Part 1 Air Quality and Environmental Monitoring Data 1980-2023

Sprint 1: Data Pull and Preliminary EDA

- Brooke Hall
- Sprint 1
- March 8, 2024

Data Sources:

1. United States Environmental Protection Agency: https://aqs.epa.gov/aqsweb/airdata/download_files.html
2. Kaggle for Daily AQI: https://www.kaggle.com/datasets/threnjen/40-years-of-air-quality-index-from-the-epa-daily
3. Data Dictionary: https://aqs.epa.gov/aqsweb/airdata/FileFormats.html

NOTES

This Jupyter notebook has been setup to combine/concatenate some of the datafiles and remove some columns that arent necessary. Preliminary EDA to achieve the shape of the data and retrieve some dim Tables to be used as information at a later date.

Import all Packages

In [1]:
```
import pandas as pd
import numpy as np
import datetime as dt
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec # for posititioning plots
import glob # For gluing directory paths
import seaborn as sns
%matplotlib inline
```

Functions for Combining, Pulling and Preliminary EDA

In [2]:
```
# Create a function to get variable name
# Useful for titles...etc

def nameof(var):
    '''
    Import a variable and return the name as a string
    '''
```

```
    for name, value in globals().items():
        if value is var:
            return name
```

In [3]:
```
# Create a function which can be used to create a master df
# from all csvs within one folder directory path
# Default Date is None

def folder_df_glued(directoryPath, date = None):
    '''
    Import a directory path and export a dataframe.
    Uses glob to get a list of the csv names within a specified directory pa

    '''

    df_glued = pd.DataFrame()

    for file_name in glob.glob(directoryPath+'*.csv'):
        x = pd.read_csv(file_name, parse_dates = date, low_memory=False)
        df_glued = pd.concat([df_glued,x],axis=0)

    df_glued['Directory Path'] = directoryPath

    return df_glued
```

In [4]:
```
# def preliminary_eda(df_list)
## Only use this for the daily files

def preliminary_eda(df_list):
    '''
    Call in the list of dataframes for daily data.
    Complete a series of preliminary eda computations.
    Create subplots for the parameter values.

    Uses the nameof function for titles and identifiers.

    '''

    # Set bold settings so the title is clear
    bold_start = '\033[1m'
    bold_end   = '\033[0m'

    # for loop to cycle through the dataframes within the list of dataframes
    # Complete for each dataframe
    for i, df in enumerate(df_list):

        # Title of section - uses nameof to get the table name
        title = print(bold_start, f'{nameof(df)}', bold_end)

        # Complete and display the following to get samples of each datafram
        display(df.shape)
        display(df.head(2))
        display(df.info())
        display(df.describe())
        display(df.corr(numeric_only = True)) # correlation table
```

```
        display(df.isna().sum()) # Any empty values
        display(df[['State Name', 'County Name', "Parameter Name", 'Date Loc

        # The following code adds plots for each
        ## first, get the unique parameters (reading levels) and the length
        list_param = df['Parameter Name'].unique()
        len_list_param = len(list_param)

        ## Set subplots
        plt.subplots(len_list_param, 1, figsize=(20,len_list_param*3))

        # For each parameter in the list, create a unique subplot
        for index, parameter in enumerate(list_param):
            mask = df['Parameter Name'] == parameter

            plt.subplot(len_list_param, 1, index+1)
            plt.scatter(x = df['Date Local'][mask], y = df['Arithmetic Mean'
            plt.xlabel('Date')
            plt.ylabel('Parameter Value')
            plt.title(f'{nameof(df)} - {parameter}')

        plt.show()
```

In [5]:
```
def plotting_eda(df_list):
    '''
    Plotting for daily data only

    '''

# Plot histogram and probability plot for both models

# Set bold settings so the title is clear
    bold_start = '\033[1m'
    bold_end   = '\033[0m'

    # for loop to cycle through the dataframes within the list of dataframes
    # Complete for each dataframe
    for i, df in enumerate(df_list):

        # Title of section - uses nameof to get the table name
        title = print(bold_start, f'{nameof(df)}', bold_end)

        list_param = df['Parameter Name'].unique()
        len_list_param = len(list_param)

        fig = plt.figure(figsize = (20,len_list_param*3))

        # set up subplot grid; how many rows and columns
        gridspec.GridSpec(len_list_param,2) # 2 by 2 grid

        # For each parameter in the list, create a unique subplot
        for index, parameter in enumerate(list_param):

            mask = df['Parameter Name'] == parameter
            ser = df['Arithmetic Mean'][mask]
```

```
            plt.subplot2grid((len_list_param,2), (index,0))  # (2,1) = total
            plt.hist(ser, bins = 100)
            plt.xlabel(f'{parameter}')
            plt.title(f'Histogram of {nameof(df)} - {parameter}')

            plt.subplot2grid((len_list_param,2), (index,1))
            plt.scatter(x = df['Date Local'][mask], y = df['Arithmetic Mean'
            plt.xlabel('Date')
            plt.ylabel('Parameter Value')
            plt.title(f'Scatter Plot of {nameof(df)} - {parameter}')

        plt.tight_layout() # Keeps plots from overlapping each other
        plt.show()
```

Import Data & Preliminary EDA

Datasets are given as flat files for each year for different categories of daily monitoring data. They need to be pulled in (collectively 1980-2023) and grouped. Then reviewed individually before determining whether they can be combined. Data manipulation is needed to get the dataset into a format that can be used, but conditions need to be checked first.

AQI Data

Notes:

> AQI Daily data 1980-2021 is pulled from Kaggle, where it is already combined for simplicity (Source 1). 2022 & 2023 are to be pulled and glued from the United States Environmental Protection Agency website (Source 2).

AQI TAKES INTO ACCOUNT CRITERIA GASES AND PARTICULATES. This value includes that data, does not include toxics...etc Any air quality data will thus be included in the AQI table, then meteorological and hazards to be included in the remainder of the model

Air Quality Index (AQI) Calculation Method

An air quality index is calculated hourly using the following five contaminants: ozone, particulate matter, sulfur dioxide, nitrogen dioxide and carbon monoxide.

For each of the contaminants measured at an air monitoring station, a sub-index is calculated first. The sub-index is calculated by dividing the concentration of a contaminant monitored by its corresponding reference value and multiplying the result by 50. A contaminant's reference value is the concentration at which air quality is considered "poor". This value is determined on the basis of criteria to protect human health. The reference values are as follows:

Contaminant	Type of Measurement	Reference Value
Ozone (O_3)	Hourly average	82 ppb
Particulate matter ($PM_{2.5}$)	Average over 3 hours	35 $\mu g/m^3$
Sulfur dioxide (SO_2)	Maximum over 4 minutes[1]	200 ppb
Nitrogen dioxide (NO_2)	Hourly average	213 ppb
Carbon monoxide (CO)	Hourly average	30 ppm

AQI Data Pull and Glue

In [10]:
```
path = '/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/'
d1 = 'data/aqiDaily/'
```

```
# Function call to import daily aqi
daily_aqi_rough = folder_df_glued(path+d1, ['Date'])
```

In [11]:
```
display(daily_aqi_rough.shape)
daily_aqi_rough.head()
```

```
(11158187, 11)
```

Out[11]:

	State Name	county Name	State Code	County Code	Date	AQI	Category	Defining Parameter	Defining Site	Nu of Repo
0	Alabama	Baldwin	1	3	2007-01-03	55	Moderate	PM2.5	01-003-0010	
1	Alabama	Baldwin	1	3	2007-01-06	23	Good	PM2.5	01-003-0010	
2	Alabama	Baldwin	1	3	2007-01-09	13	Good	PM2.5	01-003-0010	
3	Alabama	Baldwin	1	3	2007-01-12	25	Good	PM2.5	01-003-0010	
4	Alabama	Baldwin	1	3	2007-01-15	17	Good	PM2.5	01-003-0010	

In [12]:
```
# Rename so columns are clear before the join
daily_aqi_rough.rename(columns = {'county Name': 'County Name', 'Category':
                                  'Defining Parameter': 'AQI Defining Parameter'},
```

In [13]:
```
# Ensure we have the expected number of data points.
# Every county should have approximately
count = 1980
years = 2023
sum = 0

while count < years:
    mask = daily_aqi_rough['Date'].dt.year == count
    sum += len(daily_aqi_rough[mask]['County Name'].value_counts())*365
    count += 1

print(f'The expected number of rows in this dataframe is {sum}')
```

```
The expected number of rows in this dataframe is 11256965
```

In [14]:
```
daily_aqi_rough[['County Code', 'County Name']].value_counts()
```

Out[14]:
```
County Code  County Name
1            Adams                61386
3            Allen                27386
1            Albany               25578
19           Clark                24110
31           Cook                 21540
                                  ...
131          Ontonagon               25
188          Northwest Arctic        21
21           Campbell                20
161          Mason                   19
191          Washington              16
Name: count, Length: 1500, dtype: int64
```

The expected number of rows in this dataframe are greater than what is actually present. The intention was to see if there were any missing dates. It would appear that not all counties have the same counts. This shows sites raised at later dates.

AQI Setup and Preliminary EDA

In [15]:
```
# Preliminary EDA
display(daily_aqi_rough.head())
display(daily_aqi_rough.tail())
display(daily_aqi_rough.info())
display(daily_aqi_rough.isna().sum())
display(daily_aqi_rough.sample(20))

# Check if there are any duplicates by day or location and na values
dup = daily_aqi_rough[['State Name', 'County Name', 'Date']].duplicated().su
print(f'There are {dup} duplicates by County/State/Day')
```

	State Name	County Name	State Code	County Code	Date	AQI	AQI Category Sentiment	AQI Defining Parameter	Defining Site	Nur of S Repo
0	Alabama	Baldwin	1	3	2007-01-03	55	Moderate	PM2.5	01-003-0010	
1	Alabama	Baldwin	1	3	2007-01-06	23	Good	PM2.5	01-003-0010	
2	Alabama	Baldwin	1	3	2007-01-09	13	Good	PM2.5	01-003-0010	
3	Alabama	Baldwin	1	3	2007-01-12	25	Good	PM2.5	01-003-0010	
4	Alabama	Baldwin	1	3	2007-01-15	17	Good	PM2.5	01-003-0010	

	State Name	County Name	State Code	County Code	Date	AQI	AQI Category Sentiment	AQI Defining Parameter	Defining Site
322781	Wyoming	Weston	56	45	2022-12-27	33	Good	Ozone	56-045-0003
322782	Wyoming	Weston	56	45	2022-12-28	36	Good	Ozone	56-045-0003
322783	Wyoming	Weston	56	45	2022-12-29	34	Good	Ozone	56-045-0003
322784	Wyoming	Weston	56	45	2022-12-30	36	Good	Ozone	56-045-0003
322785	Wyoming	Weston	56	45	2022-12-31	37	Good	Ozone	56-045-0003

```
<class 'pandas.core.frame.DataFrame'>
Index: 11158187 entries, 0 to 322785
Data columns (total 11 columns):
 #   Column                    Dtype         
---  ------                    -----         
 0   State Name                object        
 1   County Name               object        
 2   State Code                object        
 3   County Code               int64         
 4   Date                      datetime64[ns]
 5   AQI                       int64         
 6   AQI Category Sentiment    object        
 7   AQI Defining Parameter    object        
 8   Defining Site             object        
 9   Number of Sites Reporting  int64         
 10  Directory Path            object        
dtypes: datetime64[ns](1), int64(3), object(7)
memory usage: 1021.6+ MB
None
State Name                   0
County Name                  0
State Code                   0
County Code                  0
Date                         0
AQI                          0
AQI Category Sentiment       0
AQI Defining Parameter       0
Defining Site                0
Number of Sites Reporting    0
Directory Path               0
dtype: int64
```

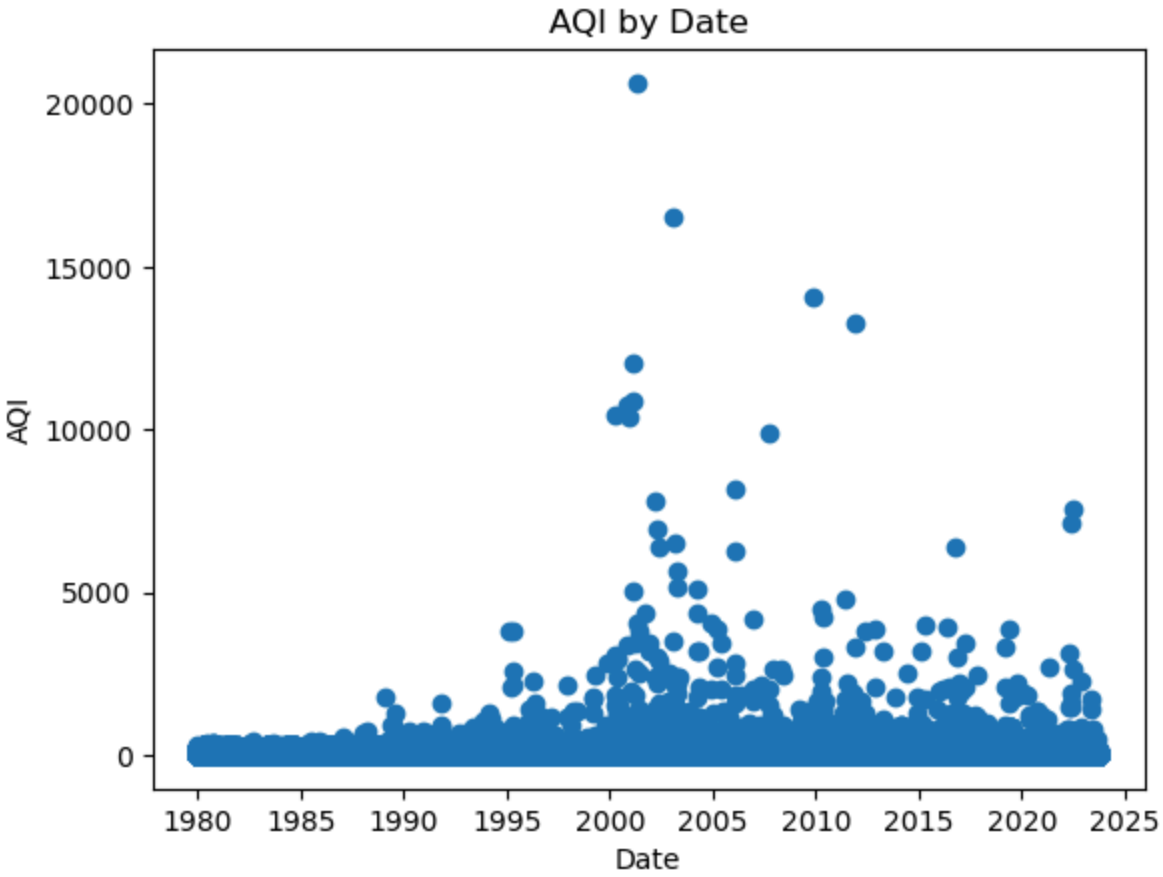
	State Name	County Name	State Code	County Code	Date	AQI	AQI Category Sentiment	AQI Defining Parameter	D
607	Alabama	Colbert	1	33	2010-08-15	54	Moderate	PM2.5	0
163510	Nevada	White Pine	32	33	2010-03-21	47	Good	Ozone	3
206459	Pennsylvania	Adams	42	1	2001-10-05	108	Unhealthy for Sensitive Groups	Ozone	4
260334	South Dakota	Minnehaha	46	99	2018-04-23	93	Moderate	Ozone	4
251895	South Carolina	York	45	91	2011-06-03	64	Moderate	Ozone	4
10762	Arkansas	Clark	05	19	2003-04-08	44	Good	Ozone	0
287624	West Virginia	Wood	54	107	2004-01-04	25	Good	PM2.5	5
216195	West Virginia	Wood	54	107	1993-04-20	42	Good	Ozone	5
303404	West Virginia	Monongalia	54	61	2013-10-20	27	Good	Ozone	5
31260	California	Sutter	6	101	2015-11-10	56	Moderate	PM2.5	0
84213	New Mexico	Santa Fe	35	49	1984-05-16	15	Good	CO	3
120781	Nebraska	Dawson	31	47	1996-09-06	44	Good	PM10	3
14959	California	Butte	6	7	2013-01-08	87	Moderate	PM2.5	0
27342	Florida	Duval	12	31	1987-11-09	30	Good	Ozone	1
133791	Michigan	Genesee	26	49	2020-02-12	43	Good	PM2.5	2
161005	Oklahoma	Oklahoma	40	109	1993-09-01	38	Good	Ozone	4
132825	Virginia	Hampton City	51	650	1981-04-11	45	Good	Ozone	5
103503	Kentucky	Jefferson	21	111	2005-11-27	63	Moderate	PM2.5	

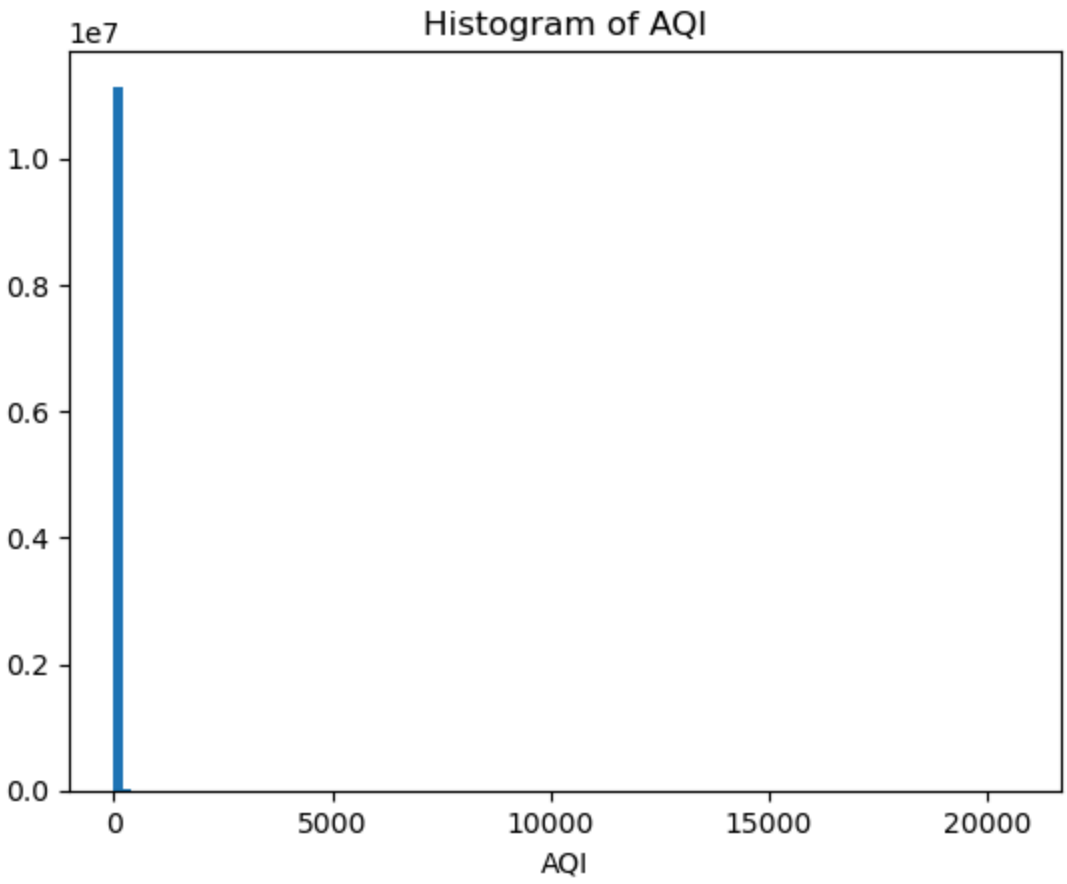
	State Name	County Name	State Code	County Code	Date	AQI	AQI Category Sentiment	AQI Defining Parameter	D
86445	Kansas	Trego	20	195	2000-11-20	35	Good	Ozone	2
64050	Louisiana	Pointe Coupee	22	77	1989-02-26	46	Good	Ozone	2

```
There are 0 duplicates by County/State/Day
```

- AQI seems to have consitent data throughout (checking head, tail and sample).
- The shape gives more insight into the number of rows - to be checked for an approximate amount compared to what is expected in this dataset, by county and day.
- The date column is the only datetime.
- The join was successful and Lat/Lon data is populated in the AQI table. When joining daily data, values can be grouped by county and joined to the aqi dataset. No data was lost - same number of rows as the original dataframe.
- There are no NaN values for any columns. There are no duplicates by County and Date. This is consistent with what's expected and shows that only one AQI exists per day and location.

In [16]:
```
# Scatterplot for preliminary overview
plt.figure()
plt.scatter(x = daily_aqi_rough['Date'], y = daily_aqi_rough['AQI'])
plt.xlabel('Date')
plt.ylabel('AQI')
plt.title('AQI by Date')
plt.show()

plt.figure()
plt.hist(daily_aqi_rough['AQI'], bins = 100)
plt.xlabel('AQI')
plt.title(f'Histogram of AQI')
plt.show()
```


AQI by Date
AQI
Date
0
5000
10000
15000
20000
1980
1985
1990
1995
2000
2005
2010
2015
2020
2025


Histogram of AQI
1e7
AQI
0.0
0.2
0.4
0.6
0.8
1.0
0
5000
10000
15000
20000

With time, AQI appears to be relatively consistent. It starts to have some higher datapoints/more variability towards 2000s. These start to decrease/improve after 2010 onto present. This may infer that air quality and pollution controls have become more stringent/closely monitored accross the states. The scatter plot is only a preliminary look and does not view by location.

