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 [8]:
```
# Dependencies and Setup
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import requests
import time

# Import API key
from config import api_key

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

# Output File (CSV)
output_data_file = "output_data/cities.csv"

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

Generate Cities List

In [2]:
```
# 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[2]: 619

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 [16]:
```
units = "&units=metric"
base_url = "http://api.openweathermap.org/data/2.5/weather?q="
cloudiness = []
country = []
date = []
humidity = []
latitude = []
longitude = []
max_temperature = []
wind_speed = []
citay = []
i=0
for city in cities:
    query_url = base_url + city + "&APPID=" + api_key + units
    i+=1
    try:
        response = requests.get(query_url).json()
        print(f"Processing Record {i} | {city}")
        try:
            cloudiness.append(response['clouds']['all'])
            country.append(response['sys']['country'])
            date.append(response['dt'])
            humidity.append(response['main']['humidity'])
            latitude.append(response['coord']['lat'])
            longitude.append(response['coord']['lon'])
            max_temperature.append(response['main']['temp_max'])
            wind_speed.append(response['wind']["speed"])
            citay.append(city)
        except KeyError:
            print("value not found...skipping")
    except NameError:
        print("city not found...skipping")
```

```
Processing Record 591 | azle
Processing Record 592 | bad vilbel
Processing Record 593 | itapetininga
Processing Record 594 | feni
Processing Record 595 | taoudenni
Processing Record 596 | tautira
Processing Record 597 | olga
Processing Record 598 | meridian
Processing Record 599 | colchester
Processing Record 600 | krasnoselkup
value not found...skipping
Processing Record 601 | hobyo
Processing Record 602 | college
Processing Record 603 | rozivka
Processing Record 604 | camaqua
Processing Record 605 | key west
Processing Record 606 | malacacheta
Processing Record 607 | toamasina
Processing Record 608 | lockhart
```

In [17]:
```
len(citay)
```

Out[17]: 546

In [19]:
```
# create a data frame from cities, lat, and temp
weather_dict = {
    "city":citay,
    "cloudiness (%)": cloudiness,
    "country":country,
    "date":date,
    "humidity (%)":humidity,
    "latitude":latitude,
    "longitude":longitude,
    "max_temperature (C)":max_temperature,
    "wind_speed (m/s)":wind_speed
}
weather_data = pd.DataFrame(weather_dict)
weather_data.head()
```

Out[19]:

	city	cloudiness (%)	country	date	humidity (%)	latitude	longitude	max_temperature (C)
0	ushuaia	40	AR	1552222800	72	-54.81	-68.31	15.00
1	saskylakh	48	RU	1552227015	68	71.97	114.09	-23.21
2	amberley	56	NZ	1552227003	100	-43.15	172.73	14.44
3	klyuchi	0	RU	1552227016	71	52.25	79.17	-11.74
4	rikitea	64	PF	1552227017	100	-23.12	-134.97	24.27

Convert Raw Data to DataFrame

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

In [20]:
```
weather_data.to_csv(header=True, index=True)
```

Out[20]:
```
',city,cloudiness (%),country,date,humidity (%),latitude,longitude,max_t
emperature (C),wind_speed (m/s)\n0,ushuaia,40,AR,1552222800,72,-54.81,-6
8.31,15.0,5.79\n1,saskylakh,48,RU,1552227015,68,71.97,114.09,-23.21,4.94
\n2,amberley,56,NZ,1552227003,100,-43.15,172.73,14.44,1.5\n3,klyuchi,0,R
U,1552227016,71,52.25,79.17,-11.74,3.81\n4,rikitea,64,PF,1552227017,100,
-23.12,-134.97,24.27,4.39\n5,hobart,40,AU,1552227016,67,-42.88,147.33,1
5.0,4.1\n6,bredasdorp,100,ZA,1552226466,56,-34.53,20.04,22.0,9.8\n7,alta
floresta,20,BR,1552222800,74,-9.87,-56.08,28.0,3.1\n8,hudson bay,68,CA,1
552226721,63,52.86,-102.4,-13.26,1.11\n9,lebu,0,ET,1552227018,22,8.96,3
8.73,22.19,3.86\n10,san cristobal,75,EC,1552226464,87,-0.39,-78.55,13.0,
2.6\n11,port alfred,92,ZA,1552226983,81,-33.59,26.89,21.67,3.13\n12,musk
egon,90,US,1552227019,80,43.23,-86.25,2.22,7.7\n13,noumea,0,NC,155222460
0,88,-22.28,166.46,23.0,1.0\n14,faanui,20,PF,1552227021,100,-16.48,-151.
75,27.62,3.31\n15,bethel,90,US,1552225980,74,60.79,-161.76,2.0,3.6\n16,b
luff,0,AU,1552227022,61,-23.58,149.07,25.17,4.49\n17,punta arenas,0,CL,1
552226588,58,-53.16,-70.91,15.0,7.2\n18,hambantota,20,LK,1552223400,78,
6.12,81.12,28.0,3.1\n19,beringovskiy,64,RU,1552227023,83,63.05,179.32,-1
3.89,13.49\n20,shankargarh,0,IN,1552227024,52,25.19,81.61,20.99,1.96\n2
1,hwange,8,ZW,1552227024,38,-18.35,26.5,32.24,1.71\n22,mataura,12,NZ,155
```

