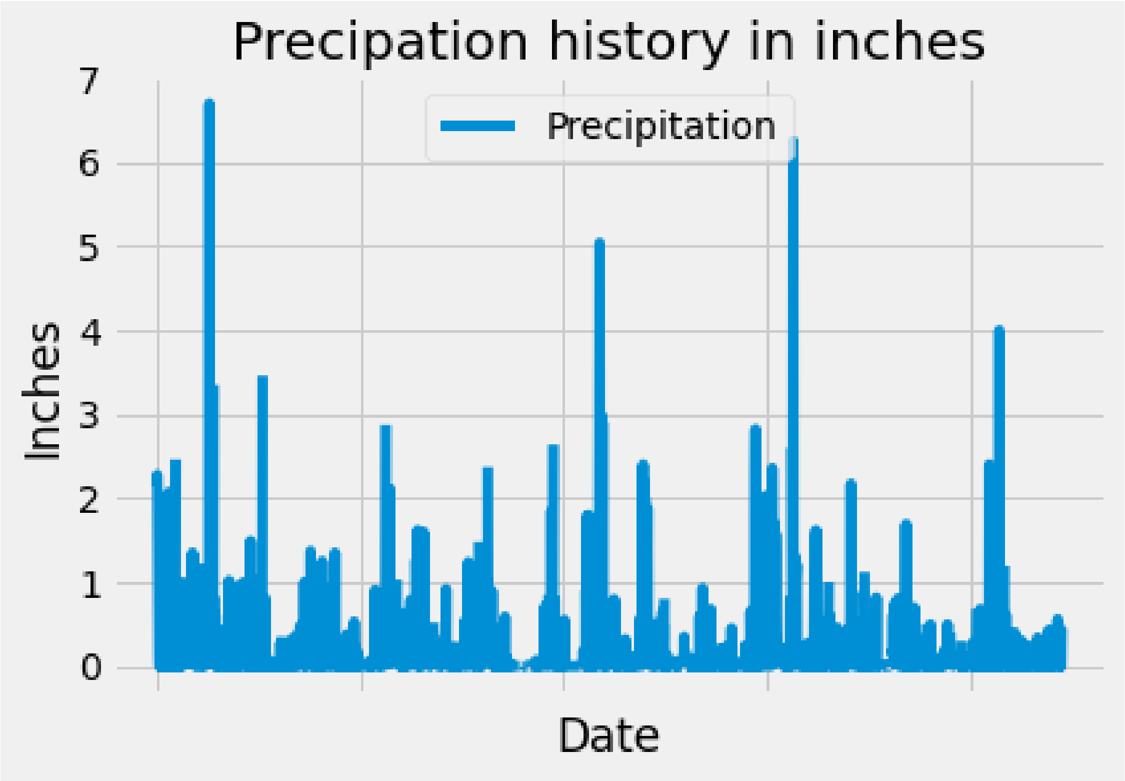
```
2017-08-23 00:00:00
2016-08-23 00:00:00
```

[25]:

```python
sel = [Measurement.date, Measurement.prcp]
results = session.query(*sel).filter(Measurement.date > firstdate).all()
#print(results)

df = pd.DataFrame(results, columns=['Date', 'Precipitation'])
df.set_index('Date', inplace=True)
df = df.sort_values('Date')


df.plot().set_xticklabels([])
plt.xlabel("Date")
plt.ylabel("Inches")
plt.title("Precipation history in inches")
plt.style.use('fivethirtyeight')
plt.legend(loc='upper center')
plt.show()
```



[12]:
```
df.describe()
```

[12]:
```
       Precipitation
count    2015.000000
mean        0.176462
std         0.460288
min         0.000000
25%         0.000000
50%         0.020000
75%         0.130000
max         6.700000
```

[13]:
```
# How many stations are available in this dataset?
sel1 = [Measurement.station, Measurement.prcp]
results1 = session.query(*sel1).group_by(Measurement.station).all()
print(f'There are {len(results1)} stations.')
```

```
There are 9 stations.
```