It's interesting that the histogram is so concentrated on the left. This indicates high outliers may be present and will require a closer look at a later date.

In [17]:
```
daily_aqi_rough
```

Out[17]:

	State Name	County Name	State Code	County Code	Date	AQI	AQI Category Sentiment	AQI Defining Parameter	Definin Sit
0	Alabama	Baldwin	1	3	2007-01-03	55	Moderate	PM2.5	01-003 001
1	Alabama	Baldwin	1	3	2007-01-06	23	Good	PM2.5	01-003 001
2	Alabama	Baldwin	1	3	2007-01-09	13	Good	PM2.5	01-003 001
3	Alabama	Baldwin	1	3	2007-01-12	25	Good	PM2.5	01-003 001
4	Alabama	Baldwin	1	3	2007-01-15	17	Good	PM2.5	01-003 001
...	...	...	...	...	...	...	...	...	
322781	Wyoming	Weston	56	45	2022-12-27	33	Good	Ozone	56-045 000
322782	Wyoming	Weston	56	45	2022-12-28	36	Good	Ozone	56-045 000
322783	Wyoming	Weston	56	45	2022-12-29	34	Good	Ozone	56-045 000
322784	Wyoming	Weston	56	45	2022-12-30	36	Good	Ozone	56-045 000
322785	Wyoming	Weston	56	45	2022-12-31	37	Good	Ozone	56-045 000

11158187 rows × 11 columns

In [18]:
```
daily_aqi_rough['Month'] = daily_aqi_rough['Date'].dt.month
daily_aqi_rough['Year'] = daily_aqi_rough['Date'].dt.year
```

In [19]:
```
# Only include relevant columns
daily_aqi_working = daily_aqi_rough.drop(columns = ['Defining Site', 'Number
```

```
daily_aqi_working.head()
```

Out[19]:

	State Name	County Name	Date	AQI	AQI Category Sentiment	AQI Defining Parameter	Month	Year
0	Alabama	Baldwin	2007-01-03	55	Moderate	PM2.5	1	2007
1	Alabama	Baldwin	2007-01-06	23	Good	PM2.5	1	2007
2	Alabama	Baldwin	2007-01-09	13	Good	PM2.5	1	2007
3	Alabama	Baldwin	2007-01-12	25	Good	PM2.5	1	2007
4	Alabama	Baldwin	2007-01-15	17	Good	PM2.5	1	2007

The aqi dataset is very clean. There are no nulls, only relevant information is included. Only one row/datapoint exists per day and location. Potential for monthly review. This table is ready for further analysis and potentially combination with the daily_combined table.

Daily Individual Parameter Monitoring Data

Notes:

> Daily parameter data are pulled yearly in csv flat files by groups of categories 1980-2021 from the United States Environmental Protection Agency website (Source 2).

Particulates and Criteria gas data is not included as these are involved in the AQI calculations

In [20]:
```
# Get folder paths for the different levels to be used in the function call
d2 = path+'data/dailyPressure/'
d3 = path+'data/dailyWind/'
d4 = path+'data/dailyTemp/'
d5 = path+'data/dailyRhDp/'
d6 = path+'data/dailyHAPS/'
d7 = path+'data/dailyVOCs/'
d8 = path+'data/dailyNONOxNOy/'
d9 = path+'data/dailyLead/'

dolc = ['Date Local', 'Date of Last Change'] # Used to set date column to pa
```

VOCs and HAPs had 150+ parameters that were not included at all sites. These files were dropped in favor of being able to manipulate the large dataset.

In [21]:
```
# Function call to import daily pressure
```

```
daily_pressure = folder_df_glued(d2, dolc)
```

In [22]:
```
# Function call to import daily wind
daily_wind = folder_df_glued(d3, dolc)
```

In [23]:
```
# Function call to import daily temperature
daily_temp = folder_df_glued(d4, dolc)
```

In [24]:
```
# Function call to import daily Relative Humidity (RH) and Dew Point (DP)
daily_RH_DP = folder_df_glued(d5, dolc)
```

In [25]:
```
# Function call to import daily NONOxNOy
daily_NONOxNOy = folder_df_glued(d8, dolc)
```

In [26]:
```
# Function call to import daily lead
daily_lead = folder_df_glued(d9, dolc)
```

In [27]:
```
# Create a list of dataframes to concatenate and join daily dataframes
df_daily_list = [daily_pressure, daily_wind, daily_temp, daily_RH_DP, daily_
```

In [28]:
```
preliminary_eda(df_daily_list)
```

daily_pressure
(2428944, 30)

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longitude	Datum	Parameter Name	Sar Dura
0	1	53	1000	64101	1	31.0921	-87.5435	NAD83	Barometric pressure	1 H
1	1	53	1000	64101	1	31.0921	-87.5435	NAD83	Barometric pressure	1 H

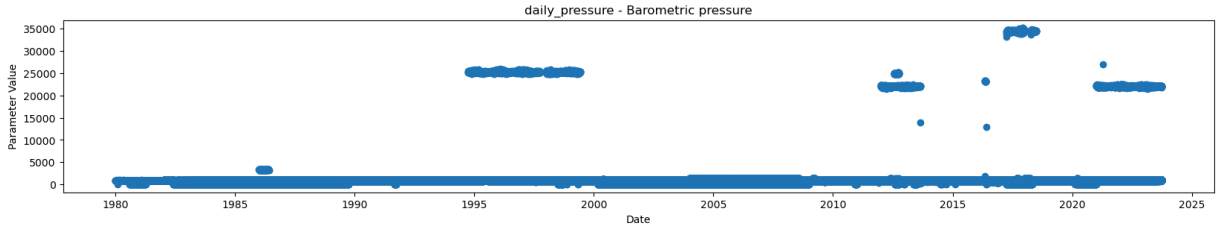
2 rows × 30 columns

```
<class 'pandas.core.frame.DataFrame'>
Index: 2428944 entries, 0 to 115935
Data columns (total 30 columns):
 #   Column               Dtype         
---  ------               -----         
 0   State Code           int64         
 1   County Code          int64         
 2   Site Num             int64         
 3   Parameter Code       int64         
 4   POC                  int64         
 5   Latitude             float64       
 6   Longitude            float64       
 7   Datum                object        
 8   Parameter Name       object        
 9   Sample Duration      object        
 10  Pollutant Standard   float64       
 11  Date Local           datetime64[ns]
 12  Units of Measure     object        
 13  Event Type           object        
 14  Observation Count    int64         
 15  Observation Percent  float64       
 16  Arithmetic Mean      float64       
 17  1st Max Value        float64       
 18  1st Max Hour         int64         
 19  AQI                  float64       
 20  Method Code          int64         
 21  Method Name          object        
 22  Local Site Name      object        
 23  Address              object        
 24  State Name           object        
 25  County Name          object        
 26  City Name            object        
 27  CBSA Name            object        
 28  Date of Last Change  datetime64[ns]
 29  Directory Path       object        
dtypes: datetime64[ns](2), float64(7), int64(8), object(13)
memory usage: 574.5+ MB
None
```

	State Code	County Code	Site Num	Parameter Code	POC	La
count	2.428944e+06	2.428944e+06	2.428944e+06	2428944.0	2.428944e+06	2.42894
mean	2.797541e+01	6.385323e+01	8.815617e+02	64101.0	1.011944e+00	4.01427
min	1.000000e+00	1.000000e+00	1.000000e+00	64101.0	1.000000e+00	1.84200
25%	1.600000e+01	1.900000e+01	5.000000e+00	64101.0	1.000000e+00	3.75565
50%	2.600000e+01	4.100000e+01	2.300000e+01	64101.0	1.000000e+00	4.13014
75%	3.900000e+01	8.500000e+01	1.005000e+03	64101.0	1.000000e+00	4.28618
max	7.200000e+01	5.100000e+02	9.997000e+03	64101.0	9.000000e+00	6.48456
std	1.657298e+01	7.017843e+01	1.754541e+03	0.0	1.830547e-01	4.30775

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longit
State Code	1.000000	-0.055062	-0.161438	NaN	-0.019768	0.425179	0.189
County Code	-0.055062	1.000000	-0.105939	NaN	-0.023219	-0.222423	0.102
Site Num	-0.161438	-0.105939	1.000000	NaN	-0.007173	-0.170152	-0.132
Parameter Code	NaN	NaN	NaN	NaN	NaN	NaN	
POC	-0.019768	-0.023219	-0.007173	NaN	1.000000	-0.014064	0.035
Latitude	0.425179	-0.222423	-0.170152	NaN	-0.014064	1.000000	0.175
Longitude	0.189598	0.102836	-0.132021	NaN	0.035881	0.175733	1.000
Pollutant Standard	NaN	NaN	NaN	NaN	NaN	NaN	
Observation Count	0.011435	0.002572	-0.009962	NaN	-0.014171	0.005666	-0.003
Observation Percent	0.011333	0.002590	-0.009934	NaN	-0.013910	0.005631	-0.003
Arithmetic Mean	0.010171	0.039717	-0.004911	NaN	0.000306	0.001729	0.021
1st Max Value	0.010328	0.039616	-0.004930	NaN	0.000408	0.002045	0.022
1st Max Hour	0.039268	-0.013145	-0.009758	NaN	0.004405	0.035893	0.034
AQI	NaN	NaN	NaN	NaN	NaN	NaN	
Method Code	-0.033129	0.000193	-0.023538	NaN	-0.004186	-0.151922	0.002

```
State Code                 0
County Code                0
Site Num                   0
Parameter Code             0
POC                        0
Latitude                   0
Longitude                  0
Datum                      0
Parameter Name             0
Sample Duration            0
Pollutant Standard     2428944
Date Local                 0
Units of Measure           0
Event Type             2410440
Observation Count          0
Observation Percent        0
Arithmetic Mean            0
1st Max Value              0
1st Max Hour               0
AQI                    2428944
Method Code                0
Method Name                0
Local Site Name       177904
Address                 1046
State Name                 0
County Name                0
City Name                  0
CBSA Name              313162
Date of Last Change        0
Directory Path             0
dtype: int64
719274
```



daily_wind

```
(12073114, 30)
```

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longitude	Datum	Parameter Name	Sar Dura
0	1	53	1000	61103	1	31.0921	-87.5435	NAD83	Wind Speed - Resultant	1 H
1	1	53	1000	61103	1	31.0921	-87.5435	NAD83	Wind Speed - Resultant	1 H

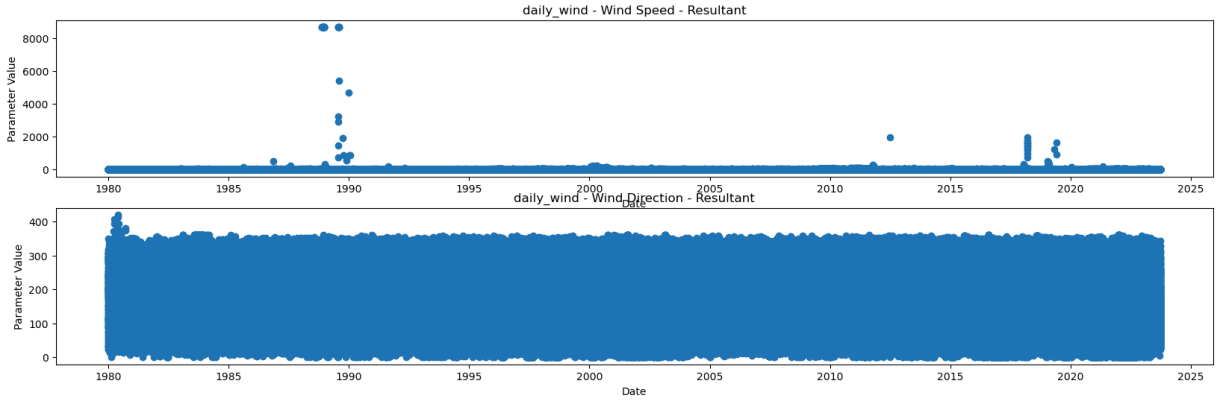
2 rows × 30 columns

```
<class 'pandas.core.frame.DataFrame'>
Index: 12073114 entries, 0 to 492944
Data columns (total 30 columns):
 #   Column               Dtype
---  ------               -----
 0   State Code           int64
 1   County Code          int64
 2   Site Num             int64
 3   Parameter Code       int64
 4   POC                  int64
 5   Latitude             float64
 6   Longitude            float64
 7   Datum                object
 8   Parameter Name       object
 9   Sample Duration      object
 10  Pollutant Standard   float64
 11  Date Local           datetime64[ns]
 12  Units of Measure     object
 13  Event Type           object
 14  Observation Count    int64
 15  Observation Percent  float64
 16  Arithmetic Mean      float64
 17  1st Max Value        float64
 18  1st Max Hour         int64
 19  AQI                  float64
 20  Method Code          int64
 21  Method Name          object
 22  Local Site Name      object
 23  Address              object
 24  State Name           object
 25  County Name          object
 26  City Name            object
 27  CBSA Name            object
 28  Date of Last Change  datetime64[ns]
 29  Directory Path       object
dtypes: datetime64[ns](2), float64(7), int64(8), object(13)
memory usage: 2.8+ GB
None
```

	State Code	County Code	Site Num	Parameter Code	POC	I
count	1.207311e+07	1.207311e+07	1.207311e+07	1.207311e+07	1.207311e+07	1.207
mean	2.649608e+01	9.415203e+01	9.323432e+02	6.110350e+04	1.068815e+00	3.735
min	1.000000e+00	1.000000e+00	1.000000e+00	6.110300e+04	1.000000e+00	1.8334
25%	6.000000e+00	2.900000e+01	7.000000e+00	6.110300e+04	1.000000e+00	3.3320
50%	2.600000e+01	6.900000e+01	3.900000e+01	6.110400e+04	1.000000e+00	3.764
75%	4.800000e+01	1.210000e+02	1.013000e+03	6.110400e+04	1.000000e+00	4.1616
max	8.000000e+01	5.100000e+02	9.997000e+03	6.110400e+04	1.400000e+01	6.709
std	1.807893e+01	9.759365e+01	1.869532e+03	4.999924e-01	4.725516e-01	5.4659

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longit
State Code	1.000000	0.345928	-0.249001	0.004815	0.021071	-0.129203	0.345
County Code	0.345928	1.000000	-0.122420	-0.003281	-0.018654	-0.373648	0.141
Site Num	-0.249001	-0.122420	1.000000	-0.000548	0.154288	-0.053653	-0.12
Parameter Code	0.004815	-0.003281	-0.000548	1.000000	-0.000448	0.005146	-0.000
POC	0.021071	-0.018654	0.154288	-0.000448	1.000000	0.034071	0.045
Latitude	-0.129203	-0.373648	-0.053653	0.005146	0.034071	1.000000	0.156
Longitude	0.345467	0.141397	-0.121051	-0.000257	0.045533	0.156822	1.000
Pollutant Standard	NaN	NaN	NaN	NaN	NaN	NaN	
Observation Count	-0.015148	-0.001631	-0.037867	0.000162	-0.173332	0.000253	-0.015
Observation Percent	-0.015664	-0.001745	-0.039554	-0.000213	-0.173743	0.000431	-0.015
Arithmetic Mean	-0.018370	-0.036669	0.007674	0.887983	0.003592	0.055974	0.003
1st Max Value	-0.033091	-0.042454	0.013634	0.934270	-0.003274	0.035522	-0.027
1st Max Hour	-0.018607	-0.018432	0.007167	-0.049778	-0.005029	0.005400	-0.024
AQI	NaN	NaN	NaN	NaN	NaN	NaN	
Method Code	-0.123429	-0.108452	0.075623	-0.003002	0.243348	0.079512	-0.019

```
State Code                  0
County Code                 0
Site Num                    0
Parameter Code              0
POC                         0
Latitude                    0
Longitude                   0
Datum                       0
Parameter Name              0
Sample Duration             0
Pollutant Standard     12073114
Date Local                  0
Units of Measure            0
Event Type             11990547
Observation Count           0
Observation Percent         0
Arithmetic Mean             0
1st Max Value               0
1st Max Hour                0
AQI                    12073114
Method Code                 0
Method Name                 0
Local Site Name       1034488
Address                  2092
State Name                  0
County Name                 0
City Name                   0
CBSA Name             1225242
Date of Last Change         0
Directory Path              0
dtype: int64
5684089
```



daily_temp

```
(8105340, 30)
```

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longitude	Datum	Parameter Name	Sa Du
0	1	53	1000	62101	1	31.0921	-87.5435	NAD83	Outdoor Temperature	1
1	1	53	1000	62101	1	31.0921	-87.5435	NAD83	Outdoor Temperature	1

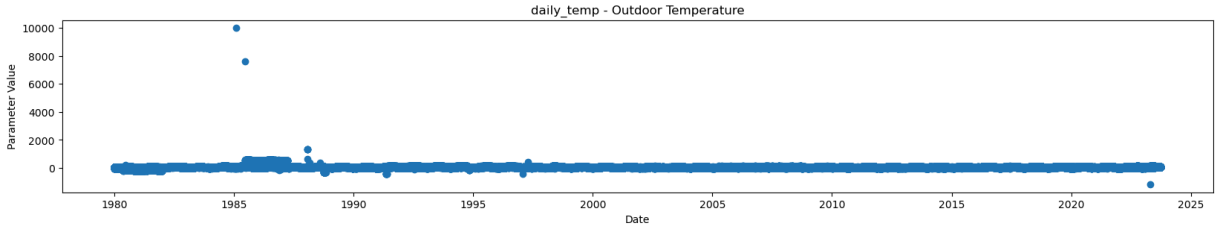
2 rows × 30 columns

```
<class 'pandas.core.frame.DataFrame'>
Index: 8105340 entries, 0 to 227251
Data columns (total 30 columns):
 #   Column               Dtype
---  ------               -----
 0   State Code           int64
 1   County Code          int64
 2   Site Num             int64
 3   Parameter Code       int64
 4   POC                  int64
 5   Latitude             float64
 6   Longitude            float64
 7   Datum                object
 8   Parameter Name       object
 9   Sample Duration      object
 10  Pollutant Standard   float64
 11  Date Local           datetime64[ns]
 12  Units of Measure     object
 13  Event Type           object
 14  Observation Count    int64
 15  Observation Percent  float64
 16  Arithmetic Mean      float64
 17  1st Max Value        float64
 18  1st Max Hour         int64
 19  AQI                  float64
 20  Method Code          int64
 21  Method Name          object
 22  Local Site Name      object
 23  Address              object
 24  State Name           object
 25  County Name          object
 26  City Name            object
 27  CBSA Name            object
 28  Date of Last Change  datetime64[ns]
 29  Directory Path       object
dtypes: datetime64[ns](2), float64(7), int64(8), object(13)
memory usage: 1.9+ GB
None
```

	State Code	County Code	Site Num	Parameter Code	POC	Lati
count	8.105340e+06	8.105340e+06	8.105340e+06	8105340.0	8.105340e+06	8.105340
mean	2.927320e+01	8.290695e+01	8.869007e+02	62101.0	1.116417e+00	3.839551
min	1.000000e+00	1.000000e+00	1.000000e+00	62101.0	1.000000e+00	0.000000
25%	8.000000e+00	2.300000e+01	7.000000e+00	62101.0	1.000000e+00	3.425239
50%	2.900000e+01	5.700000e+01	3.600000e+01	62101.0	1.000000e+00	3.883857
75%	4.800000e+01	1.090000e+02	1.007000e+03	62101.0	1.000000e+00	4.234199
max	8.000000e+01	5.100000e+02	9.997000e+03	62101.0	1.000000e+01	7.029195
std	1.842245e+01	9.124167e+01	1.833557e+03	0.0	4.256162e-01	5.533613

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longi
State Code	1.000000	0.204876	-0.154013	NaN	0.115969	0.034223	0.206
County Code	0.204876	1.000000	-0.103036	NaN	-0.054974	-0.380206	0.12
Site Num	-0.154013	-0.103036	1.000000	NaN	-0.002499	-0.051343	-0.07
Parameter Code	NaN	NaN	NaN	NaN	NaN	NaN	
POC	0.115969	-0.054974	-0.002499	NaN	1.000000	0.035768	-0.03
Latitude	0.034223	-0.380206	-0.051343	NaN	0.035768	1.000000	0.08
Longitude	0.206468	0.122152	-0.079416	NaN	-0.033185	0.088812	1.000
Pollutant Standard	NaN	NaN	NaN	NaN	NaN	NaN	
Observation Count	-0.008786	-0.011672	0.000118	NaN	0.002065	0.012537	-0.012
Observation Percent	-0.008478	-0.011706	0.000122	NaN	0.003034	0.012198	-0.012
Arithmetic Mean	-0.032678	0.163731	0.015381	NaN	-0.027938	-0.394897	-0.047
1st Max Value	-0.049480	0.131917	0.026536	NaN	-0.020976	-0.364949	-0.07
1st Max Hour	0.010704	0.005709	0.007056	NaN	0.024351	-0.041678	-0.07
AQI	NaN	NaN	NaN	NaN	NaN	NaN	
Method Code	-0.044312	-0.041222	-0.024468	NaN	0.021729	-0.047091	-0.00

```
State Code                 0
County Code                0
Site Num                   0
Parameter Code             0
POC                        0
Latitude                   0
Longitude                  0
Datum                      0
Parameter Name             0
Sample Duration            0
Pollutant Standard     8105340
Date Local                 0
Units of Measure           0
Event Type             8047822
Observation Count          0
Observation Percent        0
Arithmetic Mean            0
1st Max Value              0
1st Max Hour               0
AQI                    8105340
Method Code                0
Method Name                0
Local Site Name       976015
Address                 1046
State Name                 0
County Name                0
City Name                  0
CBSA Name            1077210
Date of Last Change        0
Directory Path             0
dtype: int64
3973790
```



daily_RH_DP

```
(3855125, 30)
```

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longitude	Datum	Parameter Name	Sar Dura
0	1	53	1000	62201	1	31.0921	-87.5435	NAD83	Relative Humidity	1 H
1	1	53	1000	62201	1	31.0921	-87.5435	NAD83	Relative Humidity	1 H

