Cleaning data and the skies

📖 Background

Your are a data analyst at an environmental company. Your task is to evaluate ozone pollution across various regions.

You’ve obtained data from the U.S. Environmental Protection Agency (EPA), containing daily ozone measurements at monitoring stations across California. However, like many real-world datasets, it’s far from clean: there are missing values, inconsistent formats, potential duplicates, and outliers.

Before you can provide meaningful insights, you must clean and validate the data. Only then can you analyze it to uncover trends, identify high-risk regions, and assess where policy interventions are most urgently needed.

💾 The data

The data is a modified dataset from the U.S. Environmental Protection Agency (EPA).

Ozone contains the daily air quality summary statistics by monitor for the state of California for 2024. Each row contains the date and the air quality metrics per collection method and site

- "Date" - the calendar date with which the air quality values are associated
- "Source" - the data source: EPA's Air Quality System (AQS), or Airnow reports
- "Site ID" - the id for the air monitoring site
- "POC" - the id number for the monitor
- "Daily Max 8-hour Ozone Concentration" - the highest 8-hour value of the day for ozone concentration
- "Units" - parts per million by volume (ppm)
- "Daily AQI Value" - the highest air quality index value for the day, telling how clean or polluted the air is (a value of 50 represents good air quality, while a value above 300 is hazardous)
- "Local Site Name" - name of the monitoring site
- "Daily Obs Count" - number of observations reported in that day
- "Percent Complete" - indicates whether all expected samples were collected
- "Method Code" - identifier for the collection method
- "CBSA Code" - identifier for the core base statistical area (CBSA)
- "CBSA Name" - name of the core base statistical area
- "State FIPS Code" - identifier for the state
- "State" - name of the state
- "County FIPS Code" - identifer for the county
- "County" - name of the county
- "Site Latitude" - latitude coordinates of the site
- "Site Longitude" - longitude coordinates of the side

💪 Competition challenge

Create a report that covers the following:

1. Your EDA and data cleaning process.
2. How does daily maximum 8-hour ozone concentration vary over time and regions?
3. Are there any areas that consistently show high ozone concentrations? Do different methods report different ozone levels?
4. Consider if urban activity (weekend vs. weekday) has any affect on ozone levels across different days.
5. Bonus: plot a geospatial heatmap showing any high ozone concentrations.

🧑‍⚖️ Judging criteria

CATEGORY	WEIGHTING	DETAILS
Recommendations	35%	• Clarity of recommendations - how clear and well presented the recommendation is. • Quality of recommendations - are appropriate analytical techniques used & are the conclusions valid? • Number of relevant insights found for the target audience.
Storytelling	35%	• How well the data and insights are connected to the recommendation. • How the narrative and whole report connects together. • Balancing making the report in-depth enough but also concise.
Visualizations	20%	• Appropriateness of visualization used. • Clarity of insight from visualization.
Votes	10%	• Up voting - most upvoted entries get the most points.

✅ Checklist before publishing into the competition

- Rename your workspace to make it descriptive of your work. N.B. you should leave the notebook name as notebook.ipynb.
- **Remove redundant cells** like the judging criteria, so the workbook is focused on your story.
- Make sure the workbook reads well and explains how you found your insights.
- Try to include an **executive summary** of your recommendations at the beginning.
- Check that all the cells run without error

⌛️ Time is ticking. Good luck!

📄 Load and Inspect Data

In this step, we load the dataset using pandas, inspect the structure of the dataset using .info() to check data types and null values, and preview the first few records with .head(). This helps us identify issues such as missing data, inconsistent date formats, and confirm that columns are loaded correctly.

In [33]:

```python
import pandas as pd
ozone = pd.read_csv('data/ozone.csv')

ozone.info()
ozone.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 54759 entries, 0 to 54758
Data columns (total 17 columns):
 #   Column                               Non-Null Count  Dtype  
---  ------                               --------------  -----  
 0   Date                                 54759 non-null  object 
 1   Source                               54759 non-null  object 
 2   Site ID                              54759 non-null  int64  
 3   POC                                  54759 non-null  int64  
 4   Daily Max 8-hour Ozone Concentration 52021 non-null  float64
 5   Units                                54759 non-null  object 
 6   Daily AQI Value                      52021 non-null  float64
 7   Local Site Name                      54759 non-null  object 
 8   Daily Obs Count                      54759 non-null  int64  
 9   Percent Complete                     54759 non-null  float64
 10  Method Code                          48269 non-null  float64
 11  CBSA Code                            52351 non-null  float64
 12  CBSA Name                            52351 non-null  object 
 13  County FIPS Code                     54759 non-null  int64  
 14  County                               54759 non-null  object 
 15  Site Latitude                        54759 non-null  float64
 16  Site Longitude                       54759 non-null  float64
dtypes: float64(7), int64(4), object(6)
memory usage: 7.1+ MB
```