In [ ]:
```

```

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.

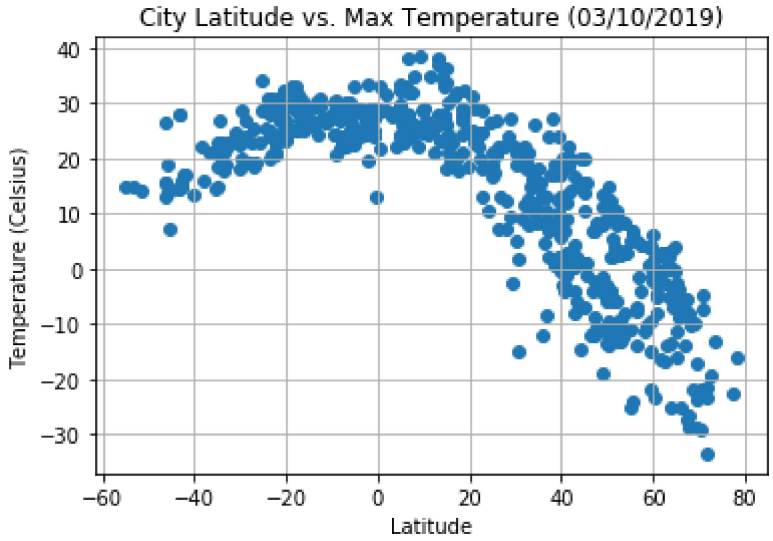
Latitude vs. Temperature Plot

In [21]:
```
# Build a scatter plot for each data type
plt.scatter(weather_data["latitude"], weather_data["max_temperature (C)"], ma

# Incorporate the other graph properties
plt.title("City Latitude vs. Max Temperature (03/10/2019)")
plt.ylabel("Temperature (Celsius)")
plt.xlabel("Latitude")
plt.grid(True)

# Save the figure
plt.savefig("CityLatitudevs.MaxTemperature03102019.png")

# Show plot
plt.show()
```



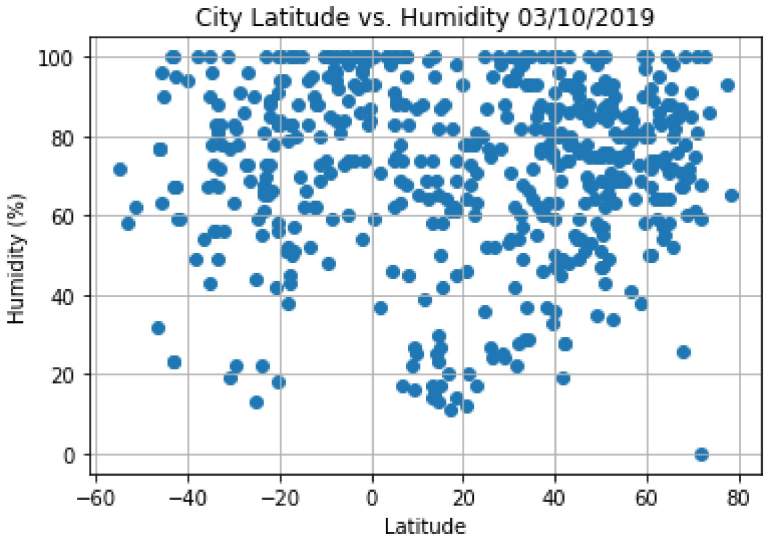
Latitude vs. Humidity Plot

In [22]:
```
# Build a scatter plot for each data type
plt.scatter(weather_data["latitude"], weather_data["humidity (%)"], marker="o

# Incorporate the other graph properties
plt.title("City Latitude vs. Humidity 03/10/2019")
plt.ylabel("Humidity (%)")
plt.xlabel("Latitude")
plt.grid(True)

# Save the figure
plt.savefig("CityLatitudevs.Humidity03102019.png")

# Show plot
plt.show()
```