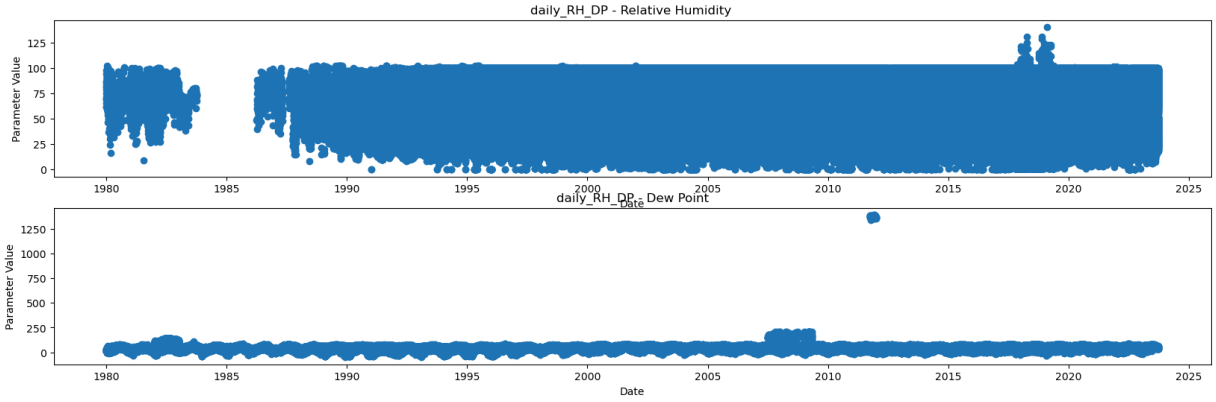
2 rows × 30 columns

```
<class 'pandas.core.frame.DataFrame'>
Index: 3855125 entries, 0 to 94571
Data columns (total 30 columns):
 #   Column               Dtype         
---  ------               -----         
 0   State Code           int64         
 1   County Code          int64         
 2   Site Num             int64         
 3   Parameter Code       int64         
 4   POC                  int64         
 5   Latitude             float64       
 6   Longitude            float64       
 7   Datum                object        
 8   Parameter Name       object        
 9   Sample Duration      object        
 10  Pollutant Standard   float64       
 11  Date Local           datetime64[ns]
 12  Units of Measure     object        
 13  Event Type           object        
 14  Observation Count    int64         
 15  Observation Percent  float64       
 16  Arithmetic Mean      float64       
 17  1st Max Value        float64       
 18  1st Max Hour         int64         
 19  AQI                  float64       
 20  Method Code          int64         
 21  Method Name          object        
 22  Local Site Name      object        
 23  Address              object        
 24  State Name           object        
 25  County Name          object        
 26  City Name            object        
 27  CBSA Name            object        
 28  Date of Last Change  datetime64[ns]
 29  Directory Path       object        
dtypes: datetime64[ns](2), float64(7), int64(8), object(13)
memory usage: 911.8+ MB
None
```

	State Code	County Code	Site Num	Parameter Code	POC	
count	3.855125e+06	3.855125e+06	3.855125e+06	3.855125e+06	3.855125e+06	3.855
mean	2.706695e+01	7.221856e+01	1.166253e+03	6.218884e+04	1.070239e+00	3.863
min	1.000000e+00	1.000000e+00	1.000000e+00	6.210300e+04	1.000000e+00	1.833
25%	8.000000e+00	2.100000e+01	7.000000e+00	6.220100e+04	1.000000e+00	3.494
50%	2.500000e+01	4.900000e+01	4.000000e+01	6.220100e+04	1.000000e+00	3.900
75%	4.400000e+01	1.010000e+02	1.011000e+03	6.220100e+04	1.000000e+00	4.214
max	8.000000e+01	5.100000e+02	9.997000e+03	6.220100e+04	8.000000e+00	6.709
std	1.801624e+01	7.543154e+01	2.313716e+03	3.230887e+01	2.949181e-01	5.182

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longit
State Code	1.000000	0.118574	-0.106488	-0.064383	-0.014979	0.135603	0.277
County Code	0.118574	1.000000	-0.103493	-0.204649	-0.001148	-0.273410	0.117
Site Num	-0.106488	-0.103493	1.000000	0.097212	-0.049676	-0.098110	-0.107
Parameter Code	-0.064383	-0.204649	0.097212	1.000000	0.074144	0.039433	-0.099
POC	-0.014979	-0.001148	-0.049676	0.074144	1.000000	-0.040734	-0.045
Latitude	0.135603	-0.273410	-0.098110	0.039433	-0.040734	1.000000	0.118
Longitude	0.277273	0.117823	-0.107250	-0.099232	-0.045607	0.118921	1.000
Pollutant Standard	NaN	NaN	NaN	NaN	NaN	NaN	
Observation Count	0.018719	0.054714	-0.208627	-0.061002	-0.181100	-0.027797	0.053
Observation Percent	0.012477	0.001681	-0.001768	0.038269	-0.005766	0.007419	-0.008
Arithmetic Mean	0.069715	0.043949	-0.042311	0.288289	0.015770	0.087427	0.209
1st Max Value	0.052974	0.008750	-0.046835	0.453390	0.009323	0.087228	0.17
1st Max Hour	0.015124	0.020854	-0.054281	-0.143694	-0.045473	0.015052	0.030
AQI	NaN	NaN	NaN	NaN	NaN	NaN	
Method Code	-0.030786	-0.058787	0.213791	0.050267	0.211353	0.031800	-0.070

```
State Code                 0
County Code                0
Site Num                   0
Parameter Code             0
POC                        0
Latitude                   0
Longitude                  0
Datum                      0
Parameter Name             0
Sample Duration            0
Pollutant Standard     3855125
Date Local                 0
Units of Measure           0
Event Type             3831416
Observation Count          0
Observation Percent        0
Arithmetic Mean            0
1st Max Value              0
1st Max Hour               0
AQI                    3855125
Method Code                0
Method Name                0
Local Site Name       338391
Address                 1391
State Name                 0
County Name                0
City Name                  0
CBSA Name             550884
Date of Last Change        0
Directory Path             0
dtype: int64
1308352
```



daily_NONOxNOy

```
(9183878, 30)
```

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longitude	Datum	Parameter Name	Sa Du
0	1	89	14	42601	1	34.68547	-86.58816	WGS84	Nitric oxide (NO)	1
1	1	89	14	42601	1	34.68547	-86.58816	WGS84	Nitric oxide (NO)	1

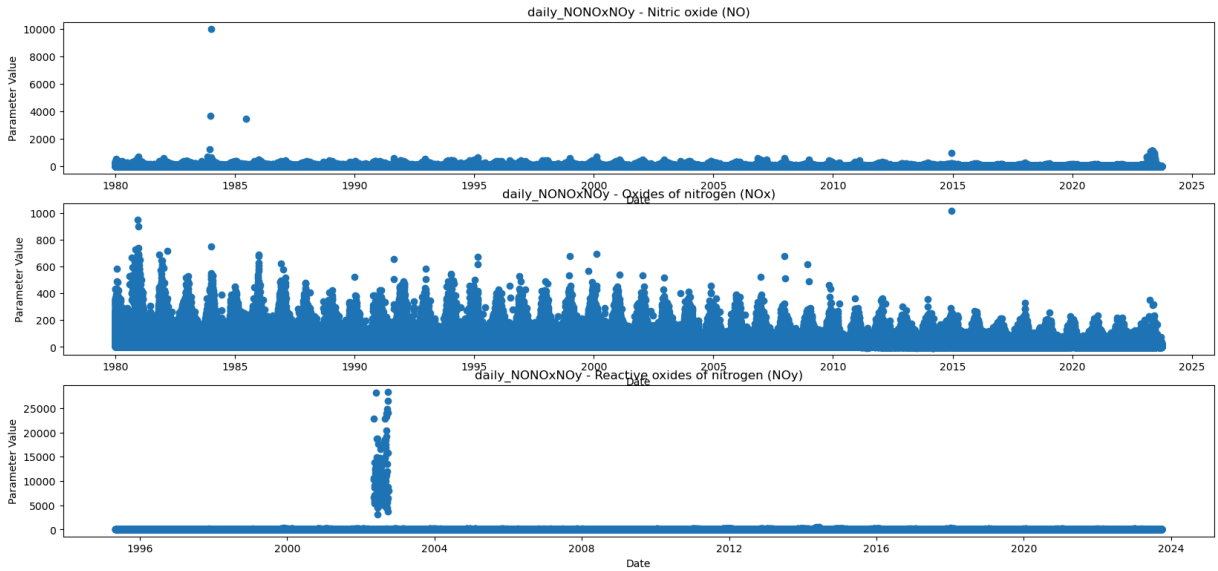
2 rows × 30 columns

```
<class 'pandas.core.frame.DataFrame'>
Index: 9183878 entries, 0 to 76666
Data columns (total 30 columns):
 #   Column               Dtype
---  ------               -----
 0   State Code           int64
 1   County Code          int64
 2   Site Num             int64
 3   Parameter Code       int64
 4   POC                  int64
 5   Latitude             float64
 6   Longitude            float64
 7   Datum                object
 8   Parameter Name       object
 9   Sample Duration      object
 10  Pollutant Standard   float64
 11  Date Local           datetime64[ns]
 12  Units of Measure     object
 13  Event Type           object
 14  Observation Count    int64
 15  Observation Percent  float64
 16  Arithmetic Mean      float64
 17  1st Max Value        float64
 18  1st Max Hour         int64
 19  AQI                  float64
 20  Method Code          int64
 21  Method Name          object
 22  Local Site Name      object
 23  Address              object
 24  State Name           object
 25  County Name          object
 26  City Name            object
 27  CBSA Name            object
 28  Date of Last Change  datetime64[ns]
 29  Directory Path       object
dtypes: datetime64[ns](2), float64(7), int64(8), object(13)
memory usage: 2.1+ GB
None
```

	State Code	County Code	Site Num	Parameter Code	POC	
count	9.183878e+06	9.183878e+06	9.183878e+06	9.183878e+06	9.183878e+06	9.183
mean	2.626103e+01	8.099170e+01	1.010753e+03	4.260182e+04	1.177985e+00	3.729
min	1.000000e+00	1.000000e+00	1.000000e+00	4.260000e+04	1.000000e+00	0.000
25%	6.000000e+00	2.500000e+01	8.000000e+00	4.260100e+04	1.000000e+00	3.394
50%	2.500000e+01	5.900000e+01	4.100000e+01	4.260100e+04	1.000000e+00	3.764
75%	4.200000e+01	9.900000e+01	1.013000e+03	4.260300e+04	1.000000e+00	4.073
max	8.000000e+01	8.100000e+02	9.997000e+03	4.260300e+04	9.000000e+00	7.020
std	1.811319e+01	9.565523e+01	1.964426e+03	1.057739e+00	5.918857e-01	4.734

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longit
State Code	1.000000	0.228599	-0.208781	-0.019962	-0.041889	0.118119	0.423
County Code	0.228599	1.000000	-0.090611	0.006813	-0.016096	-0.260338	0.100
Site Num	-0.208781	-0.090611	1.000000	-0.011188	0.042615	-0.074878	-0.145
Parameter Code	-0.019962	0.006813	-0.011188	1.000000	-0.118712	-0.050044	-0.063
POC	-0.041889	-0.016096	0.042615	-0.118712	1.000000	-0.010888	-0.053
Latitude	0.118119	-0.260338	-0.074878	-0.050044	-0.010888	1.000000	0.212
Longitude	0.423545	0.100047	-0.145259	-0.063441	-0.053671	0.212769	1.000
Pollutant Standard	NaN	NaN	NaN	NaN	NaN	NaN	
Observation Count	0.022558	0.003293	-0.017977	0.011623	-0.006234	0.060316	0.057
Observation Percent	0.019787	0.002512	-0.017483	0.011682	-0.006194	0.059364	0.054
Arithmetic Mean	-0.043257	-0.029802	0.005384	0.100311	0.018709	-0.003359	-0.013
1st Max Value	-0.037519	-0.025034	0.005940	0.065715	0.015782	-0.011749	-0.018
1st Max Hour	0.016658	0.012186	-0.020866	0.093506	-0.001994	0.008156	0.039
AQI	NaN	NaN	NaN	NaN	NaN	NaN	
Method Code	0.096888	-0.047863	0.031361	-0.307977	0.299999	0.117306	0.108

```
State Code                 0
County Code                0
Site Num                   0
Parameter Code             0
POC                        0
Latitude                   0
Longitude                  0
Datum                      0
Parameter Name             0
Sample Duration            0
Pollutant Standard     9183878
Date Local                 0
Units of Measure           0
Event Type             9100541
Observation Count          0
Observation Percent        0
Arithmetic Mean            0
1st Max Value              0
1st Max Hour               0
AQI                    9183878
Method Code                0
Method Name                0
Local Site Name       899972
Address                 2092
State Name                 0
County Name                0
City Name                  0
CBSA Name             599104
Date of Last Change        0
Directory Path             0
dtype: int64
3837872
```



daily_lead

```
(914238, 30)
```

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longitude	Datum	Parameter Name	
0	1	109	3	12128	1	31.790479	-85.978974	NAD83	Lead (TSP) STP	C
1	1	109	3	12128	1	31.790479	-85.978974	NAD83	Lead (TSP) STP	C

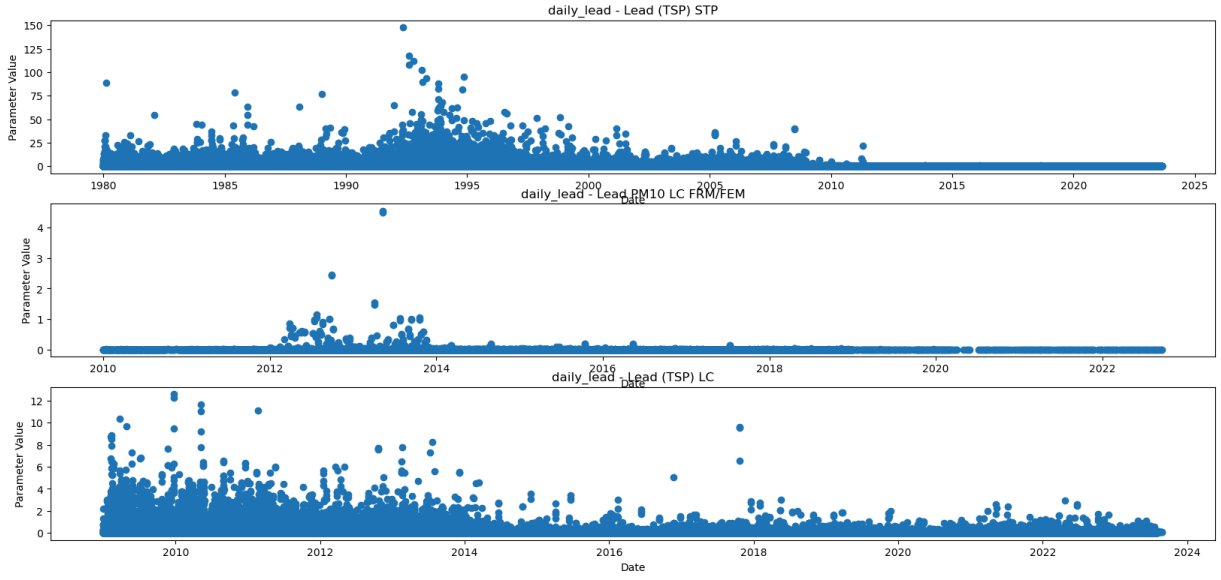
2 rows × 30 columns

```
<class 'pandas.core.frame.DataFrame'>
Index: 914238 entries, 0 to 26415
Data columns (total 30 columns):
 #   Column               Non-Null Count   Dtype         
---  ------               --------------   -----         
 0   State Code           914238 non-null  int64         
 1   County Code          914238 non-null  int64         
 2   Site Num             914238 non-null  int64         
 3   Parameter Code       914238 non-null  int64         
 4   POC                  914238 non-null  int64         
 5   Latitude             914238 non-null  float64       
 6   Longitude            914238 non-null  float64       
 7   Datum                914238 non-null  object        
 8   Parameter Name       914238 non-null  object        
 9   Sample Duration      914238 non-null  object        
 10  Pollutant Standard   914238 non-null  object        
 11  Date Local           914238 non-null  datetime64[ns]
 12  Units of Measure     914238 non-null  object        
 13  Event Type           37251 non-null   object        
 14  Observation Count    914238 non-null  int64         
 15  Observation Percent  914238 non-null  float64       
 16  Arithmetic Mean      914238 non-null  float64       
 17  1st Max Value        914238 non-null  float64       
 18  1st Max Hour         914238 non-null  int64         
 19  AQI                  0 non-null       float64       
 20  Method Code          914238 non-null  int64         
 21  Method Name          914238 non-null  object        
 22  Local Site Name      573995 non-null  object        
 23  Address              914189 non-null  object        
 24  State Name           914238 non-null  object        
 25  County Name          914238 non-null  object        
 26  City Name            914238 non-null  object        
 27  CBSA Name            866931 non-null  object        
 28  Date of Last Change  914238 non-null  datetime64[ns]
 29  Directory Path       914238 non-null  object        
dtypes: datetime64[ns](2), float64(6), int64(8), object(14)
memory usage: 216.2+ MB
None
```

	State Code	County Code	Site Num	Parameter Code	POC
count	914238.000000	914238.000000	914238.000000	914238.000000	914238.000000
mean	29.732644	84.700117	798.701120	13454.797780	1.545641
min	1.000000	1.000000	1.000000	12128.000000	1.000000
25%	17.000000	33.000000	8.000000	12128.000000	1.000000
50%	29.000000	71.000000	29.000000	12128.000000	1.000000
75%	42.000000	101.000000	1002.000000	12128.000000	2.000000
max	80.000000	810.000000	9997.000000	85129.000000	9.000000
std	15.919389	87.263064	1741.313684	8141.503264	1.268945

	State Code	County Code	Site Num	Parameter Code	POC	Latitude	Longit
State Code	1.000000	0.169195	-0.160012	-0.038607	-0.129730	-0.064834	0.450
County Code	0.169195	1.000000	-0.066370	-0.005902	0.007335	-0.124406	0.069
Site Num	-0.160012	-0.066370	1.000000	0.000011	0.018964	-0.025062	-0.102
Parameter Code	-0.038607	-0.005902	0.000011	1.000000	0.036882	0.045572	-0.002
POC	-0.129730	0.007335	0.018964	0.036882	1.000000	-0.089329	-0.227
Latitude	-0.064834	-0.124406	-0.025062	0.045572	-0.089329	1.000000	0.139
Longitude	0.450234	0.069723	-0.102655	-0.002619	-0.227321	0.139518	1.000
Observation Count	NaN	NaN	NaN	NaN	NaN	NaN	
Observation Percent	NaN	NaN	NaN	NaN	NaN	NaN	
Arithmetic Mean	0.000744	-0.010446	-0.033931	-0.030442	-0.044898	0.048649	-0.005
1st Max Value	0.000744	-0.010446	-0.033931	-0.030442	-0.044898	0.048649	-0.005
1st Max Hour	-0.030785	-0.014871	0.012693	-0.006798	0.052211	0.025551	0.003
AQI	NaN	NaN	NaN	NaN	NaN	NaN	
Method Code	-0.143946	-0.020190	0.027126	0.396728	0.061246	0.026919	-0.025

```
State Code                0
County Code               0
Site Num                  0
Parameter Code            0
POC                       0
Latitude                  0
Longitude                 0
Datum                     0
Parameter Name            0
Sample Duration           0
Pollutant Standard        0
Date Local                0
Units of Measure          0
Event Type           876987
Observation Count          0
Observation Percent        0
Arithmetic Mean            0
1st Max Value              0
1st Max Hour               0
AQI                   914238
Method Code                0
Method Name                0
Local Site Name       340243
Address                   49
State Name                 0
County Name                0
City Name                  0
CBSA Name              47307
Date of Last Change        0
Directory Path             0
dtype: int64
502692
```



- The purpose of the above preliminary analysis is to confirm consistency across datasets and to get some preliminary findings/ideas of what levels are available. All datasets have the same columns and rows.

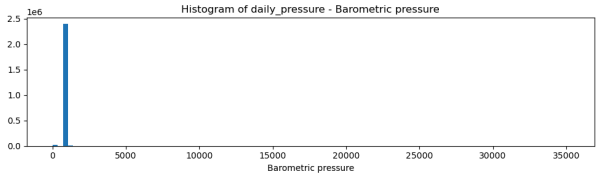
- There are some interesting groupings/trends with time depending on the levels, but at this point - there is no distinction by location so the plots do not give much useful information.
- Many columns contain redundant information. This dataset will be looked at by state and county rather than site.
- It's worth noting that daily HAPs and Daily VOCs have high numbers of parameters being monitored. These do not appear to show any noteworthy trends, but should be investigated further to determine whether they can be normalized and categorized as a whole for use in modeling later (i.e. normalized VOCs). Information from many of these levels is not available for all counties and dates. Although interesting, may not be possible to include for a robust model. These were removed from this file to make it easier to manage. But may be revisited at a later date.

