In [1]:
```
# Import the dependencies.
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Import the requests library.
import requests
import time
```

In [2]:
```
# Import the datetime module from the datetime library.
from datetime import datetime

# Import the API key and lat-lon to City Name CitiPy
import sys
sys.path.append("../")
from config import weather_api_key
from citipy import citipy

# Import linear regression from the SciPy stats module.
from scipy.stats import linregress
```

In [3]:
```
# Starting URL for Weather Map API Call.
#setting value for number of calls, (to check code with smaller set and avoid hitting API limitations)
NUM_CALLS=2000

params = {"units": "Imperial",
    "APPID": weather_api_key,
    "q": "Boston"} #q is city
#params
#base_url = "https://jsonplaceholder.typicode.com/" #Dummy placeholder, exceeded limit for a 60 calls a minute
base_url = "http://api.openweathermap.org/data/2.5/weather"
```

In [4]:
```
# Create a set of random latitude and longitude combinations.
lats = np.random.uniform(low=-90.000, high=90.000, size=NUM_CALLS)
lngs = np.random.uniform(low=-180.000, high=180.000, size=NUM_CALLS)
lat_lngs = zip(lats, lngs)
lat_lngs
```

Out[4]:
```
<zip at 0x1a466ff7448>
```

Latitude and longitude

In [5]:
```
# Add the latitudes and longitudes to a list.
coordinates = list(lat_lngs)
#coordinates
# Create a list for holding the cities.
cities = []
failed_cities = []
```

```
# Identify the nearest city for each latitude and longitude combination.
for coordinate in coordinates:
    city = citipy.nearest_city(coordinate[0], coordinate[1]).city_name

    # If the city is unique, then we will add it to the cities list.
    if city not in cities:
        cities.append(city)
# Print the city count to confirm sufficient count.
len(cities)
```

Out[5]: 743

In [6]:
```
# Create an empty list to hold the weather data.
city_data = []
skipped_cities = []
# Print the beginning of the logging.
print("Beginning Data Retrieval     ")
print("-----------------------------")

# Create counters.
record_count = 1
set_count = 1
```

```
Beginning Data Retrieval
-----------------------------
```

In [7]:
```
# Loop through all the cities in the list.
for i, city in enumerate(cities):
    if (i == 0):
        print(f"Processing Record {record_count} of Set {set_count} | {city}")

    # Group cities in sets of 50 for logging purposes.
    if (i % 50 == 0 and i >= 50):
        set_count += 1
        record_count = 1
        print(f"Processing Record {record_count} of Set {set_count} | {city}")
        time.sleep(30) #Need to sleep a few seconds (60?) so API calls don't stop
    # Create endpoint URL with each city.
    #city_url = base_url + "&q=" + city.replace(" ","+") needing + is deprecated, using dictionary of params
    params['q'] = city

    # Log the URL, record, and set numbers and the city.
    #print(f"Processing Record {record_count} of Set {set_count} | {city}")
    # Add 1 to the record count.
    record_count += 1
    # Run an API request for each of the cities.
    try:
        # Parse the JSON and retrieve data.
        city_weather = requests.get(base_url, params).json()
        # Parse out the needed data.
        city_lat = city_weather["coord"]["lat"]
```

```
        city_lng = city_weather["coord"]["lon"]
        city_max_temp = city_weather["main"]["temp_max"]
        city_humidity = city_weather["main"]["humidity"]
        city_clouds = city_weather["clouds"]["all"]
        city_wind = city_weather["wind"]["speed"]
        city_country = city_weather["sys"]["country"]
        # Convert the date to ISO standard.
        city_date = datetime.utcfromtimestamp(city_weather["dt"]).strftime('%Y-%m-%d %H:%M:%S')
        #Concatenate Description as comma separated it can contain 7 or more Descriptions https://openweathermap.org/weather-conditions
        weather_desc = ""
        for weather in city_weather["weather"]:
            if len(weather_desc) > 0:
                weather_desc = concatante + ", " + weather["description"]
            else:
                weather_desc = weather["description"]
        # Append the city information into city_data list.
        city_data.append({"City": city.title(),
            "Lat": city_lat,
            "Lng": city_lng,
            "Max Temp": city_max_temp,
            "Humidity": city_humidity,
            "Cloudiness": city_clouds,
            "Wind Speed": city_wind,
            "Country": city_country,
            "Current Description": weather_desc,
            "Date": city_date})

    # If an error is experienced, skip the city.
    except:
        #print(f"{city.title()} not found. Skipping...")
        skipped_cities.append({"City": city.title(),
            "Coords": coordinates[(set_count-1)*10+record_count-1]})
        pass #We are catching errors, commenting out pass!

    # Indicate that Data Loading is complete.
    # print("-----------------------------")
    # print("Data Retrieval Complete      ")
    # print("-----------------------------")
```

```
Processing Record 1 of Set 1 | kalmunai
Processing Record 1 of Set 2 | sorland
Processing Record 1 of Set 3 | caravelas
Processing Record 1 of Set 4 | alice springs
Processing Record 1 of Set 5 | erdenet
Processing Record 1 of Set 6 | sur
Processing Record 1 of Set 7 | sukhumi
Processing Record 1 of Set 8 | mossendjo
Processing Record 1 of Set 9 | karratha
Processing Record 1 of Set 10 | chimore
Processing Record 1 of Set 11 | shrewsbury
Processing Record 1 of Set 12 | aswan
Processing Record 1 of Set 13 | williston
```

```
Processing Record 1 of Set 14 | prince albert
Processing Record 1 of Set 15 | saint-vallier
```

