WeatherPy

Analysis

- As expected, the weather becomes significantly warmer as one approaches the equator (0 Deg. Latitude). More interestingly, however, is the fact that the southern hemisphere tends to be warmer this time of year than the northern hemisphere. This may be due to the tilt of the earth.
- There is no strong relationship between latitude and cloudiness. However, it is interesting to see that a strong band of cities sits at 0, 80, and 100% cloudiness.
- There is no strong relationship between latitude and wind speed. However, in northern hemispheres there is a flurry of cities with over 20 mph of wind.

Note

- Instructions have been included for each segment. You do not have to follow them exactly, but they are included to help you think through the steps.

In [81]:

```
# Dependencies and Setup
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import requests
import time
import random
import json


# Import API key
from api_keys import api_key

# Incorporated citipy to determine city based on latitude and longitude
from citipy import citipy

# Range of latitudes and longitudes
lat_range = (-90, 90)
lng_range = (-180, 180)
```

Generate Cities List

In [82]:

```
# List for holding lat_lngs and cities
lat_lngs=[]
cities = []

# Create a set of random lat and lng combinations
lats = np.random.uniform(low=-90.000, high=90.000, size=1500)
lngs = np.random.uniform(low=-180.000, high=180.000, size=1500)
lat_lngs = zip(lats, lngs)

# Identify nearest city for each lat, lng combination
for lat_lng in lat_lngs:
    city = citipy.nearest_city(lat_lng[0], lat_lng[1]).city_name

    # If the city is unique, then add it to a our cities list
    if city not in cities:
        cities.append(city)

# Print the city count to confirm sufficient count
len(cities)
```

Out[82]:

```
604
```

Perform API Calls

- Perform a weather check on each city using a series of successive API calls.
- Include a print log of each city as it'sbeing processed (with the city number and city name).

In [83]:

```
row_count = 0
url = "http://api.openweathermap.org/data/2.5/weather?"
units = "imperial"
cities_name =[]
cities_country=[]
cities_lat = []
cities_lng = []
cities_temp = []
cities_windspeed = []
cities_humidity = []
cities_cloudiness = []
cities_date = []


for index, city in enumerate(cities):

    #build query
    #build query
    query_url = f"{url}appid={api_key}&q={city}&units={units}"

    #print to ensure loop is working correctly

    print("Processing City # " + str(row_count) + " City Name: " + str(c
    print(query_url)
    row_count += 1

 #  #Run requests to grab the JSON at the requested URL
    city_data = requests.get(query_url).json()

    #Append data to lists; use try except to skip cities with errors
    try:
        cities_name.append(city_data['name'])
        cities_country.append(city_data['sys']['country'])
        cities_lat.append(city_data['coord']['lat'])
        cities_lng.append(city_data['coord']['lon'])
        cities_temp.append(city_data["main"]["temp"])
        cities_windspeed.append(city_data["wind"]["speed"])
        cities_humidity.append(city_data["main"]["humidity"])
        cities_cloudiness.append(city_data["clouds"]["all"])
        cities_date.append(city_data['dt'])
    except:
        print("Error with data. Skipping")
```

```
Processing City # 0 City Name: price
http://api.openweathermap.org/data/2.5/weather?appid=cd1166e654293a0b0
9e9dfae00edca3d&q=price&units=imperial
(http://api.openweathermap.org/data/2.5/weather?appid=cd1166e654293a0b
09e9dfae00edca3d&q=price&units=imperial)
Processing City # 1 City Name: khatanga
http://api.openweathermap.org/data/2.5/weather?appid=cd1166e654293a0b0
9e9dfae00edca3d&q=khatanga&units=imperial
(http://api.openweathermap.org/data/2.5/weather?appid=cd1166e654293a0b
09e9dfae00edca3d&q=khatanga&units=imperial)
Processing City # 2 City Name: biak
http://api.openweathermap.org/data/2.5/weather?appid=cd1166e654293a0b0
9e9dfae00edca3d&q=biak&units=imperial
(http://api.openweathermap.org/data/2.5/weather?appid=cd1166e654293a0b
09e9dfae00edca3d&q=biak&units=imperial)
Processing City # 3 City Name: marcona
http://api.openweathermap.org/data/2.5/weather?appid=cd1166e654293a0b0
9e9dfae00edca3d&q=marcona&units=imperial
(http://api.openweathermap.org/data/2.5/weather?appid=cd1166e654293a0b
```

In [84]:

```
# print(json.dumps(city_data, sort_keys=True, indent=4))
```

Convert Raw Data to DataFrame

- Export the city data into a .csv.
- Display the DataFrame

In [163]:

```
weather_dict = {
    "City": cities_name,
    "Country" : cities_country,
    "Latitude": cities_lat,
    "Longitude": cities_lng,
    "Max Temp": cities_temp,
    "Wind Speed": cities_windspeed,
    "Humidity": cities_humidity,
    "Cloudiness": cities_cloudiness,
    "Date": cities_date
}
weather_df = pd.DataFrame(weather_dict)
weather_df.head()
```