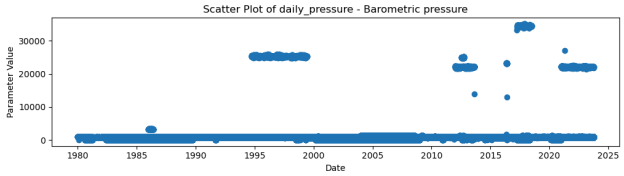
Next

- Need to look into what level of location granularity yields no duplicates.
- Only interested in the Arithmetic Mean for each parameter reading.
- AQI - Can be removed and obtained from AQI table later.
- Method Code and Method Name give the same information. Remove Method
- Local site name can be dropped. It should be covered by address.

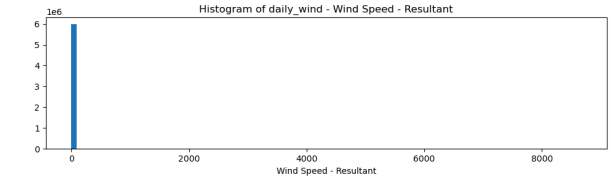
In [29]:
```
plotting_eda(df_daily_list)
```

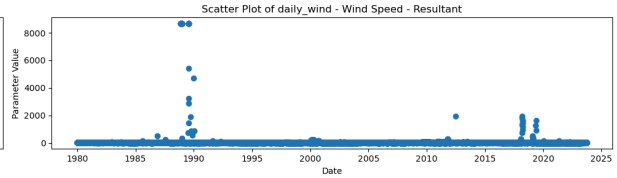
daily_pressure

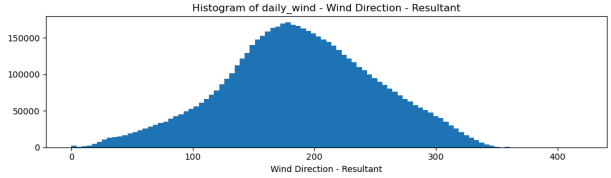


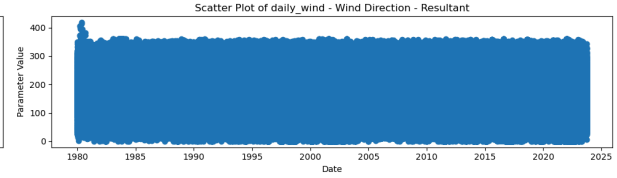


daily_wind

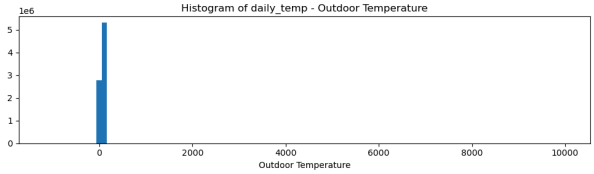


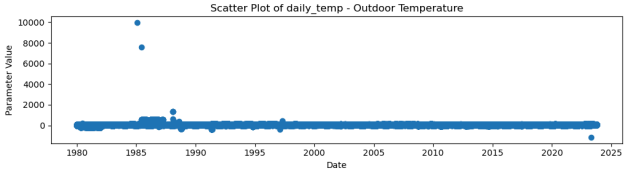




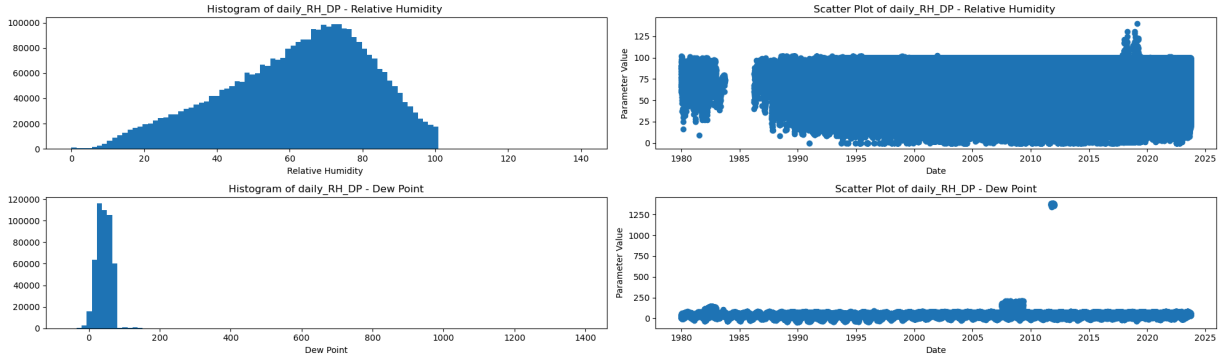


daily_temp

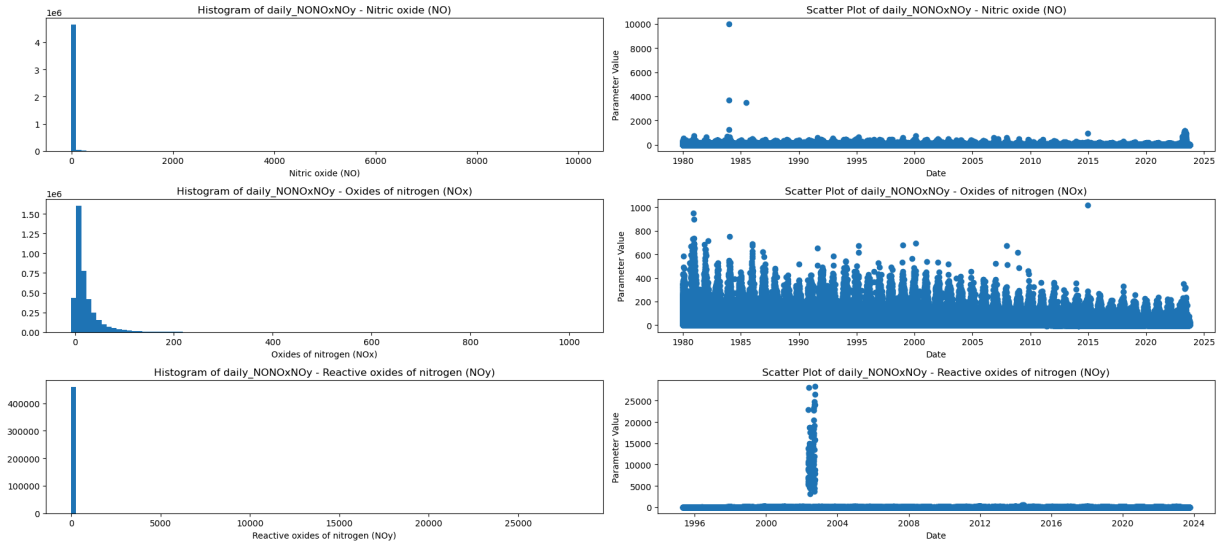




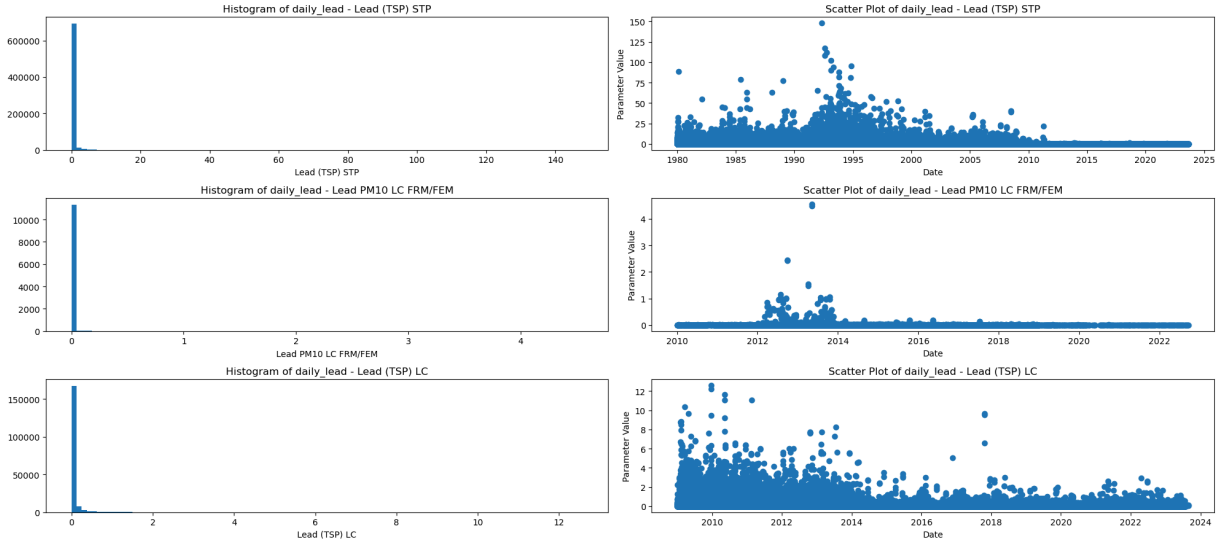
daily_RH_DP



daily_NONOxNOy



daily_lead



- Some of these levels show high skews. This indicates outliers in the dataset that will need to be reviewed by parameter. From the data dictionary, event type is also an indicator of abnormal levels.
- There are almost 200 parameters. Each will need to be investigated and cleaned.

In [30]:
```
# All columns are the same, use daily_lead as an example.
df_daily_list[0].columns
```

Out[30]:
```
Index(['State Code', 'County Code', 'Site Num', 'Parameter Code', 'POC',
       'Latitude', 'Longitude', 'Datum', 'Parameter Name', 'Sample Duratio
n',
       'Pollutant Standard', 'Date Local', 'Units of Measure', 'Event Typ
e',
       'Observation Count', 'Observation Percent', 'Arithmetic Mean',
       '1st Max Value', '1st Max Hour', 'AQI', 'Method Code', 'Method Nam
e',
       'Local Site Name', 'Address', 'State Name', 'County Name', 'City Nam
e',
       'CBSA Name', 'Date of Last Change', 'Directory Path'],
      dtype='object')
```

Create Combined Daily File

Useful for review together, to determine which columns are appropriate and complete manipulations at once. Some columns & information may be removed if already deemed not important for the model. Justification given and useful information extracted as dim Files for later use. The file is massive and memory needs to be considered in later manipulations.

In [31]:
```
# Create a combined file with only the needed
df_daily_combined_rough = pd.DataFrame()

for index, df in enumerate(df_daily_list):
    var = nameof(df)
    columns_keep = ['State Name', 'County Name', 'Address', 'State Name', 'C
                    'Observation Percent', 'Arithmetic Mean', 'Method Name',

    df_daily_combined_rough = pd.concat([df_daily_combined_rough, df[columns
```

In [32]:
```
# Add Month and Year Information similar to aqi
df_daily_combined_rough['Month'] = df_daily_combined_rough['Date Local'].dt.
df_daily_combined_rough['Year'] = df_daily_combined_rough['Date Local'].dt.y
df_daily_combined_rough.head(2)
```

Out[32]:

	State Name	County Name	Address	State Name	County Name	City Name	CBSA Name	Date Local	Parameter Name
0	Alabama	Escambia	Jack Springs Rd	Alabama	Escambia	Not in a city	NaN	2023-01-01	Barometric pressure
1	Alabama	Escambia	Jack Springs Rd	Alabama	Escambia	Not in a city	NaN	2023-01-02	Barometric pressure

In [33]:
```
print(nameof(df_daily_combined_rough))
display(df_daily_combined_rough.shape)
display(df_daily_combined_rough.head())
display(df_daily_combined_rough.info())
display(df_daily_combined_rough.isna().sum())
```

```
df_daily_combined_rough
(36560639, 19)
```

	State Name	County Name	Address	State Name	County Name	City Name	CBSA Name	Date Local	Parameter Name	P St
0	Alabama	Escambia	Jack Springs Rd	Alabama	Escambia	Not in a city	NaN	2023-01-01	Barometric pressure	
1	Alabama	Escambia	Jack Springs Rd	Alabama	Escambia	Not in a city	NaN	2023-01-02	Barometric pressure	
2	Alabama	Escambia	Jack Springs Rd	Alabama	Escambia	Not in a city	NaN	2023-01-03	Barometric pressure	
3	Alabama	Escambia	Jack Springs Rd	Alabama	Escambia	Not in a city	NaN	2023-01-04	Barometric pressure	
4	Alabama	Escambia	Jack Springs Rd	Alabama	Escambia	Not in a city	NaN	2023-01-05	Barometric pressure	

```
<class 'pandas.core.frame.DataFrame'>
Index: 36560639 entries, 0 to 26415
Data columns (total 19 columns):
 #   Column               Dtype         
---  ------               -----         
 0   State Name           object        
 1   County Name          object        
 2   Address              object        
 3   State Name           object        
 4   County Name          object        
 5   City Name            object        
 6   CBSA Name            object        
 7   Date Local           datetime64[ns]
 8   Parameter Name       object        
 9   Pollutant Standard   object        
 10  Units of Measure     object        
 11  Observation Count    int64         
 12  Observation Percent  float64       
 13  Arithmetic Mean      float64       
 14  Method Name          object        
 15  Event Type           object        
 16  Directory Path       object        
 17  Month                int32         
 18  Year                 int32         
dtypes: datetime64[ns](1), float64(2), int32(2), int64(1), object(13)
memory usage: 5.2+ GB
None
State Name                    0
County Name                   0
Address                    7716
State Name                    0
County Name                   0
City Name                     0
CBSA Name               3812909
Date Local                    0
Parameter Name                0
Pollutant Standard     35646401
Units of Measure              0
Observation Count             0
Observation Percent           0
Arithmetic Mean               0
Method Name                   0
Event Type             36257753
Directory Path                0
Month                         0
Year                          0
dtype: int64
```

In [34]:
```
# Check the all dataframes are included and expected numbers of rows match.
sum = 0

for i, l in enumerate(df_daily_list):
    sum = sum + l.shape[0]

if sum == df_daily_combined_rough.shape[0]:
    print(f'Total rows is {sum}, which matches the expected value. The conca
```

```
else:
    print('Unsuccessful concatenation')
```

Total rows is 36560639, which matches the expected value. The concatenation was successful.

In [35]:
```
# Check expected parameters and directory folders are present
df_daily_combined_rough[['Directory Path', 'Parameter Name']].value_counts()
```

Out[35]:
```
Directory Path
Parameter Name
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyTemp/      Outdoor Temperature                   8105340
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyWind/      Wind Direction - Resultant            6069851

Wind Speed - Resultant              6003263
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyNONOxNOy/  Nitric oxide (NO)                     4717938

Oxides of nitrogen (NOx)            4005338
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyRhDp/      Relative Humidity                     3376749
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyPressure/  Barometric pressure                   2428944
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyLead/      Lead (TSP) STP                         717289
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyRhDp/      Dew Point                              478376
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyNONOxNOy/  Reactive oxides of nitrogen (NOy)     460602
/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/data/dai
lyLead/      Lead (TSP) LC                          185415

Lead PM10 LC FRM/FEM                  11534
Name: count, dtype: int64
```

In [36]:
```
# Check only one set of units per level using daily lead as an example
display(df_daily_combined_rough.groupby(['Parameter Name', 'Units of Measure
df_daily_combined_rough.groupby(['Parameter Name', 'Method Name'])[['Paramet
```

		Parameter Name	Units of Measure
Parameter Name	**Units of Measure**		
Barometric pressure	**Millibars**	1	1
Dew Point	**Degrees Fahrenheit**	1	1
Lead (TSP) LC	**Micrograms/cubic meter (LC)**	1	1
Lead (TSP) STP	**Micrograms/cubic meter (25 C)**	1	1
Lead PM10 LC FRM/FEM	**Micrograms/cubic meter (LC)**	1	1
Nitric oxide (NO)	**Parts per billion**	1	1
Outdoor Temperature	**Degrees Fahrenheit**	1	1
Oxides of nitrogen (NOx)	**Parts per billion**	1	1
Reactive oxides of nitrogen (NOy)	**Parts per billion**	1	1
Relative Humidity	**Percent relative humidity**	1	1
Wind Direction - Resultant	**Degrees Compass**	1	1
Wind Speed - Resultant	**Knots**	1	1

Out[36]:

Parameter Name	Method Name	Parameter Name	Method Name
Barometric pressure	INSTRUMENTAL - ANEROID	1	1
	INSTRUMENTAL - BAROMETRIC PRESSURE TRANSDUCER	1	1
	INSTRUMENTAL - BAROMETRIC SENSOR	1	1
	INSTRUMENTAL - MERCURIAL	1	1
	Instrumental - Met One AIO2 Sonic Weather Sensor	1	1
...	...	...	...
Wind Speed - Resultant	Instrumental - RM Young Ultrasonic Anemometer Model 86004	1	1
	Instrumental - Vaisala WS425	1	1
	RM Young Ultrasonic Wind Sensor - Vector Average Data Logger	1	1
	RM Young Ultrasonic Wind Sensor model 85004 - Vector Average Data Logger	1	1
	Ultrasonic Wind Sensor MD1425A - Vector Average Data Logger	1	1

161 rows × 2 columns

- Expected directories and parameters are present.
- Pollutant Standard has the most nulls, but this is expected as they arent applicable for all parameters.
- Measurements appear to be consistent per parameter and are not needed for the model.
- Methods are not consistent and will be needed for the model. Chi squared testing useful to determine whether or not to be included in the model at a later date.
- Detail information such as measurement unit and pollutant standard by parameter should be sumarized in a dim table and removed from the modeling table.

In [37]:
```
df_daily_combined_working = df_daily_combined_rough.drop(columns = ['Polluta
```

df_daily_combined_working is ready for further analysis and EDA.
df_daily_combined_rough can be used to create some dim tables.

Dim Files

Dim files need to be created with certain inforamtion

From AQI Table

Information from the rough table, so it doesn't need to be included in the working table.

In [38]:
```
# Sentiments for AQI
dimAQUISentiment = daily_aqi_rough.groupby('AQI Category Sentiment')['AQI'].
dimAQUISentiment
```

Out[38]:

AQI Category Sentiment	min	max
Good	0	50
Hazardous	301	20646
Moderate	51	100
Unhealthy	151	200
Unhealthy for Sensitive Groups	101	150
Very Unhealthy	201	300

From Daily Combined Table

Information from the rough table, so it doesn't need to be included in the working table.

In [39]:
```
dimDirectory = df_daily_combined_rough.groupby('Parameter Name')[['Directory
dimDirectory.head()
```

Out[39]:

Parameter Name	Directory Path
Barometric pressure	/Users/brookehall/Desktop/Brainstation/Deliver...
Dew Point	/Users/brookehall/Desktop/Brainstation/Deliver...
Lead (TSP) LC	/Users/brookehall/Desktop/Brainstation/Deliver...
Lead (TSP) STP	/Users/brookehall/Desktop/Brainstation/Deliver...
Lead PM10 LC FRM/FEM	/Users/brookehall/Desktop/Brainstation/Deliver...

The dimAQISentiment table can be used to understand ranges for healthy/unhealthy...etc. This wont necessarily be dropped from the dataset at this time.

In [40]:
```
# Parameter Units Used in Measurements
dimParameterUnits = df_daily_combined_rough[['Parameter Name', 'Units of Mea
    ['Parameter Name', 'Units of Measure']).agg('first').reset_index().set_i
dimParameterUnits.head()
```

Out[40]:

Parameter Name	Units of Measure
Barometric pressure	Millibars
Dew Point	Degrees Fahrenheit
Lead (TSP) LC	Micrograms/cubic meter (LC)
Lead (TSP) STP	Micrograms/cubic meter (25 C)
Lead PM10 LC FRM/FEM	Micrograms/cubic meter (LC)

Therefore, units of measure will be summarized in this dim table and can be dropped from the file.

In [41]:
```
# PollutantStandards
dimPollutantStandard =  df_daily_combined_rough.groupby('Pollutant Standard'
dimPollutantStandard
```

Out[41]:

Pollutant Standard	Parameter Name
Lead 3-Month 2009	Lead (TSP) LC
Lead 3-Month PM10 Surrogate 2009	Lead PM10 LC FRM/FEM
Lead Quarterly 1978	Lead (TSP) STP

Pollutant Standards can be dropped from the dataset.

From External Table

In [43]:
```
Location = pd.read_csv(path+'data/aqs_sites.csv')
Location.head()
```

Out[43]:

	State Code	County Code	Site Number	Latitude	Longitude	Datum	Elevation	Land Use
0	01	1	1	32.437458	-86.472891	WGS84	64.0	RESIDENTIAL
1	01	1	2	32.428470	-86.443585	WGS84	0.0	AGRICULTURAL
2	01	1	3	32.332659	-86.791521	WGS84	41.0	FOREST
3	01	3	1	0.000000	0.000000	NAD27	0.0	UNKNOWN
4	01	3	2	30.552367	-87.706911	WGS84	0.0	COMMERCIAL

5 rows × 28 columns

Need to look for a unique identifier or combination of unique identifiers to join to another dataframe.