[14]:
```
# What are the most active stations?
sel2 = [Measurement.station, func.count(Measurement.station)]
results2 = session.query(*sel2).group_by(Measurement.station).order_by(func.
 ↪count(Measurement.station).desc()).all()
```

```
print(results2)
```

```
[('USC00519281', 2772), ('USC00519397', 2724), ('USC00513117', 2709),
('USC00519523', 2669), ('USC00516128', 2612), ('USC00514830', 2202),
('USC00511918', 1979), ('USC00517948', 1372), ('USC00518838', 511)]
```

[15]:

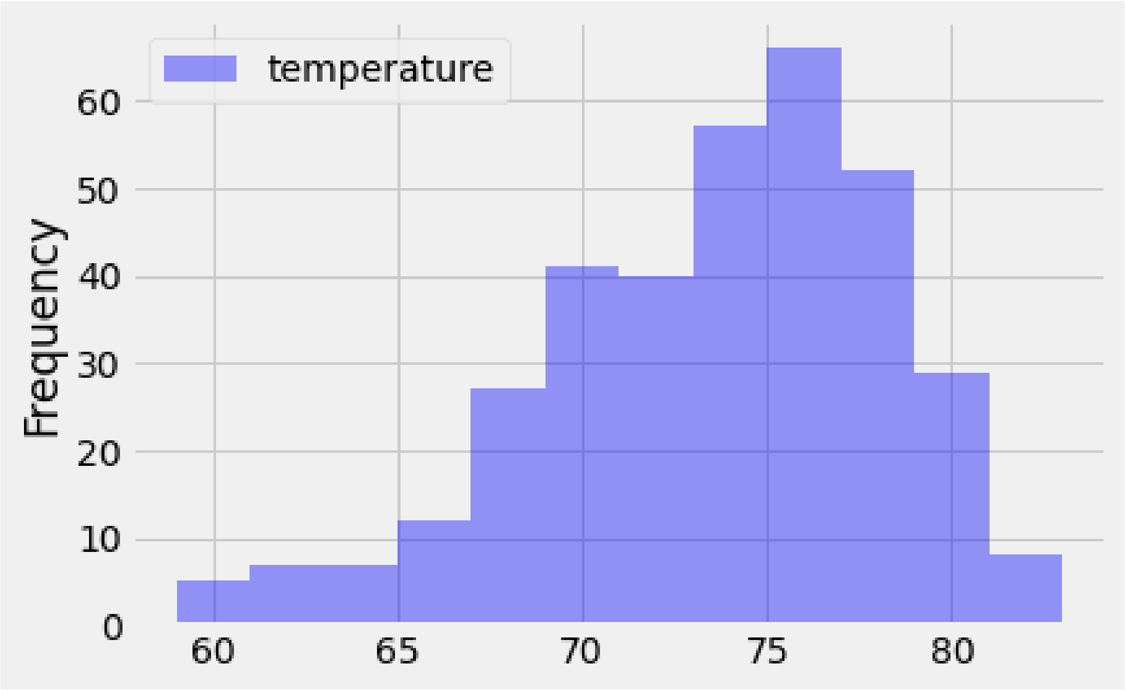
```
# Calculate the lowest temperature recorded, highest temperature recorded, and␣
 ↪average temperature of the most active station
maxstation = results2[0][0]
mintemp = session.query(func.min(Measurement.tobs)).filter(Measurement.station␣
 ↪== maxstation).first()
maxtemp = session.query(func.max(Measurement.tobs)).filter(Measurement.station␣
 ↪== maxstation).first()
avgtemp = session.query(func.avg(Measurement.tobs)).filter(Measurement.station␣
 ↪== maxstation).first()
print([mintemp[0], maxtemp[0], avgtemp[0]])
```

```
[54.0, 85.0, 71.66378066378067]
```

[16]:

```
yeartemp = session.query(Measurement.tobs).filter(Measurement.date > firstdate).
 ↪filter(Measurement.station == maxstation).all()
histogram = sns.distplot(yeartemp, bins=12, kde=False, label='temperature',␣
 ↪color='blue')
histogram.set(ylabel='Frequency')
plt.legend()
plt.show()
```

```
C:\Users\13306\Anaconda3\lib\site-packages\seaborn\distributions.py:2557:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `histplot` (an axes-level function for
histograms).
  warnings.warn(msg, FutureWarning)
```