Out[163]:

	City	Country	Latitude	Longitude	Max Temp	Wind Speed	Humidity	Cloudiness	Date
0	Price	US	39.60	-110.81	71.31	8.05	13	1	1555709511
1	Khatanga	RU	71.98	102.47	-7.33	4.99	95	45	1555709556
2	Biak	ID	-0.91	122.88	65.93	0.67	99	74	1555709556
3	Hithadhoo	MV	-0.60	73.08	85.01	15.23	77	100	1555709323
4	Lebu	ET	8.96	38.73	62.60	4.70	63	20	1555707600

In [164]:

```
weather_df.to_csv("weather_df.csv", encoding="utf-8", index=False)
```

Plotting the Data

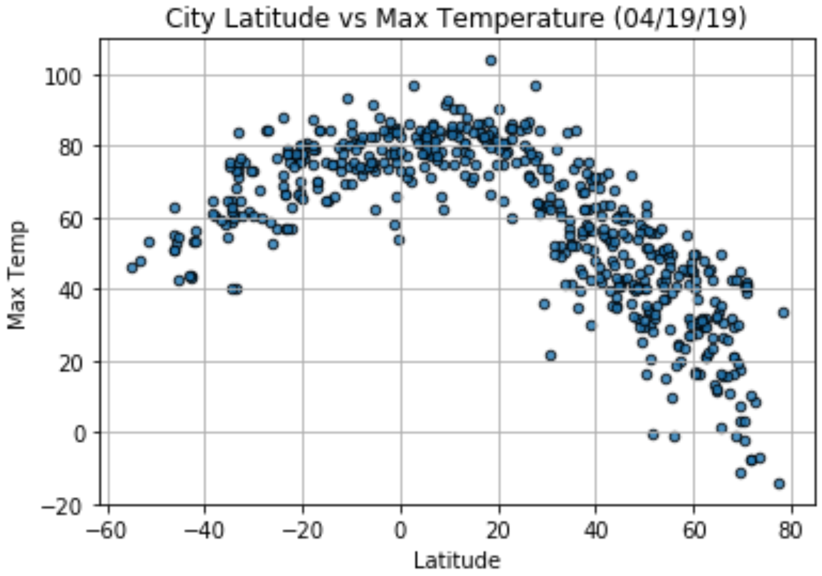
- Use proper labeling of the plots using plot titles (including date of analysis) and axes labels.
- Save the plotted figures as .pngs.

Latitude vs. Temperature Plot

```
In [171]: # plt.scatter(weather_df["Lat"], weather_df["Max Temp"], edgecolor="blac
          #              alpha=0.8, label="Max Temps")
          # plt.title("City Latitude vs Temperature (04/19/19)")
          # plt.ylabel("Temperature (F)")
          # plt.xlabel("Latitude")
          # plt.grid()
          # plt.show()

          weather_df.plot(kind= "scatter", x= "Latitude", y= "Max Temp", grid=True
                          alpha=0.8, title= "City Latitude vs Max Temperature (04/

          plt.savefig("Images/LatvsTemp.png")
          plt.show()
```

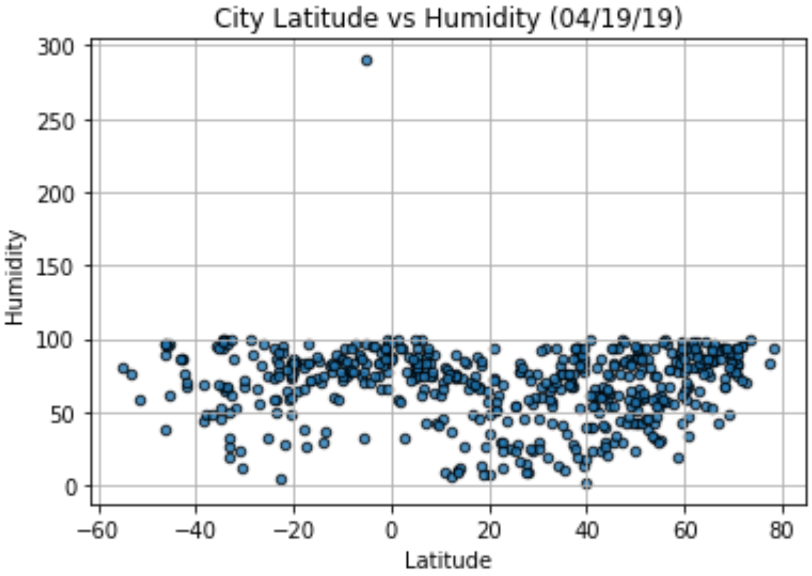


Latitude vs. Humidity Plot

In [172]:
```
# plt.scatter(weather_df["Lat"],weather_df["Humidity"],edgecolor="black"
#              alpha=0.8, label="Humidity")
# plt.title("City Latitude vs Humidity (04/19/19)")
# plt.ylabel("Humidity %")
# plt.xlabel("Latitude")
# plt.grid = True
# plt.show

weather_df.plot(kind= "scatter", x= "Latitude", y= "Humidity", grid=True
            alpha=0.8, title= "City Latitude vs Humidity (04/19/19)")

plt.savefig("Images/LatvsHumidity.png")
plt.show()
```