In [44]:
```
display(Location.shape)
display(len(Location['Site Number'].unique()))
display(Location.nunique())
display(Location.isna().sum())
```

```
(20832, 28)
1208
```

```
State Code                56
County Code              244
Site Number             1208
Latitude               19255
Longitude              19271
Datum                      4
Elevation               2393
Land Use                  10
Location Setting           4
Site Established Date     4115
Site Closed Date          2952
Met Site State Code         25
Met Site County Code        63
Met Site Site Number        78
Met Site Type                6
Met Site Distance          396
Met Site Direction          16
GMT Offset                   8
Owning Agency              794
Local Site Name           5704
Address                  20344
Zip Code                  5732
State Name                  56
County Name               1391
City Name                 3886
CBSA Name                  857
Tribe Name                 113
Extraction Date              1
dtype: int64
```

```
State Code                    0
County Code                   0
Site Number                   0
Latitude                      7
Longitude                     6
Datum                         0
Elevation                     0
Land Use                   1145
Location Setting           1012
Site Established Date         0
Site Closed Date           4870
Met Site State Code       20480
Met Site County Code      20480
Met Site Site Number      20480
Met Site Type             18784
Met Site Distance         20275
Met Site Direction        20320
GMT Offset                    0
Owning Agency                 0
Local Site Name           14942
Address                       3
Zip Code                  10127
State Name                    0
County Name                   0
City Name                     0
CBSA Name                  2614
Tribe Name                20576
Extraction Date               0
dtype: int64
```

There are no clear unique identfiers by site/monitor. The aqi dataset is by county. Select a Lat/Lon value by County. Even if some sights are slightly off, this is an acceptable degree of risk as it will still be located in the same county (close local area).

Notes:

- State and County are to be used for joins later. Codes would require less memory, but there are less County Codes than County Names. This may indicate duplicates exist and inconsistent naming or duplicate Codes.

In [45]:
```
mask1 = Location['Latitude'] != 0
mask2 = Location['Longitude'] != 0
mask3 = Location['Latitude'].isna() == False
mask4 = Location['Longitude'].isna() == False

Loc_no_null = Location[mask1 & mask2 & mask3 & mask4]
Loc_no_null.isna().sum()
```

Out[45]:
```
State Code                  0
County Code                 0
Site Number                 0
Latitude                    0
Longitude                   0
Datum                       0
Elevation                   0
Land Use                 1145
Location Setting         1011
Site Established Date       0
Site Closed Date         4851
Met Site State Code     19605
Met Site County Code    19605
Met Site Site Number    19605
Met Site Type           17911
Met Site Distance       19400
Met Site Direction      19445
GMT Offset                  0
Owning Agency               0
Local Site Name         14069
Address                     3
Zip Code                 9252
State Name                  0
County Name                 0
City Name                   0
CBSA Name                2571
Tribe Name              19703
Extraction Date             0
dtype: int64
```

Loc_no_null is cleaned without 0s or nas for Lat and Long

In [46]:
```
dimLocation = Loc_no_null[['State Code', 'County Code', 'State Name', 'Count
                           'Longitude']].groupby(['State Name', 'County Name'])
display(dimLocation.isna().sum())
dimLocation.head()
```

```
State Name     0
County Name    0
State Code     0
County Code    0
Latitude       0
Longitude      0
dtype: int64
```

Out[46]:

	State Name	County Name	State Code	County Code	Latitude	Longitude
0	Alabama	Autauga	01	1	32.437458	-86.472891
1	Alabama	Baldwin	01	3	30.552367	-87.706911
2	Alabama	Barbour	01	5	31.891272	-85.135210
3	Alabama	Bibb	01	7	32.904048	-87.233057
4	Alabama	Blount	01	9	33.957020	-86.464443

In [47]:
```
dimLocation[['County Name', 'County Code']].value_counts()
```

Out[47]:
```
County Name     County Code
Adams           1              7
Crawford        33             3
Clay            27             3
Adair           1              3
Cherokee        21             3
                              ..
Granite         39             1
Grand Traverse  55             1
Grand Forks     35             1
Grand           49             1
Yuma            27             1
Name: count, Length: 2060, dtype: int64
```

Therefore, dimLocation can be used as a dim table and saved later. This data can be removed from the current AQI and daily monitoring data as they will be joined by County and State. Codes appear to have duplicates. Referenced FIPS codes online and they don't appear to be accurate: https://transition.fcc.gov/oet/info/maps/census/fips/fips.txt

Therefore, even though County Codes require less memory, they appear to be less accurate and will not be used to join.

Save Rough Files

In [50]:
```
# Tables saved for further EDA
path = '/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/'
destination_path = path+'capstoneWorking/data/cleanData/'
destination_path2 = path+'capstoneWorking/data/workingData/'
```

In [51]:
```
#dimFiles
dimParameterUnits.to_csv(destination_path + 'dimParameterUnits.csv')
dimLocation.to_csv(destination_path + 'dimLocation.csv')
dimAQUISentiment.to_csv(destination_path + 'dimAQUISentiment.csv')
dimPollutantStandard.to_csv(destination_path + 'dimPollutionStandards.csv')
```

In [52]:
```
# Fact Files
df_daily_combined_working.to_csv(destination_path2 + 'dailyCombinedWorking.c
```

In [53]:
```
daily_aqi_working.to_csv(destination_path2 + 'aqiWorking.csv')
```

In [54]:
```
# Want to reset all variables to reduce load on the kernel for next steps. D
%reset
```

```
Once deleted, variables cannot be recovered. Proceed (y/[n])? y
```

Part 2: EDA

At this point, the following datasets are available.

- Fact files (more analysis needed):
  - aqiWorking.csv
  - dailyCombinedWorking.csv
- dim files - no extra steps needed at this point. But useful for additional information in storytelling, modeling and for providing context.
  - dimParameterUnits.csv
  - dimLocation.csv
  - dimAQUISentiment.csv
  - dimPollutionStandards.csv

there are two working files to be analyzed further and dim Files with useful information.

NOTES

- Event Types are Outliers. Remove.

In [55]:
```
import pandas as pd
import numpy as np
import datetime as dt
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec # for posititioning plots
import seaborn as sns
import plotly.express as px
%matplotlib inline
```

Functions

In [56]:
```
# Re-create variable name function

def nameof(var):
    '''
    Import a variable and return the name as a string
    '''

    for name, value in globals().items():
        if value is var:
            return name
```

EDA

In [60]:
```
# Tables saved for further EDA
path = '/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/'
```

```
dp = path+'capstoneWorking/data/cleanData/'
dp2 = path+'capstoneWorking/data/workingData/'
```

In [61]:
```
daily_aqi_working = pd.read_csv(dp2+'aqiWorking.csv', parse_dates = ['Date']
```

In [62]:
```
daily_combined_working = pd.read_csv(dp2+'dailyCombinedWorking.csv',
                                     parse_dates = ['Date Local'], low_memor
```

Daily Combined Dataset

Further cleaning needed

In [63]:
```
display(daily_combined_working.shape)
display(daily_combined_working.info())

pd.set_option('display.max_rows', None)
daily_combined_working.isna().sum()
```

```
(36560639, 18)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36560639 entries, 0 to 36560638
Data columns (total 18 columns):
 #   Column               Dtype
---  ------               -----
 0   Unnamed: 0           int64
 1   State Name           object
 2   County Name          object
 3   Address              object
 4   State Name.1         object
 5   County Name.1        object
 6   City Name            object
 7   CBSA Name            object
 8   Date Local           datetime64[ns]
 9   Parameter Name       object
 10  Observation Count    int64
 11  Observation Percent  float64
 12  Arithmetic Mean      float64
 13  Method Name          object
 14  Event Type           object
 15  Directory Path       object
 16  Month                int64
 17  Year                 int64
dtypes: datetime64[ns](1), float64(2), int64(4), object(11)
memory usage: 4.9+ GB
None
```

Out[63]:
```
Unnamed: 0                    0
State Name                    0
County Name                   0
Address                    7716
State Name.1                  0
County Name.1                 0
City Name                     0
CBSA Name                3812909
Date Local                    0
Parameter Name                0
Observation Count             0
Observation Percent           0
Arithmetic Mean               0
Method Name                   0
Event Type                36257753
Directory Path                0
Month                         0
Year                          0
dtype: int64
```

Investigate Event Type.

In [64]:
```
daily_combined_working['Event Type'].value_counts()
```

Out[64]:
```
Event Type
Included    302514
Excluded       372
Name: count, dtype: int64
```

Based on the data dictionary events are abnormal environmental events i.e. wilfire. Included/excluded indicate whether the results are included/excluded in the summary. These result in duplicate entries for the data. They do not represent the dataset adequately. Both will be excluded. Null datapoints in this case are desired.

In [65]:
```
# Check if there are any duplicates by day, parameter & location and na valu
dup = daily_combined_working[['State Name', 'County Name', 'Date Local', 'Pa
print(f'There are {dup} duplicates by County/State/Day/Parameter')
```

```
There are 16026069 duplicates by County/State/Day/Parameter
```

In [66]:
```
# Look into Event Type = Excluded means there was an extenuating circumstanc
mask1 = daily_combined_working['Event Type'] != 'Excluded'
mask2 = daily_combined_working['Event Type'] != 'Included'

daily_combined_working_new = daily_combined_working[mask1 & mask2]
daily_combined_working_new.shape
```

Out[66]:
```
(36257753, 18)
```

In [67]:
```
# Check if there are any duplicates by day, parameter & location and na valu
dup = daily_combined_working_new[['State Name', 'County Name', 'Date Local',
print(f'There are {dup} duplicates by County/State/Day/Parameter')
```

```
There are 15875376 duplicates by County/State/Day/Parameter
```

Slight improvement. Review at method level.

In [68]:
```
# Check if there are any duplicates by day, parameter & location and na valu
dup = daily_combined_working_new[['State Name', 'County Name', 'Date Local',
print(f'There are {dup} duplicates by County/State/Day/Parameter/Method')
```

```
There are 14179389 duplicates by County/State/Day/Parameter/Method
```

In [69]:
```
# Check if there are any duplicates by day, parameter & location and na valu
dup = daily_combined_working_new[['State Name', 'County Name', 'City Name',
print(f'There are {dup} duplicates by City Name/County/State/Day/Parameter/M
```

```
There are 6427296 duplicates by City Name/County/State/Day/Parameter/Method
```

In [70]:
```
# Check if there are any duplicates by day, parameter & location and na valu
dup = daily_combined_working_new[['State Name', 'County Name', 'City Name',
                                  'Date Local', 'Parameter Name', 'Method Na
print(f'There are {dup} duplicates by CBSA/City Name/County/State/Day/Parame
```

```
There are 6427296 duplicates by CBSA/City Name/County/State/Day/Parameter/Me
thod
```

Create Grouping by City, State, Date, Parameter and Method. Mean. Method will need to be checked at a later date for significance.

In [71]:
```
daily_combined_group = daily_combined_working_new.groupby(
                    ['State Name', 'Date Local', 'Parameter Name', 'Method N
                    'Observation Percent', 'Arithmetic Mean']].agg('mean').r
```

In [72]:
```
daily_combined_group.shape
```

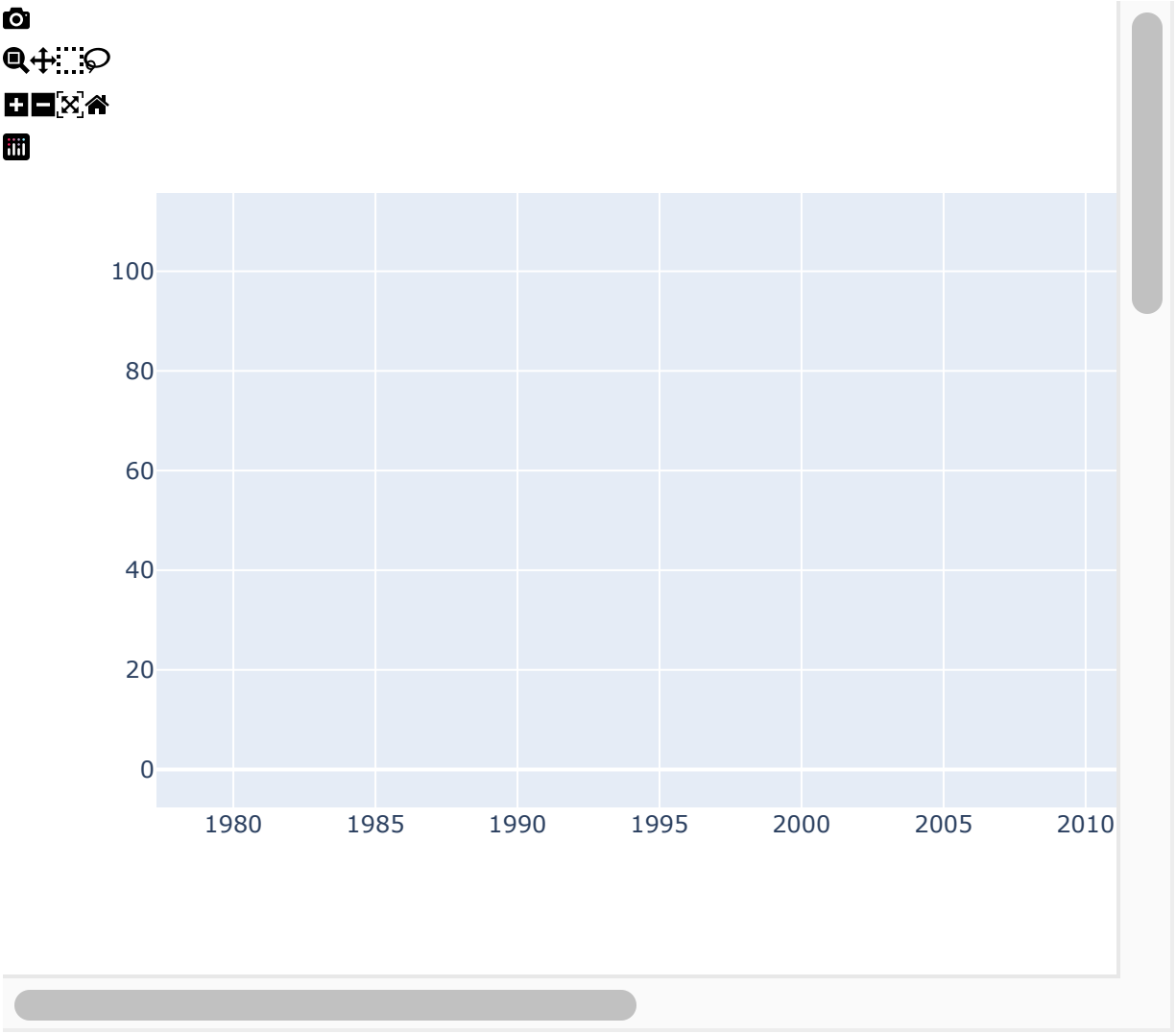
Out[72]:
```
(6138704, 7)
```

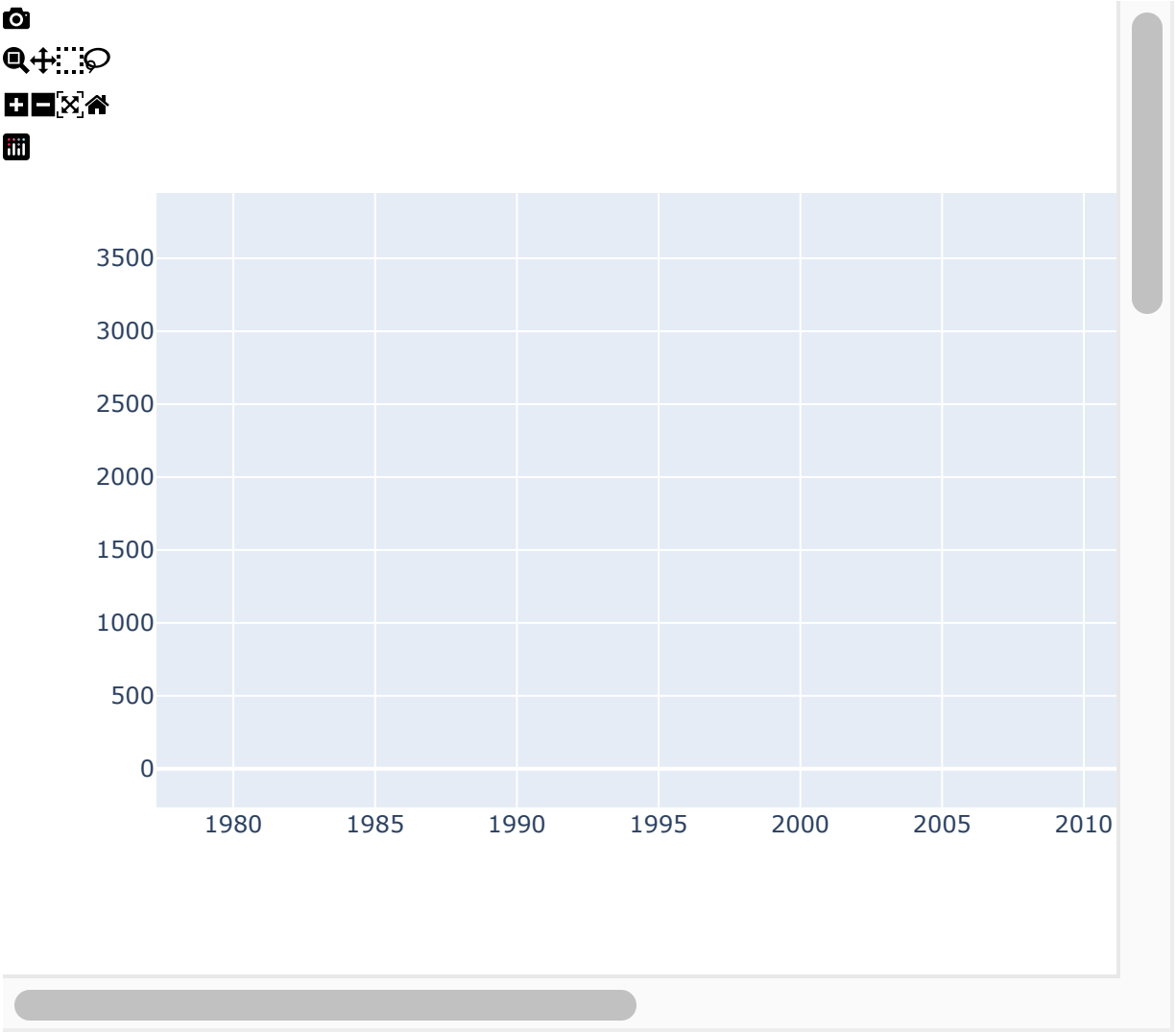
In [73]:
```
# Check if there are any duplicates by day, parameter & location and na valu
dup = daily_combined_group[['State Name', 'Date Local', 'Parameter Name', 'M
print(f'There are {dup} duplicates by State/Day/Parameter/Method')
```

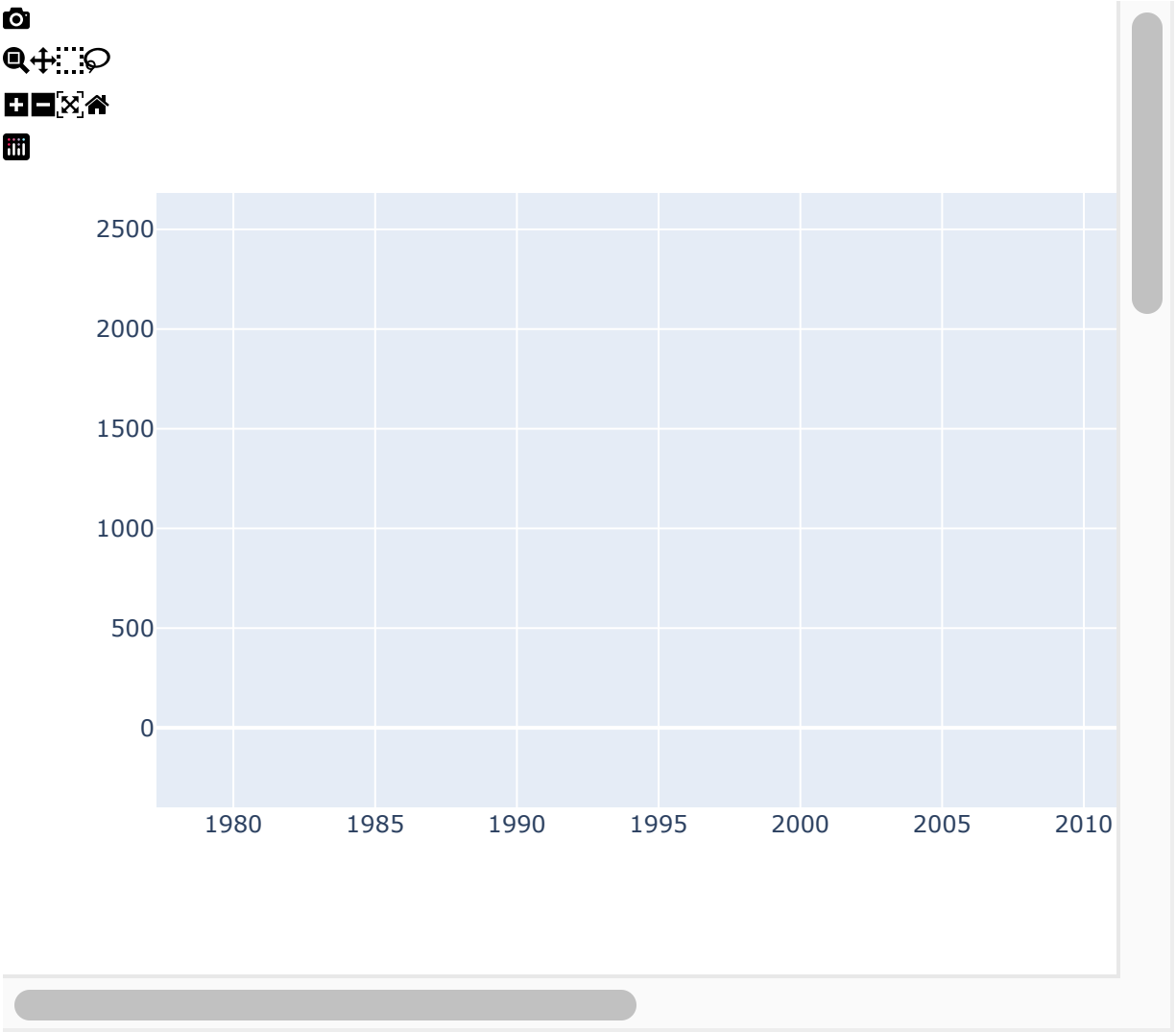
```
There are 0 duplicates by State/Day/Parameter/Method
```