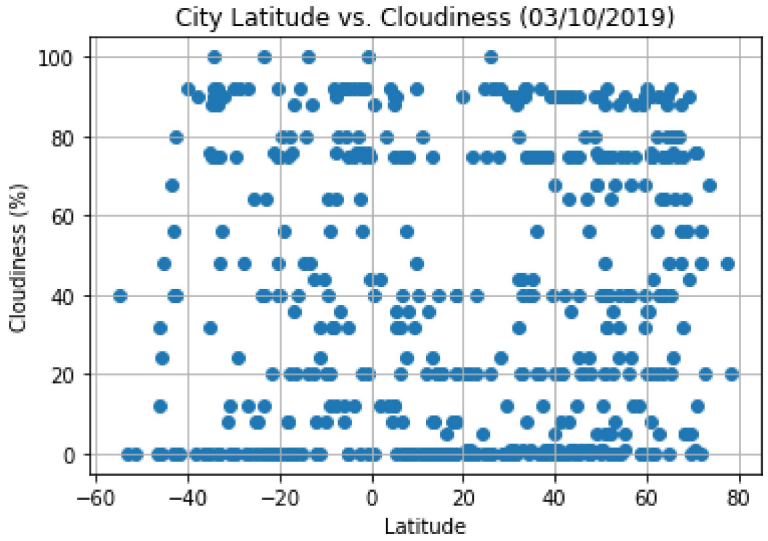
Latitude vs. Cloudiness Plot

In [23]:
```
# Build a scatter plot for each data type
plt.scatter(weather_data["latitude"], weather_data["cloudiness (%)"], marker=

# Incorporate the other graph properties
plt.title("City Latitude vs. Cloudiness (03/10/2019)")
plt.ylabel("Cloudiness (%)")
plt.xlabel("Latitude")
plt.grid(True)

# Save the figure
plt.savefig("CityLatitudevs.Cloudiness03102019.png")

# Show plot
plt.show()
```



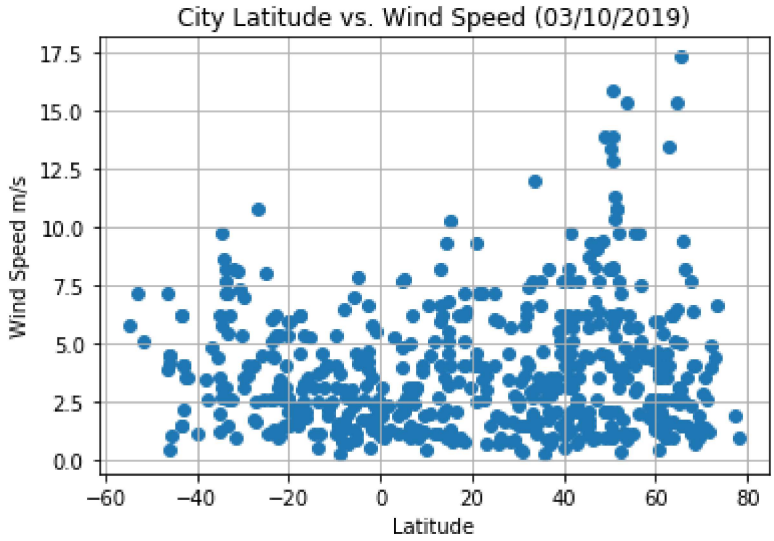
Latitude vs. Wind Speed Plot

In [25]:
```
# Build a scatter plot for each data type
plt.scatter(weather_data["latitude"], weather_data["wind_speed (m/s)"], marke

# Incorporate the other graph properties
plt.title("City Latitude vs. Wind Speed (03/10/2019)")
plt.ylabel("Wind Speed m/s")
plt.xlabel("Latitude")
plt.grid(True)

# Save the figure
plt.savefig("CityLatitudevs.WindSpeed03102019.png")

# Show plot
plt.show()
```



In [ ]:
```

```