In [13]:
```
# Convert the array of dictionaries to a Pandas DataFrame.
city_data_df = pd.DataFrame(city_data)
city_data_df.head(10)
```

Out[13]:

	City	Lat	Lng	Max Temp	Humidity	Cloudiness	Wind Speed	Country	Current Description	Date
0	Kalmunai	7.4167	81.8167	77.18	79	93	4.92	LK	overcast clouds	2021-03-28 22:48:13
1	Talnakh	69.4865	88.3972	-14.22	99	59	4.16	RU	broken clouds	2021-03-28 22:48:14
2	Snasa	64.2457	12.3778	35.67	98	100	3.18	NO	light rain	2021-03-28 22:48:14
3	Ushuaia	-54.8000	-68.3000	46.40	61	75	14.97	AR	broken clouds	2021-03-28 22:48:14
4	Bredasdorp	-34.5322	20.0403	60.80	88	100	11.50	ZA	overcast clouds	2021-03-28 22:48:14
5	Hilo	19.7297	-155.0900	71.60	78	90	4.61	US	overcast clouds	2021-03-28 22:48:14
6	Chiredzi	-21.0500	31.6667	64.63	84	5	5.19	ZW	clear sky	2021-03-28 22:48:15
7	Sampit	-2.5333	112.9500	74.10	97	100	0.72	ID	moderate rain	2021-03-28 22:48:15
8	Victoria	22.2855	114.1577	75.99	87	23	6.98	HK	few clouds	2021-03-28 22:48:09
9	Carnarvon	-24.8667	113.6333	71.60	94	40	5.75	AU	scattered clouds	2021-03-28 22:48:15

In [14]:
```
new_column_order = ["City", "Country", "Lat", "Lng", "Max Temp", "Humidity", "Cloudiness", "Wind Speed", "Current Description", "Date"]
city_data_df = city_data_df[new_column_order]
city_data_df
```

Out[14]:

	City	Country	Lat	Lng	Max Temp	Humidity	Cloudiness	Wind Speed	Current Description	Date
0	Kalmunai	LK	7.4167	81.8167	77.18	79	93	4.92	overcast clouds	2021-03-28 22:48:13
1	Talnakh	RU	69.4865	88.3972	-14.22	99	59	4.16	broken clouds	2021-03-28 22:48:14
2	Snasa	NO	64.2457	12.3778	35.67	98	100	3.18	light rain	2021-03-28 22:48:14
3	Ushuaia	AR	-54.8000	-68.3000	46.40	61	75	14.97	broken clouds	2021-03-28 22:48:14
4	Bredasdorp	ZA	-34.5322	20.0403	60.80	88	100	11.50	overcast clouds	2021-03-28 22:48:14
...	...	...	...	...	...	...	...	...	...	...
663	Key Largo	US	25.0865	-80.4473	81.00	74	1	5.01	clear sky	2021-03-28 22:57:37
664	Kashi	CN	39.4547	75.9797	53.60	43	40	8.95	scattered clouds	2021-03-28 22:57:37
665	Geraldton	AU	-28.7667	114.6000	66.20	77	14	4.61	few clouds	2021-03-28 22:57:38
666	San Cristobal	VE	7.7669	-72.2250	78.80	73	40	9.22	scattered clouds	2021-03-28 22:57:38

	City	Country	Lat	Lng	Max Temp	Humidity	Cloudiness	Wind Speed	Current Description	Date
667	Terrace	CA	54.5163	-128.6035	41.00	48	75	11.50	light snow	2021-03-28 22:57:38

668 rows × 10 columns

In [17]:
```
# Create the output file (CSV).
output_data_file = "WeatherPy_Database.csv"
# Export the City_Data into a CSV.
city_data_df.to_csv(output_data_file, index_label="City_ID")
```

In [18]:
```
#write error's out here
output_data_file = "WeatherPy_DB_skipped.csv"
skipped_cities_df = pd.DataFrame(skipped_cities)
# Export the City_Data into a CSV.
skipped_cities_df.to_csv(output_data_file, index_label="City_ID")
```