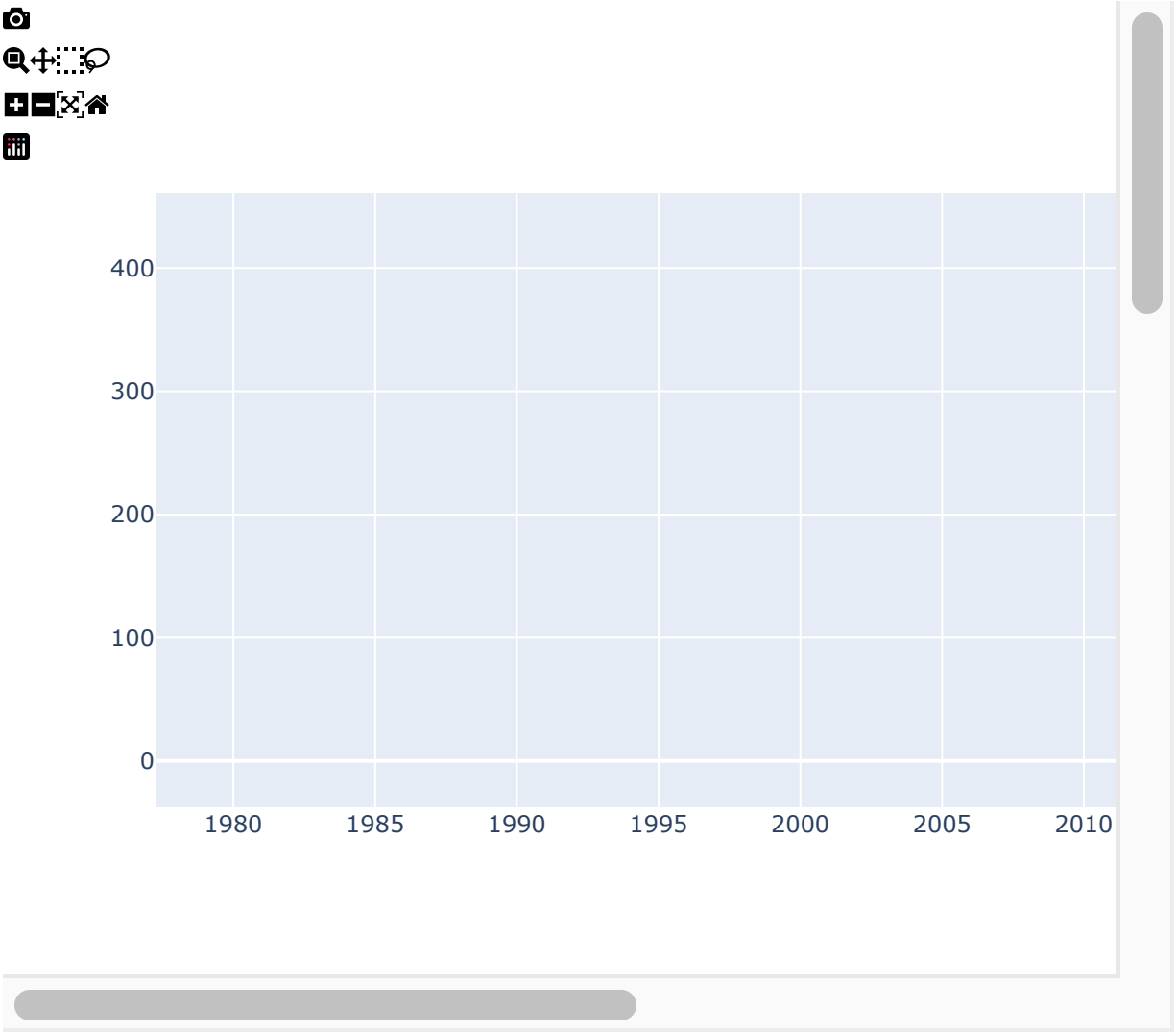
There are no duplicates at the level of the city. This is acceptible. As the data is to be looked at by state.

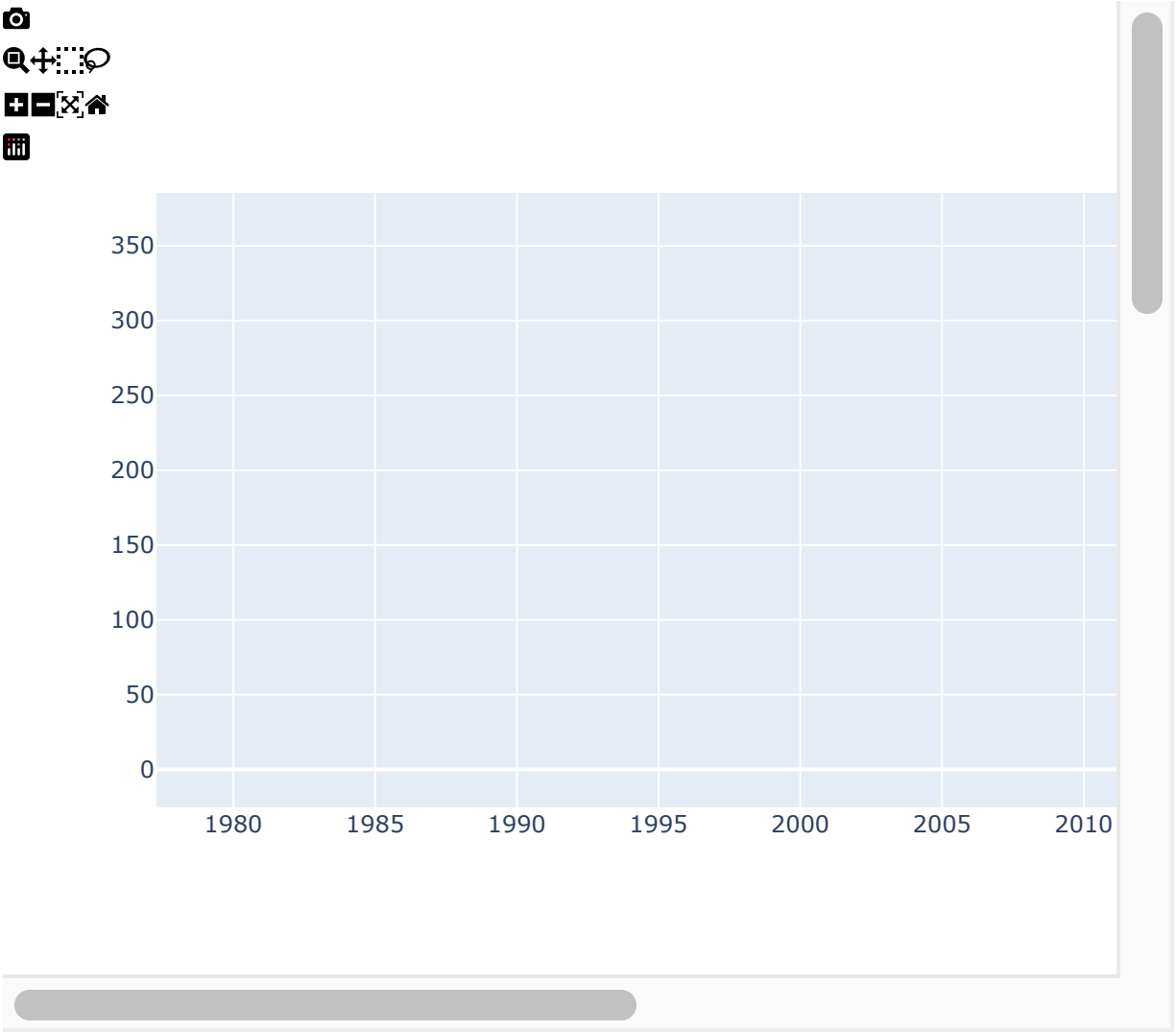
In [74]:
```
list_param = daily_combined_group['Parameter Name'].unique()
len_list_param = len(list_param)

for index, parameter in enumerate(list_param):
    mask = daily_combined_group['Parameter Name'] == parameter

    plt.figure()
    fig = px.scatter(daily_combined_group[mask], x="Date Local", y="Arithmet
                     title = parameter, width=1000,height=500)
    fig.show()
```