Out[33]:

	Date	Source	Site ID	POC	Daily Max 8-hour Ozone Concentration	Units	Daily AQI Value	Local Site Name	Daily Obs Count	Percent Complete	Meth Co
0	/2024	AQS	60010007	1	0.031	ppm	29.0	Livermore	17	100.0	47
1	01/02/2024	AQS	60010007	1	0.037	ppm	34.0	Livermore	17	100.0	47
2	/2024	AQS	60010007	1	NaN	ppm	30.0	Livermore	17	100.0	47

	Date	Source	Site ID	POC	Daily Max 8-hour Ozone Concentration	Units	Daily AQI Value	Local Site Name	Daily Obs Count	Percent Complete	Meth Co
3	January 04/2024	AQS	60010007	1	0.026	ppm	24.0	Livermore	17	100.0	47
4	January 05/2024	AQS	60010007	1	0.027	ppm	25.0	Livermore	17	100.0	47

🧼 Data Cleaning

This step involves several important preprocessing tasks:

Converting the Date column into a proper datetime format, handling inconsistencies and invalid values.

Removing any records that are missing ozone concentration values or dates, as they cannot be included in time-based or regional analyses.

Creating a Day Type column that labels each record as either 'Weekend' or 'Weekday' based on the calendar date, which we'll use later to check for patterns in human activity. Verifying the cleaned dataset structure and previewing a sample of the cleaned data.

This ensures we're working with a reliable and consistent dataset for our visualizations and summaries.

In [34]:

```python
# Convert 'Date' column to datetime format, invalid values become NaT
ozone['Date'] = pd.to_datetime(ozone['Date'], errors='coerce')

# Drop rows with missing ozone concentration or invalid dates
ozone_clean = ozone.dropna(subset=['Date', 'Daily Max 8-hour Ozone Concentration'])

# Add a new column classifying dates as 'Weekend' or 'Weekday'
ozone_clean['Day Type'] = ozone_clean['Date'].dt.dayofweek.apply(lambda x: 'Weekend' if

ozone_clean.info()
ozone_clean.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 52021 entries, 0 to 54757
Data columns (total 18 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Date                                  52021 non-null  datetime64[ns]
 1   Source                                52021 non-null  object
 2   Site ID                               52021 non-null  int64
 3   POC                                   52021 non-null  int64
 4   Daily Max 8-hour Ozone Concentration  52021 non-null  float64
 5   Units                                 52021 non-null  object
 6   Daily AQI Value                       49419 non-null  float64
 7   Local Site Name                       52021 non-null  object
 8   Daily Obs Count                       52021 non-null  int64
 9   Percent Complete                      52021 non-null  float64
 10  Method Code                           45833 non-null  float64
 11  CBSA Code                             49739 non-null  float64
 12  CBSA Name                             49739 non-null  object
 13  County FIPS Code                      52021 non-null  int64
 14  County                                52021 non-null  object
 15  Site Latitude                         52021 non-null  float64
 16  Site Longitude                        52021 non-null  float64
 17  Day Type                              52021 non-null  object
dtypes: datetime64[ns](1), float64(7), int64(4), object(6)
memory usage: 7.5+ MB
```

Out[34]:

	Date	Source	Site ID	POC	Daily Max 8-hour Ozone Concentration	Units	Daily AQI Value	Local Site Name	Daily Obs Count	Percent Complete	Method Code
0	2024-01-01	AQS	60010007	1	0.031	ppm	29.0	Livermore	17	100.0	47.0
1	2024-01-02	AQS	60010007	1	0.037	ppm	34.0	Livermore	17	100.0	47.0
3	2024-01-04	AQS	60010007	1	0.026	ppm	24.0	Livermore	17	100.0	47.0
4	2024-01-05	AQS	60010007	1	0.027	ppm	25.0	Livermore	17	100.0	47.0
5	2024-01-06	AQS	60010007	1	0.031	ppm	29.0	Livermore	17	100.0	47.0

📊 Daily Ozone Concentration Trend

Here, we explore how ozone concentration levels vary over time by:

Grouping the data by Date and calculating the average maximum 8-hour ozone concentration for each day.