[17]:
```
def calc_temps(start_date, end_date):
    return session.query(func.min(Measurement.tobs), func.avg(Measurement.
 ↪tobs), func.max(Measurement.tobs)).\
        filter(Measurement.date >= start_date).filter(Measurement.date <=␣
 ↪end_date).all()

print(calc_temps('2012-02-28', '2012-03-05'))
```

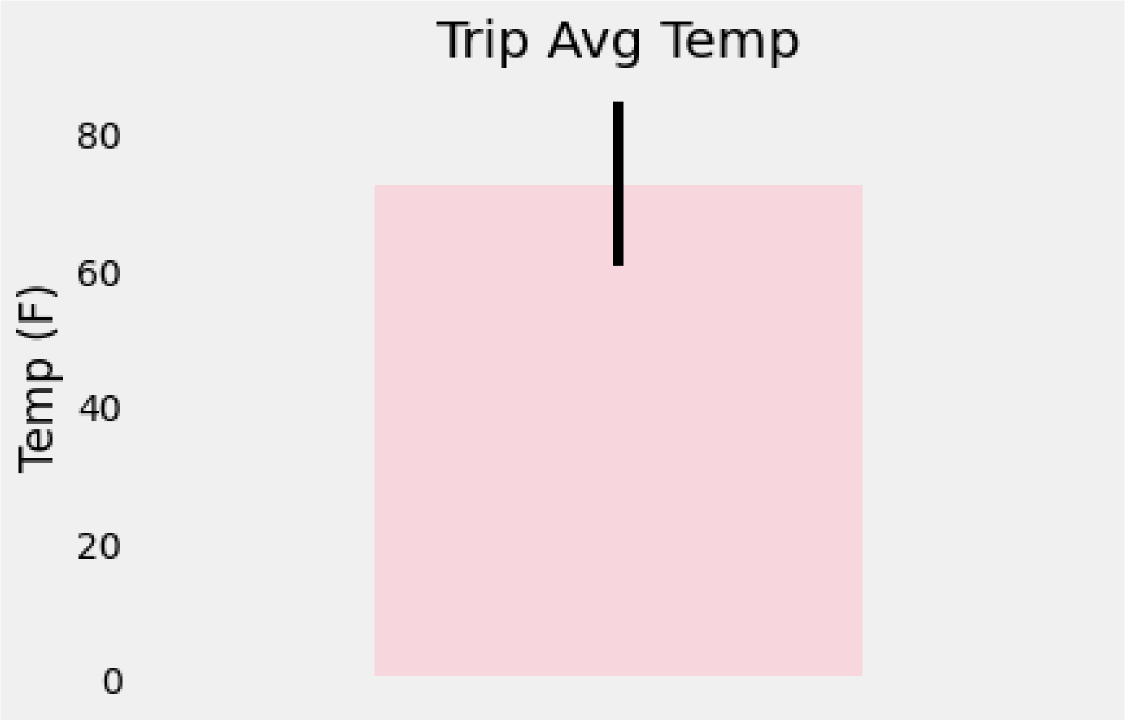
```
[(62.0, 69.57142857142857, 74.0)]
```

[18]:
```
# Calculate the tmin, tavg, and tmax for the trip using data from 3 years ago␣
 ↪for those same dates
tripstart = dt.date(2020,4,13)
tripend = dt.date(2020,4,17)
oldstart = tripstart - dt.timedelta(days=1096)
oldend = tripend - dt.timedelta(days=1096)
print(calc_temps(oldstart, oldend))
```

```
[(67.0, 72.83333333333333, 79.0)]
```

[19]:
```
tripdf = pd.DataFrame(calc_temps(oldstart, oldend), columns=['Min Temp', 'Avg␣
 ↪Temp', 'Max Temp'])
avgtemp = tripdf['Avg Temp']
peak = tripdf.iloc[0]['Max Temp'] - tripdf.iloc[0]['Min Temp']
avgtemp.plot(kind='bar', yerr=peak, alpha=0.5, color='pink')
```

```
plt.title("Trip Avg Temp")
plt.ylabel("Temp (F)")
plt.xticks([])
plt.grid()
plt.show()
```