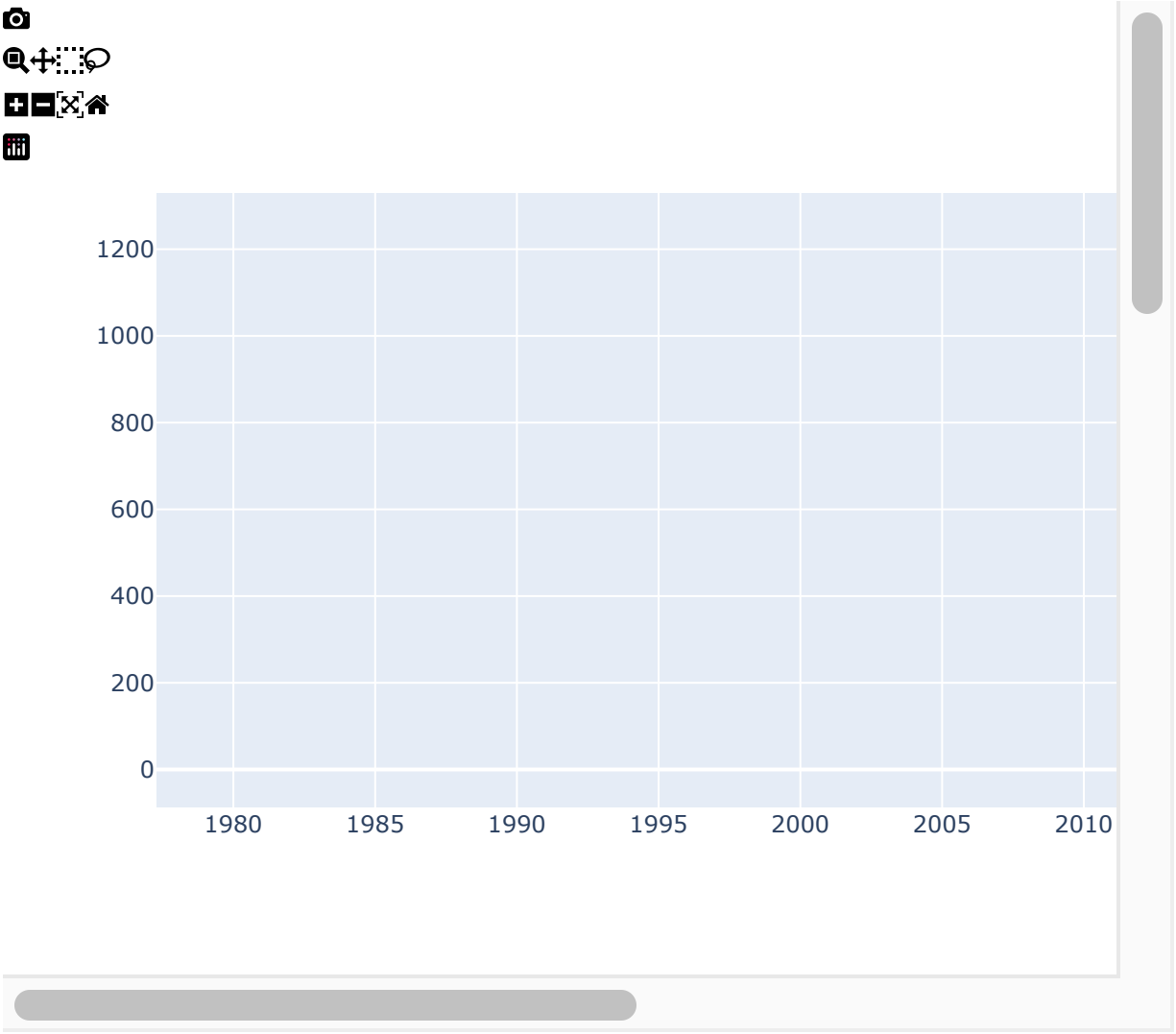

100
80
60
40
20
0
1980
1985
1990
1995
2000
2005
2010

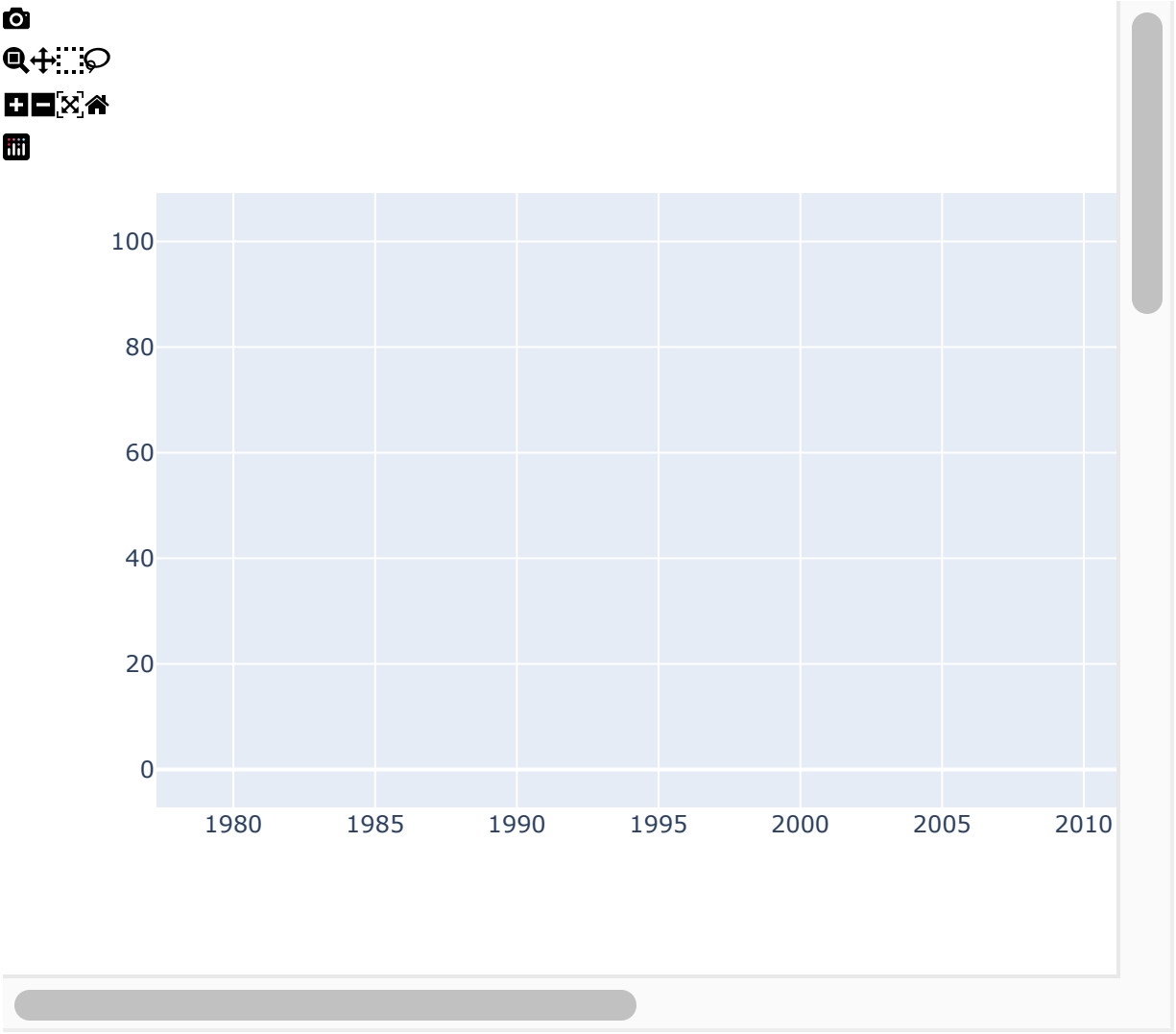

3500
3000
2500
2000
1500
1000
500
0
1980
1985
1990
1995
2000
2005
2010

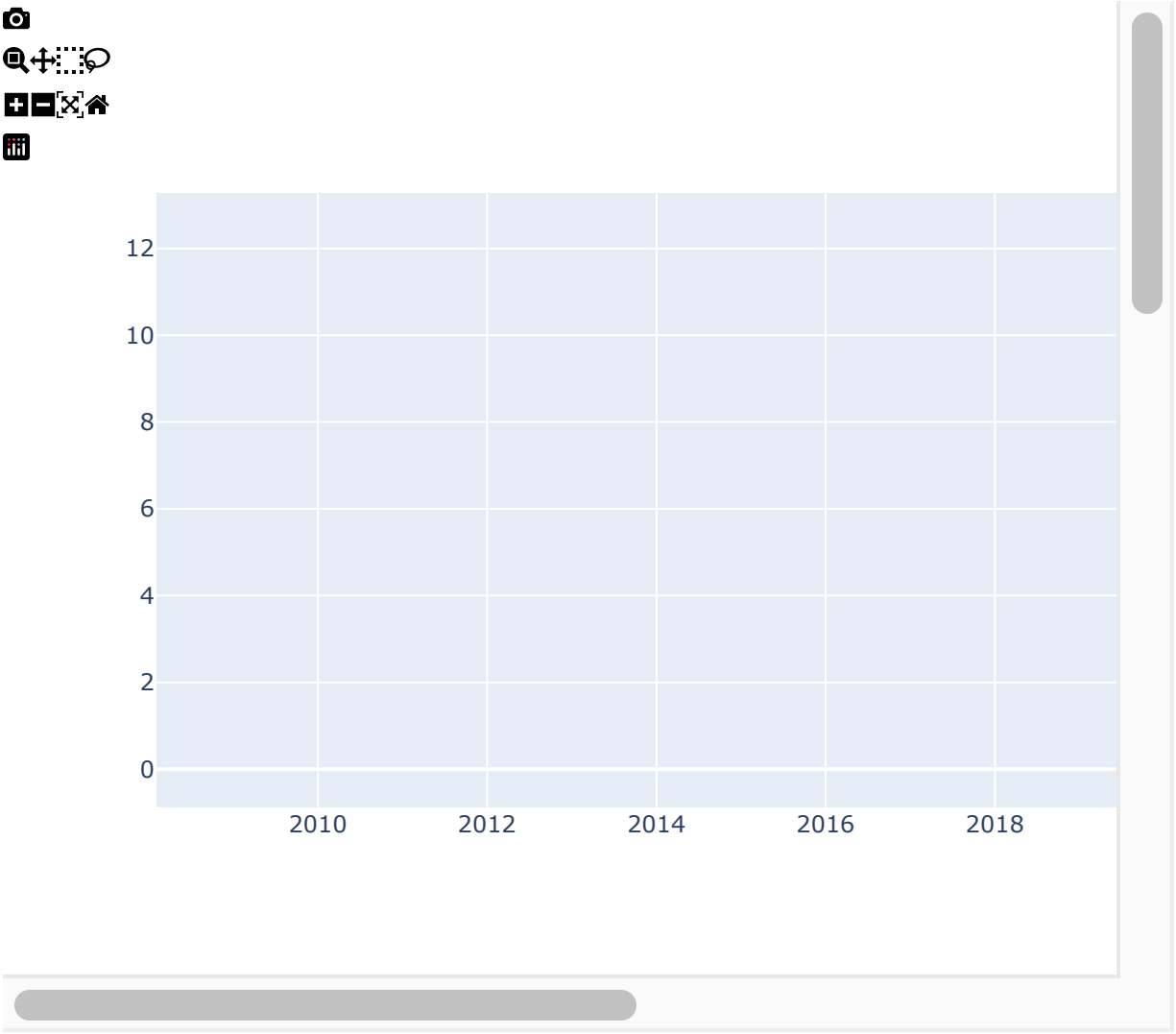

2500
2000
1500
1000
500
0
1980
1985
1990
1995
2000
2005
2010

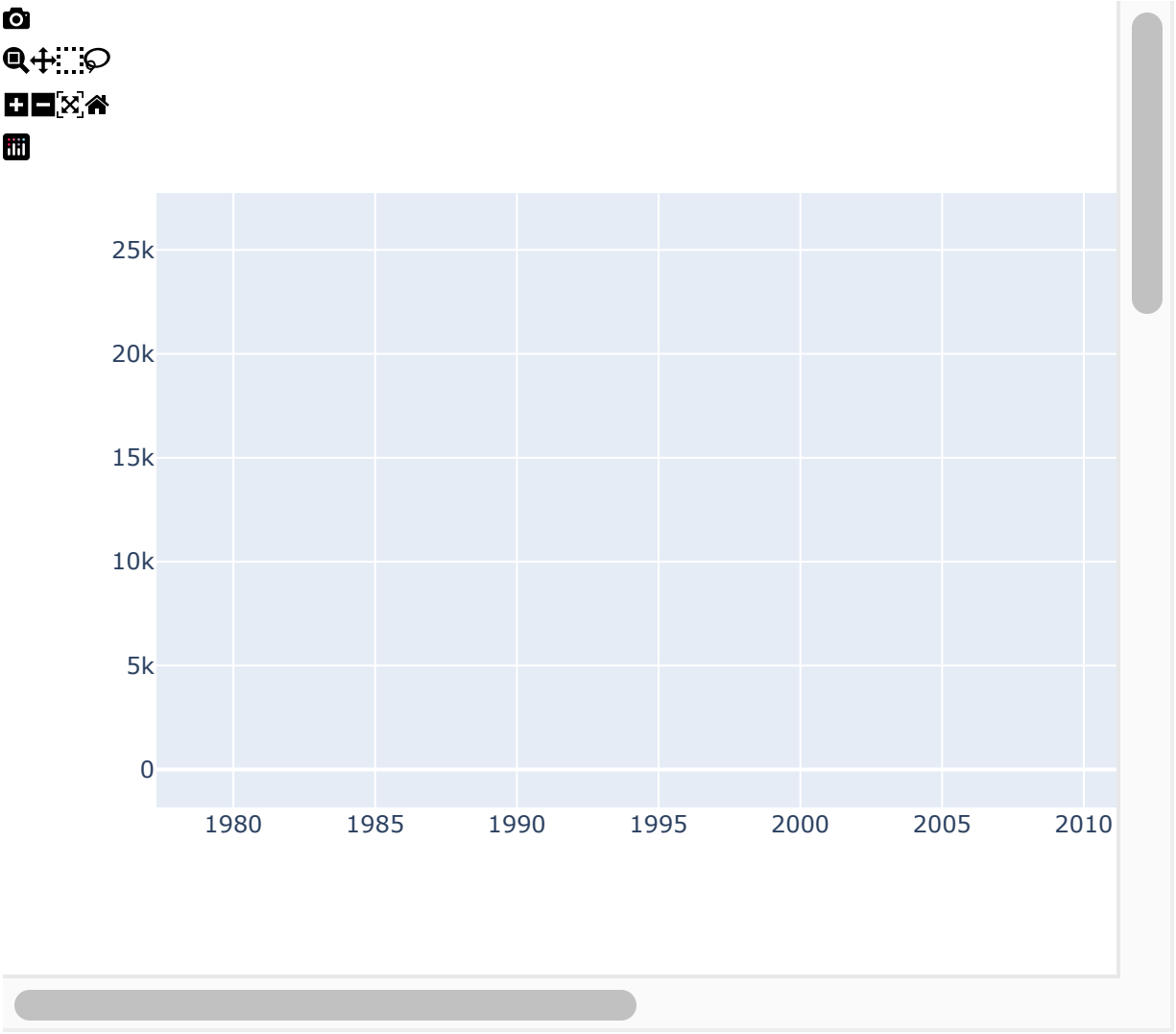

400
300
200
100
0
1980
1985
1990
1995
2000
2005
2010

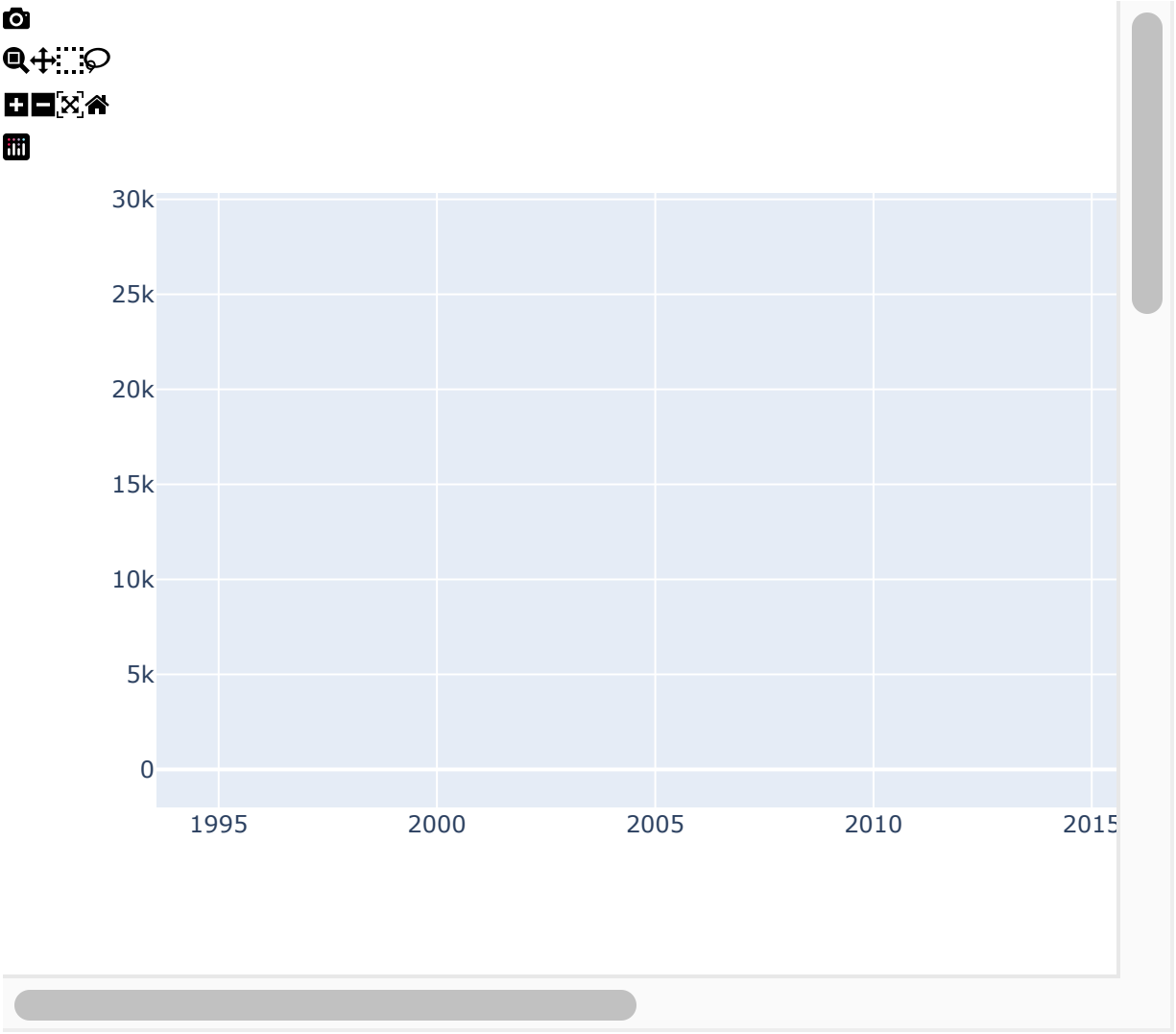

350
300
250
200
150
100
50
0
1980
1985
1990
1995
2000
2005
2010

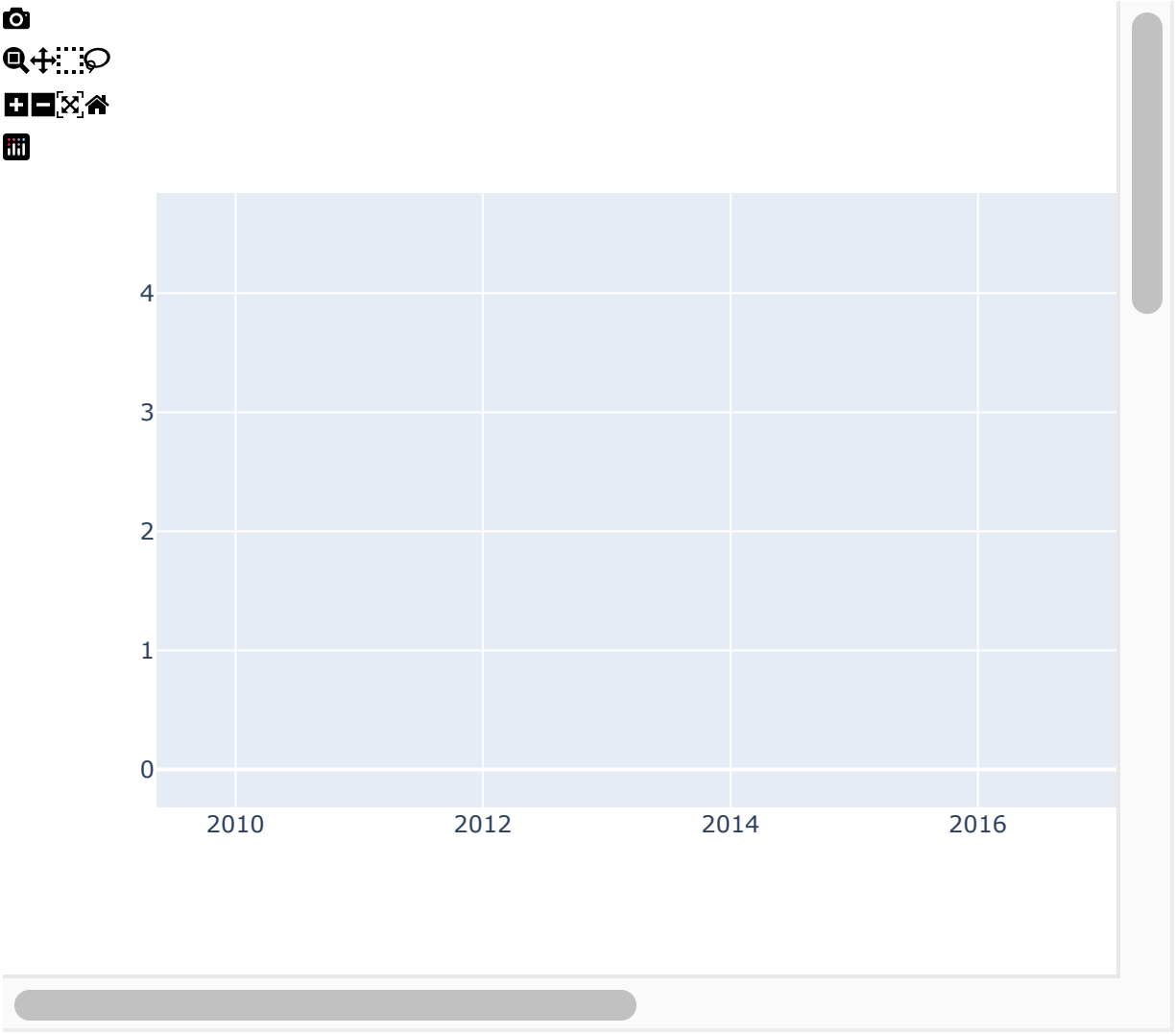

1200
1000
800
600
400
200
0
1980
1985
1990
1995
2000
2005
2010

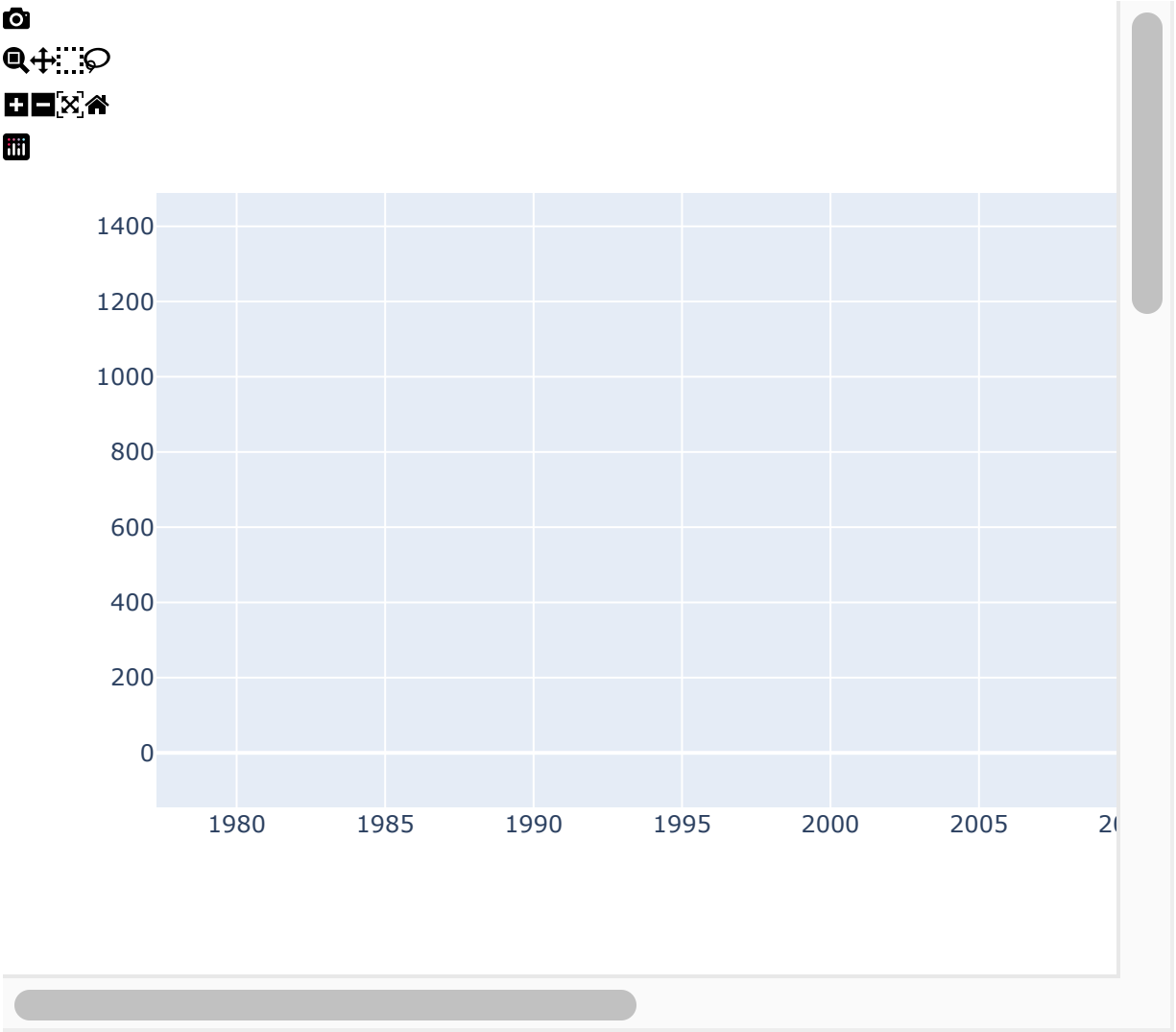

100
80
60
40
20
0
1980
1985
1990
1995
2000
2005
2010


12
10
8
6
4
2
0
2010
2012
2014
2016
2018


25k
20k
15k
10k
5k
0
1980
1985
1990
1995
2000
2005
2010


30k
25k
20k
15k
10k
5k
0
1995
2000
2005
2010
2015


4
3
2
1
0
2010
2012
2014
2016




```
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
<Figure size 640x480 with 0 Axes>
```

There's lots of cyclical variation by state. There are spikes in certain states, likely linked to weather events. Location will be important to include.

In [83]:
```
# Fact Files
dp2 = path+'capstoneWorking/data/workingData/'

daily_combined_group.to_csv(dp2 + 'combinedReduced.csv')
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[83], line 4
      1 # Fact Files
      2 dp2 = path+'capstoneWorking/data/workingData/'
----> 4 daily_combined_group.to_csv(dp2 + 'combinedReduced.csv')

NameError: name 'daily_combined_group' is not defined
```

In [77]:
```
%reset
```

```
Once deleted, variables cannot be recovered. Proceed (y/[n])? y
```

Another Variable Reset to be Able to Handle the Combined File

In [78]:
```
import pandas as pd
import numpy as np
import datetime as dt
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec # for posititioning plots
import seaborn as sns
import plotly.express as px
%matplotlib inline
```

In [84]:
```
path = '/Users/brookehall/Desktop/Brainstation/Deliverables/CapstoneFolder/'
dp = 'capstoneWorking/data/cleanData/'
dp2 = 'capstoneWorking/data/workingData/'
```

In [86]:
```
daily_combined_grouped = pd.read_csv(path + dp2 + 'combinedReduced.csv')
```

In [89]:
```
daily_aqi_working = pd.read_csv(path+dp2+'aqiWorking.csv', parse_dates = ['D
```

In [90]:
```
daily_combined_grouped.shape
```

Out[90]:
```
(6138704, 8)
```

AQI Findings of Sentiment and Defining Parameter

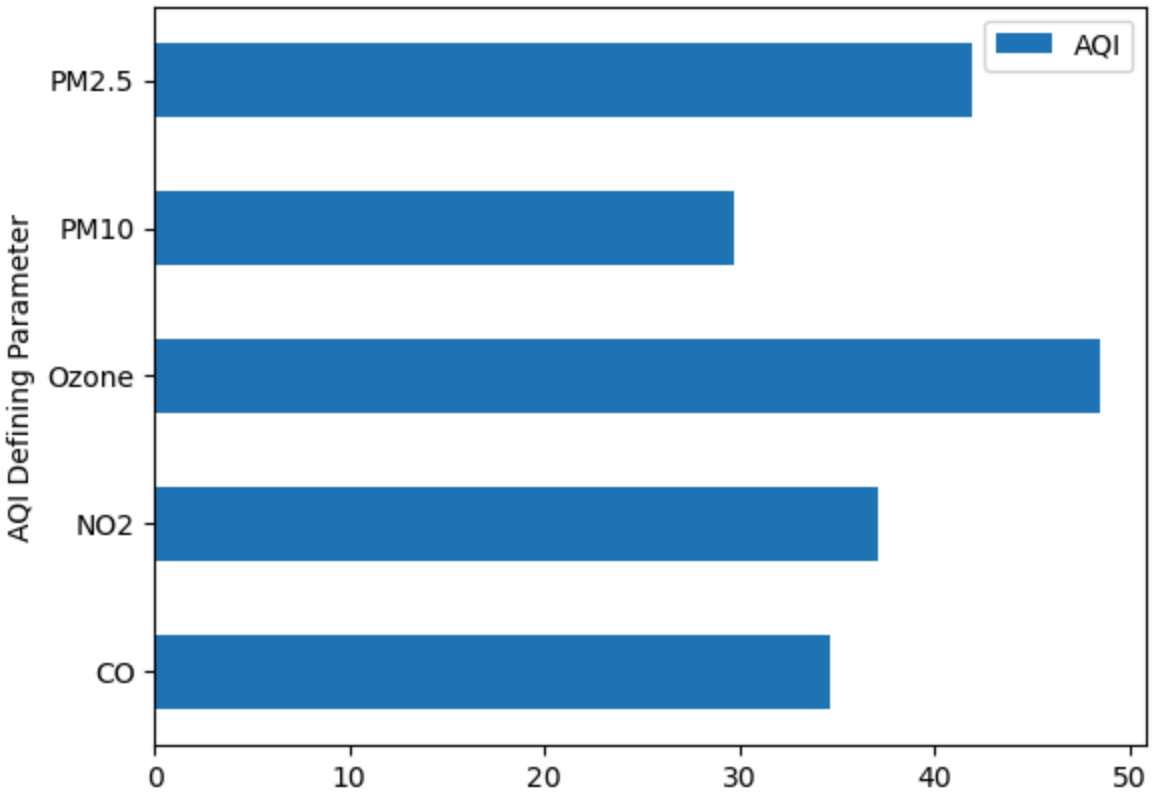
In [91]:
```
daily_aqi_working = daily_aqi_working.drop(columns = ['Unnamed: 0'])
daily_aqi_working.head()
```

Out[91]:

	State Name	County Name	Date	AQI	AQI Category Sentiment	AQI Defining Parameter	Month	Year
0	Alabama	Baldwin	2007-01-03	55	Moderate	PM2.5	1	2007
1	Alabama	Baldwin	2007-01-06	23	Good	PM2.5	1	2007
2	Alabama	Baldwin	2007-01-09	13	Good	PM2.5	1	2007
3	Alabama	Baldwin	2007-01-12	25	Good	PM2.5	1	2007
4	Alabama	Baldwin	2007-01-15	17	Good	PM2.5	1	2007

In [92]:
```
daily_aqi_working[['AQI', 'AQI Defining Parameter']].groupby('AQI Defining P
```

Out[92]: <Axes: ylabel='AQI Defining Parameter'>



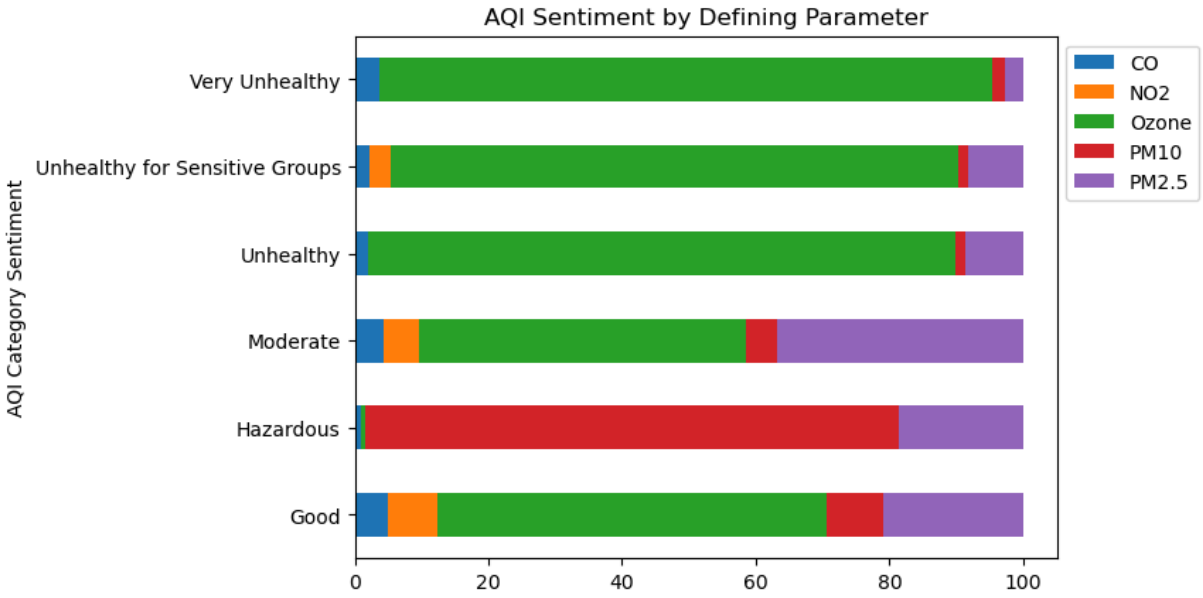
Ozone as the key variable appears to result in the highest AQI on average.