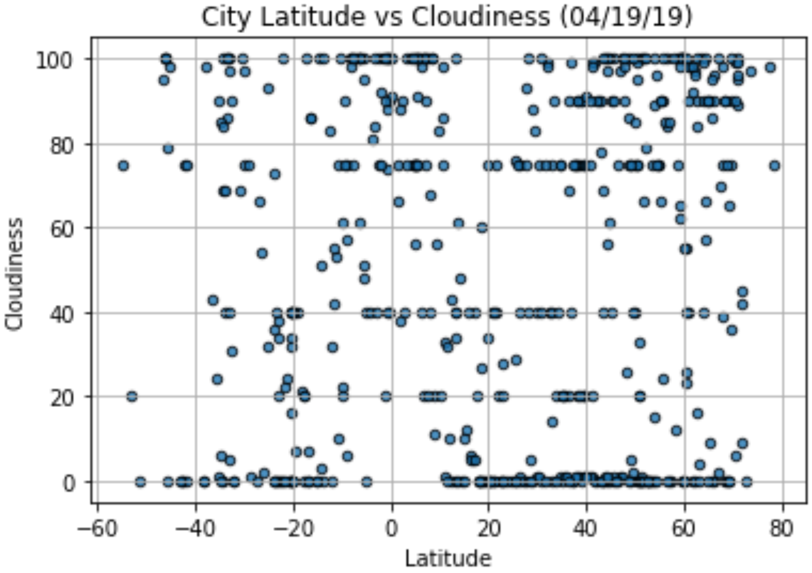
Latitude vs. Cloudiness Plot

In [173]:

```
# plt.scatter(weather_df["Lat"], weather_df["Cloudiness"], edgecolor="bl
#              alpha=0.8, label="Cloudiness")
# plt.title("City Latitude vs Cloudiness (04/19/19)")
# plt.ylabel("Cloudiness %")
# plt.xlabel("Latitude")
# plt.grid = True
# plt.show

weather_df.plot(kind= "scatter", x= "Latitude", y= "Cloudiness", grid=Tr
            alpha=0.8, title= "City Latitude vs Cloudiness (04/19/19)")

plt.savefig("Images/LatvsCloudiness.png")
plt.show()
```



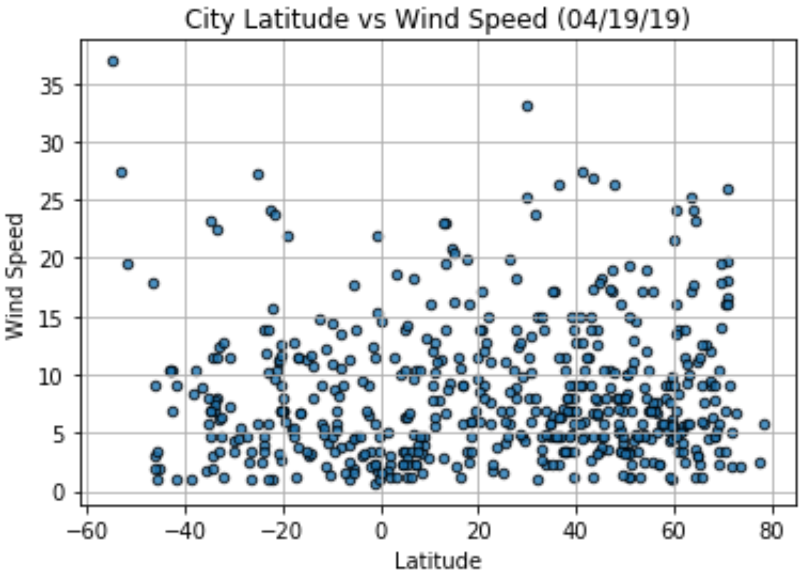
Latitude vs. Wind Speed Plot

In [174]:

```
# plt.scatter(weather_df["Lat"], weather_df["Wind Speed"], edgecolor="bl
#             alpha=0.8, label="Windspeed")
# plt.title("City Latitude vs. Wind Speed (04/19/19)")
# plt.ylabel("Wind Speed (mph)")
# plt.xlabel("Latitude")
# plt.grid = True

weather_df.plot(kind= "scatter", x= "Latitude", y= "Wind Speed", grid=Tr
                alpha=0.8, title= "City Latitude vs Wind Speed (04/19/19

plt.savefig("Images/LatvsWindspeed.png")
plt.show()
```



In [ ]:

```
```