[20]:

```
#Total amount of rainfall per weather station for the trip dates using the␣
 ↪previous year's matching dates.
finalsel = [Measurement.station, Station.name, func.sum(Measurement.prcp),␣
 ↪Station.latitude, Station.longitude, Station.elevation]
finalresults = session.query(*finalsel).filter(Measurement.station==Station.
 ↪station).filter(Measurement.date <= oldend).filter(Measurement.date >=␣
 ↪oldstart).group_by(Measurement.station).order_by(func.sum(Measurement.prcp).
 ↪desc()).all()
for result in finalresults:
    print(result)
```

```
('USC00516128', 'MANOA LYON ARBO 785.2, HI US', 5.359999999999999, 21.3331,
-157.8025, 152.4)
('USC00519281', 'WAIHEE 837.5, HI US', 4.779999999999999, 21.45167,
-157.84888999999998, 32.9)
('USC00513117', 'KANEOHE 838.1, HI US', 2.31, 21.4234, -157.8015, 14.6)
('USC00519523', 'WAIMANALO EXPERIMENTAL FARM, HI US', 0.6599999999999999,
21.33556, -157.71139, 19.5)
```

```
('USC00519397', 'WAIKIKI 717.2, HI US', 0.29000000000000004, 21.2716, -157.8168,
3.0)
('USC00514830', 'KUALOA RANCH HEADQUARTERS 886.9, HI US', 0.29, 21.5213,
-157.8374, 7.0)
('USC00517948', 'PEARL CITY, HI US', None, 21.3934, -157.9751, 11.9)
```

[21]:
```
# Function to calculate the daily normals
def daily_normals(date):
    """Daily Normals.
    Args:
        date (str): A date string in the format '%m-%d'
    Returns:
        A list of tuples containing the daily normals, tmin, tavg, and tmax
    """
    sel = [func.min(Measurement.tobs), func.avg(Measurement.tobs), func.
 ↪max(Measurement.tobs)]
    return session.query(*sel).filter(func.strftime("%m-%d", Measurement.date)␣
 ↪== date).all()

daily_normals("01-01")
```

[21]:
```
[(62.0, 69.15384615384616, 77.0)]
```

[22]:
```
# Calculate the daily normals for the trip

trip_start = '2018-01-01'
trip_end = '2018-01-07'

trip_dates = pd.date_range(trip_start, trip_end, freq='D')

trip_month_day = trip_dates.strftime('%m-%d')

normals = []
for date in trip_month_day:
    normals.append(*daily_normals(date))

normals
```

[22]:
```
[(62.0, 69.15384615384616, 77.0),
 (60.0, 69.39622641509433, 77.0),
 (62.0, 68.9090909090909, 77.0),
 (58.0, 70.0, 76.0),
 (56.0, 67.96428571428571, 76.0),
 (61.0, 68.96491228070175, 76.0),
 (57.0, 68.54385964912281, 76.0)]
```

[23]:

```
df = pd.DataFrame(normals, columns=['tmin', 'tavg', 'tmax'])
df['date'] = trip_dates
df.set_index(['date'],inplace=True)
df.head()
```

[23]:

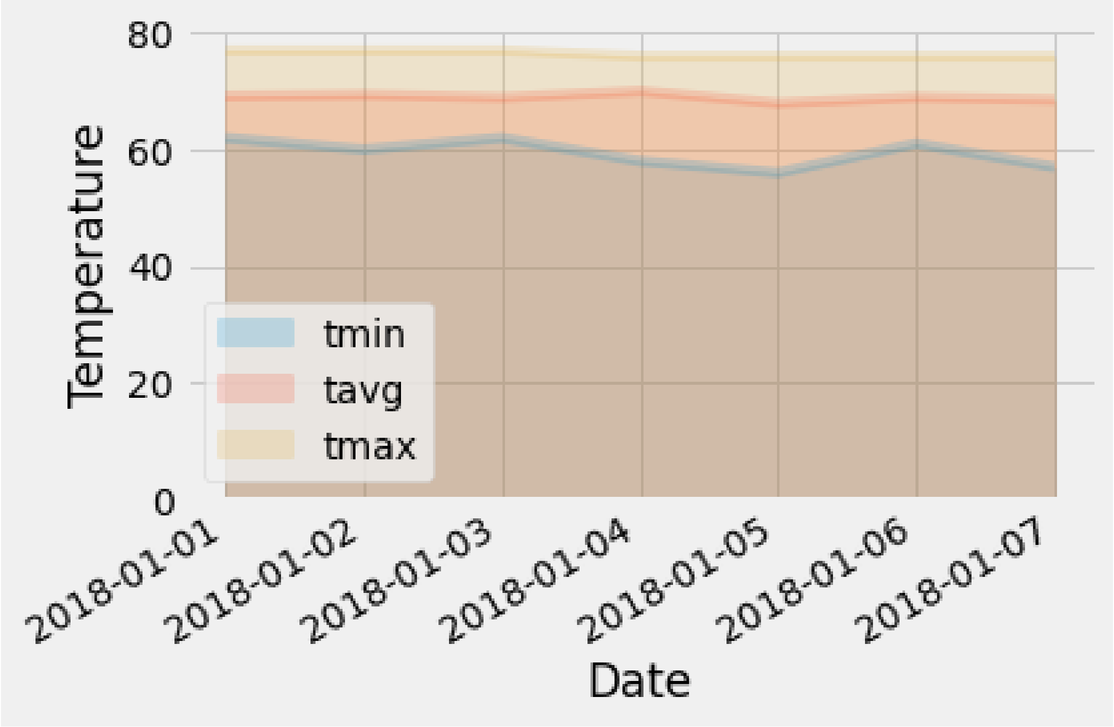
```
            tmin       tavg  tmax
date
2018-01-01  62.0  69.153846  77.0
2018-01-02  60.0  69.396226  77.0
2018-01-03  62.0  68.909091  77.0
2018-01-04  58.0  70.000000  76.0
2018-01-05  56.0  67.964286  76.0
```

[24]:

```
df.plot(kind='area', stacked=False, x_compat=True, alpha=.2)
plt.tight_layout()
plt.xlabel("Date")
plt.ylabel("Temperature")
```

[24]:

```
Text(9.310000000000002, 0.5, 'Temperature')
```



2.1 Stats

```
[27]: # "tobs" is "temperature observations"
      df = pd.read_csv('Resources/hawaii_measurements.csv')
      df.head()
```

```
[27]:        station        date  prcp  tobs
      0  USC00519397  2010-01-01  0.08    65
      1  USC00519397  2010-01-02  0.00    63
      2  USC00519397  2010-01-03  0.00    74
      3  USC00519397  2010-01-04  0.00    76
      4  USC00519397  2010-01-06   NaN    73
```

```
[28]: df.tail()
```

```
[28]:            station        date  prcp  tobs
      19545  USC00516128  2017-08-19  0.09    71
      19546  USC00516128  2017-08-20   NaN    78
      19547  USC00516128  2017-08-21  0.56    76
      19548  USC00516128  2017-08-22  0.50    76
      19549  USC00516128  2017-08-23  0.45    76
```

```
[29]: df.date.dtype
```

```
[29]: dtype('O')
```

```
[30]: df.date = pd.to_datetime(df.date, infer_datetime_format=True)
      df = df.set_index(df['date'])
      df = df.drop(columns='date')
      df.head()
```

```
[30]:                 station  prcp  tobs
      date
      2010-01-01  USC00519397  0.08    65
      2010-01-02  USC00519397  0.00    63
      2010-01-03  USC00519397  0.00    74
      2010-01-04  USC00519397  0.00    76
      2010-01-06  USC00519397   NaN    73
```

2.1.1 Compare June and December data across all years

```
[31]: from scipy import stats
```

```
[32]: jun_data = df[df.index.month == 6]
      dec_data = df[df.index.month == 12]
```

```
[33]: jun_data.mean()
```

```
[33]: prcp     0.136360
      tobs    74.944118
      dtype: float64
```

```
[34]: dec_data.mean()
```

```
[34]: prcp     0.216819
      tobs    71.041529
      dtype: float64
```

```
[35]: jun_temp = jun_data.tobs
      dec_temp = dec_data.tobs
```

```
[36]: # Run paired t-test
      stats.ttest_ind(jun_temp, dec_temp)
```

```
[36]: Ttest_indResult(statistic=31.60372399000329, pvalue=3.9025129038616655e-191)
```

2.1.2 Analysis

Across all the stations, the mean temperatures in June and December temperature in years 2010-2017 differ by 3.9 degrees Celsius. An unpaired t-test was conducted, and with an extremely low p-value, the difference is deemed statistically significant.