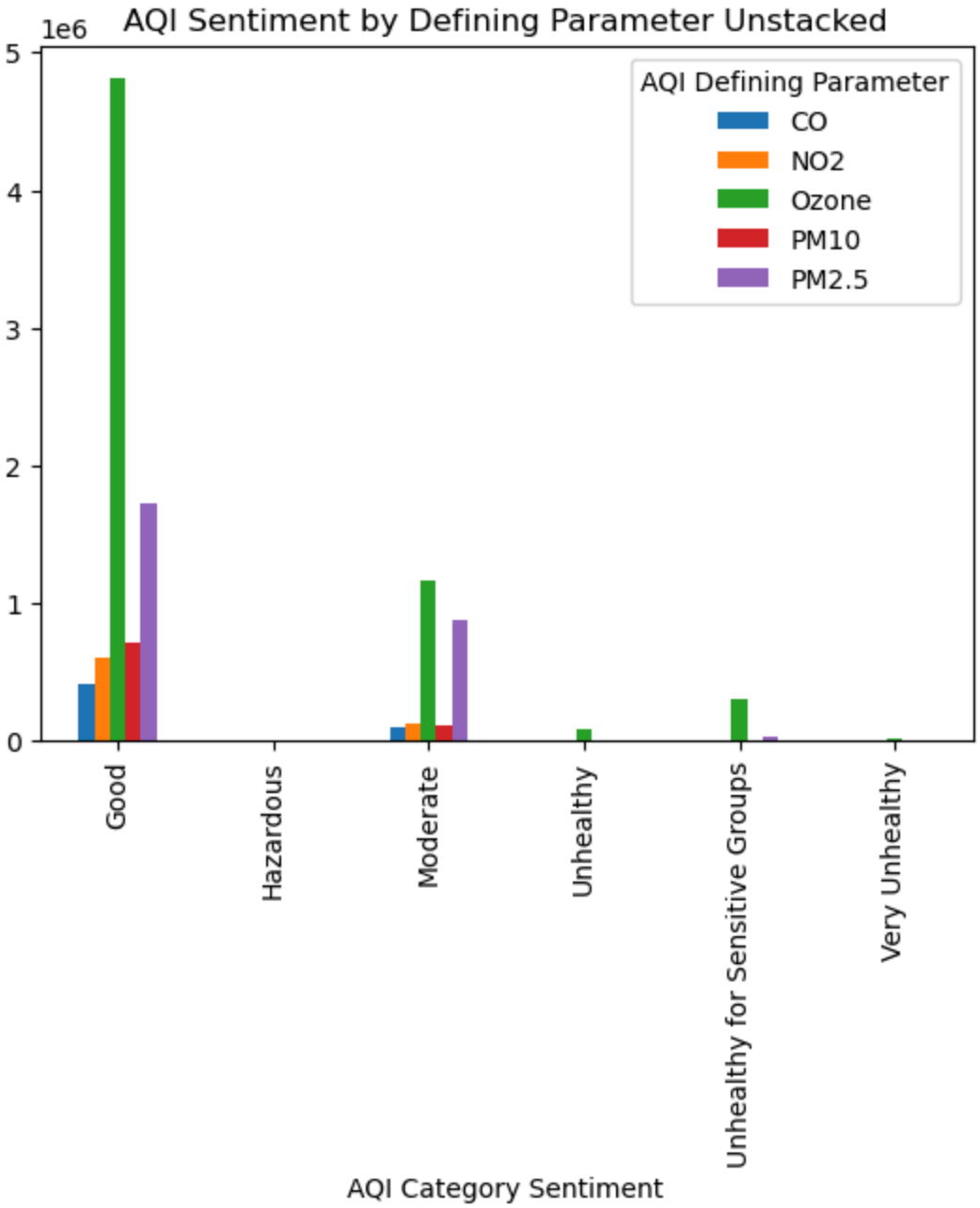
In [145…
```
def plot_dummy_drop_selection(category):

    '''
    Compute plots of the category (column name) input

    Parameters
    ----------
```

```
    ratio, value_counts

    Returns
    -------
    Two plots of value count and ratio WNV.

    '''

    ## Variable
    # create calculation for rate data to plot
    rate_data = (daily_aqi_working.groupby([category, 'AQI Defining Paramete

    ## Plots both rate data and data by count to get good representation.
    sort_order = ['Good', 'Moderate',  'Unhealthy for Sensitive Groups', 'Ur
    #rate_data[category] = pd.Categorical(rate_data[category], categories =

    rate_data.unstack().plot(kind = 'barh', stacked = True, title = 'AQI Ser
    plt.legend(bbox_to_anchor=(1.0, 1.0))

    daily_aqi_working.groupby([category])['AQI Defining Parameter'].value_co
    plt.title('AQI Sentiment by Defining Parameter Unstacked')
```

In [146…

```
plot_dummy_drop_selection('AQI Category Sentiment')
```





Group AQI by State

To be joined with the rest of the data to review relationships.

In [95]:
```
aqi_state = daily_aqi_working.groupby(['State Name', 'Date'])[['AQI']].agg('
```

Complete a pivot table for correlation review

For right now, assume method contribution is negligible and remove it.

In [96]:
```
daily_combined_grouped_new = daily_combined_grouped.groupby(
                    ['State Name', 'Date Local', 'Parameter Name'])[['Observ
                    'Observation Percent', 'Arithmetic Mean']].agg('mean').r
```

In [97]:
```
daily_combined_grouped_new.rename(columns = {'Arithmetic Mean': 'AM'}, inpla
```

In [98]:
```
daily_combined_parameter_pivot = daily_combined_grouped_new.pivot(index = ['
                                 columns = ['Parameter Name'], values = ['AM']).
```

In [99]:
```
display(daily_combined_parameter_pivot.info())
display(daily_combined_parameter_pivot.shape)
display(daily_combined_parameter_pivot.isna().sum())
display(daily_combined_parameter_pivot.head())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 735840 entries, 0 to 735839
Data columns (total 14 columns):
 #   Column                                    Non-Null Count   Dtype  
---  ------                                    --------------   -----  
 0   (Date Local, )                            735840 non-null  object 
 1   (State Name, )                            735840 non-null  object 
 2   (AM, Barometric pressure)                 374303 non-null  float64
 3   (AM, Dew Point)                           143603 non-null  float64
 4   (AM, Lead (TSP) LC)                       34086 non-null   float64
 5   (AM, Lead (TSP) STP)                      99561 non-null   float64
 6   (AM, Lead PM10 LC FRM/FEM)                7681 non-null    float64
 7   (AM, Nitric oxide (NO))                   590102 non-null  float64
 8   (AM, Outdoor Temperature)                 593356 non-null  float64
 9   (AM, Oxides of nitrogen (NOx))            532447 non-null  float64
 10  (AM, Reactive oxides of nitrogen (NOy))   253017 non-null  float64
 11  (AM, Relative Humidity )                  479451 non-null  float64
 12  (AM, Wind Direction - Resultant)          488241 non-null  float64
 13  (AM, Wind Speed - Resultant)              483011 non-null  float64
dtypes: float64(12), object(2)
memory usage: 78.6+ MB
None
(735840, 14)
            Parameter Name
Date Local                                          0
State Name                                          0
AM          Barometric pressure                361537
            Dew Point                          592237
            Lead (TSP) LC                      701754
            Lead (TSP) STP                     636279
            Lead PM10 LC FRM/FEM               728159
            Nitric oxide (NO)                  145738
            Outdoor Temperature                142484
            Oxides of nitrogen (NOx)           203393
            Reactive oxides of nitrogen (NOy)  482823
            Relative Humidity                  256389
            Wind Direction - Resultant         247599
            Wind Speed - Resultant             252829
dtype: int64
```

Parameter Name	Date Local	State Name	Barometric pressure	Dew Point	Lead (TSP) LC	Lead (TSP) STP	Lead PM10 LC FRM/FEM	Nitric oxide (NO)	Ter
0	1980-01-01	Alabama	NaN	NaN	NaN	0.050	NaN	54.458333	
1	1980-01-01	Alaska	NaN	NaN	NaN	NaN	NaN	1.027536	-
2	1980-01-01	Arizona	NaN	NaN	NaN	NaN	NaN	59.166667	
3	1980-01-01	Arkansas	NaN	NaN	NaN	NaN	NaN	0.000000	
4	1980-01-01	California	NaN	NaN	NaN	0.575	NaN	75.636977	

In [100…
```
# Combines Multi index rows
daily_combined_parameter_pivot.columns = daily_combined_parameter_pivot.colu
```

In [101…
```
daily_combined_parameter_pivot['Date Local'] = pd.to_datetime(daily_combinec
```

Lots of values are missing for the different states. It's possible readings started at a later date, but if there are gaps in data this needs to be reviewed.

In [102…
```
aqi_combined_pivot = pd.merge(aqi_state, daily_combined_parameter_pivot, how
aqi_combined_pivot.head()
```

Out[102…

	State Name	Date	AQI	Date Local	AM__Barometric pressure	AM__Dew Point	AM__Lead (TSP) LC	AM__L (TSP)
0	Alabama	1980-01-01	57.333333	1980-01-01	NaN	NaN	NaN	
1	Alabama	1980-01-02	71.333333	1980-01-02	NaN	NaN	NaN	
2	Alabama	1980-01-03	88.333333	1980-01-03	NaN	NaN	NaN	
3	Alabama	1980-01-04	35.666667	1980-01-04	NaN	NaN	NaN	
4	Alabama	1980-01-05	24.000000	1980-01-05	NaN	NaN	NaN	

This is missing sentiment, method and Lat/Lon contribution. Use dim Location table and group by state. This is a dummy numeric value at this point to see preliminary.

In [147…
```
dimLocation = pd.read_csv(path+dp+'dimLocation.csv', low_memory = False)
```

In [148…
```
dimLocState = dimLocation.groupby('State Name')[['Latitude', 'Longitude']].a
dimLocState.head()
```

Out[148…

	State Name	Latitude	Longitude
0	Alabama	32.437458	-86.472891
1	Alaska	55.324675	-160.508331
2	Arizona	36.940833	-109.108056
3	Arkansas	34.499264	-91.574296
4	California	37.876870	-122.266913

In [149…
```
preliminary_eda_merged = pd.merge(aqi_combined_pivot, dimLocState, how = 'le
```

In [150…
```
preliminary_eda_merged['Month'] = preliminary_eda_merged['Date'].dt.month
preliminary_eda_merged['Year'] = preliminary_eda_merged['Date'].dt.year
```

preliminary_eda_merged to be used in EDA from here on. Certain assumptions made at this point will need to be iterated over.

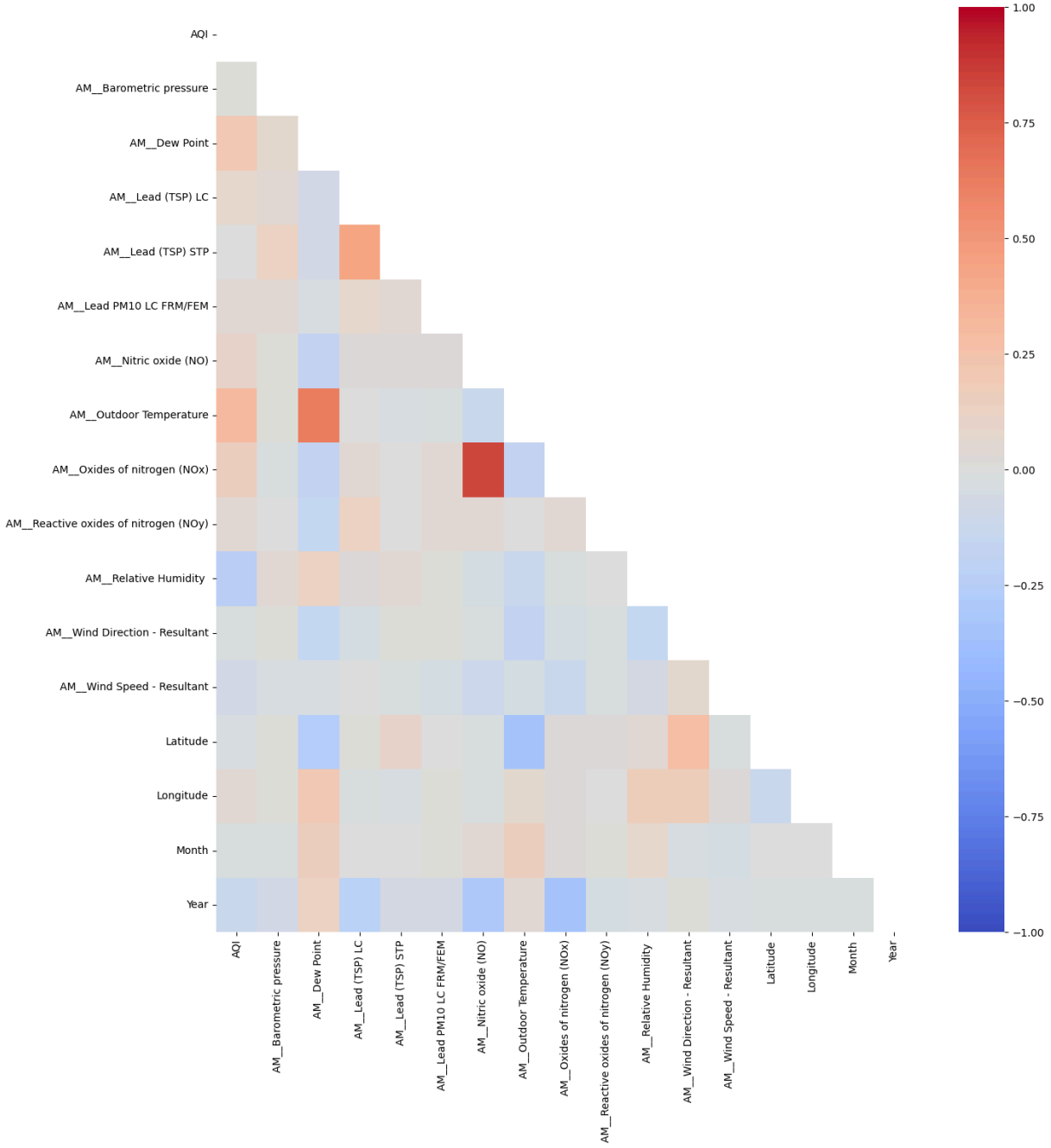
EDA Correlation

In [151…
```
corr_monthly_combined = preliminary_eda_merged.corr(numeric_only = True)
```

In [152…

```
mask = np.triu(corr_monthly_combined) #triu = upper triangle

#let's plot our correlation heatmap
plt.figure(figsize=(15,16))

#annot=True means that the numbers will be displayed inside the boxes themse
sns.heatmap(corr_monthly_combined.round(2), mask=mask,annot=True, cmap='cool
plt.show()
```

```
/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/matrix.py:2
60: FutureWarning:

Format strings passed to MaskedConstant are ignored, but in future may error
or produce different behavior
```



Correlations:

- Mean of Oxides of Nitrogen and Nitric Oxide
- Mean of Lead (TSP) LC and LEAD (TSP) STP
- Mean of Outdoor Temperature and Dew Point
- The highest relationship with AQI is Outdoor temperature

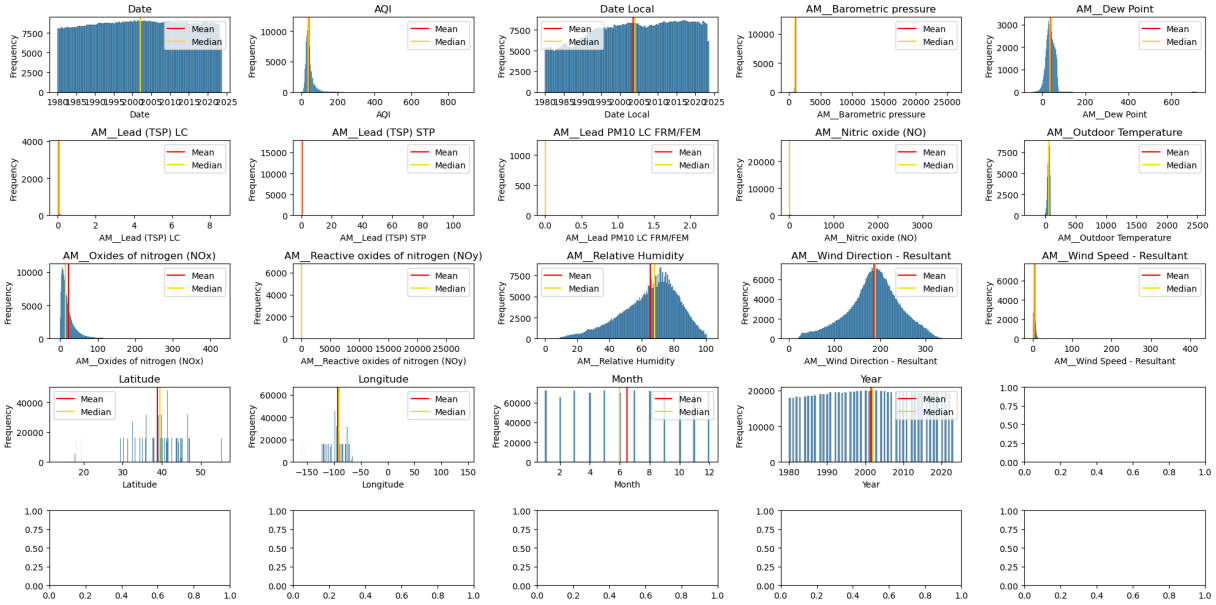
In [153…

```
# Check out the distributions of numeric columns.

plt.subplots(5,5, figsize=(20,10))

# Creating the for loop
for index, col in enumerate(preliminary_eda_merged.select_dtypes(exclude = '

    # Creating a hist for every iteration
    plt.subplot(5,5, index)
    sns.histplot(preliminary_eda_merged[col])

    # Set the titles and labels
    plt.title(col)
    plt.xlabel(col)
    plt.ylabel('Frequency')

    # Adding median and mean lines
    plt.axvline(preliminary_eda_merged[col].mean(), c='red', label='Mean')
    plt.axvline(preliminary_eda_merged[col].median(), c='gold', label='Media
    plt.legend()

plt.tight_layout()

plt.show()
```

```
/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:
```

```
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.

/Users/brookehall/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.p
y:1119: FutureWarning:

use_inf_as_na option is deprecated and will be removed in a future version.
```

```
Convert inf values to NaN before operating instead.
```



Most columns look fairly normally distributed, but there are some high skews (mostly right skews - high data points). However, this could be the presence of outliers.

SAVE PIVOT

In [157…

```
# Fact Files
dp2 = path+'capstoneWorking/data/workingData/'

preliminary_eda_merged.to_csv(dp2 + 'AQIpreliminaryMerged.csv')
```

Summary

What was Completed:

- Pulled in data from 1980 - 2023 and reviewed.
- AQI and combined datasets reviewed for colinearity by day and state. But does not include Method level or consideration of sentiment.
- Correlation between variables has been reviewed
- Preliminary analysis of trends with time and by state

Findings

- As a broad review - there appears to be little correlation/multicolinearity between AQI and any of the available numeric data in the final table (does not include HAPs and VOCs). With the exception of some colinearity between outdoor temperature and AQI.

Questions/Next Steps

- Unsure about impact of Methods on readings/changes with time
- Look into summing Observations and looking into them that way
- Does normalization need to occur to reduce significance/impact of method used or state
- Join between AQI and concentrations
- See if certain groupings can happen to include HAPs and VOCs.
- See if normalization needs to occur for method data
- Closer inspection of missing data - does a smaller date range need to be used?
- Dummy variables of Sentiments for AQI or count of occurences or overall...etc. Maybe a weighting and score or a count of occurence.
- Group Some variables - i.e. lead, nitric oxides, HAPs, VOCs. Normalizing data so impact is included.
- Dealing with nulls for correlation significance

Model

- Time series model for Target variable = AQI.
- From this, log model of good/bad states