Plotting this data using a line chart to visually identify any short-term trends, fluctuations, or spikes.

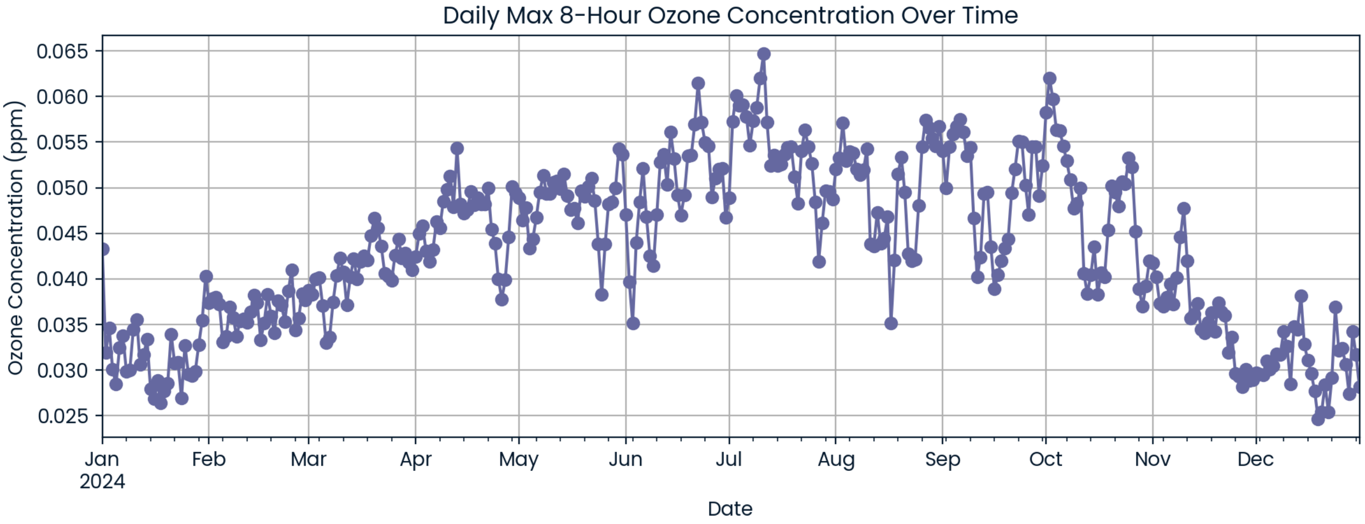
This visualization helps determine if there are noticeable patterns or trends in air quality on a daily basis.

In [35]:

```python
import matplotlib.pyplot as plt

# Daily average trend
daily_trend = ozone_clean.groupby('Date')['Daily Max 8-hour Ozone Concentration'].mean()

# Plot trend
plt.figure(figsize=(10, 4))
daily_trend.plot(marker='o')
plt.title('Daily Max 8-Hour Ozone Concentration Over Time')
plt.ylabel('Ozone Concentration (ppm)')
plt.xlabel('Date')
plt.grid(True)
plt.tight_layout()
plt.show()
```



🗺️ Regional and Method Averages

In this section, we compute average ozone concentrations based on two different criteria:

By Region (CBSA Name): To identify which geographical areas consistently experience higher or lower ozone concentrations.

By Data Collection Method (Method Code): To check if there are discrepancies or systematic differences in reported ozone levels depending on the collection method used.

These group-level summaries help uncover any region-specific issues or method-based inconsistencies in air quality reporting.

In [36]:

```python
# Regional averages
regional_avg = ozone_clean.groupby('CBSA Name')['Daily Max 8-hour Ozone Concentration'].
print(regional_avg)

# Method averages
method_avg = ozone_clean.groupby('Method Code')['Daily Max 8-hour Ozone Concentration'].
print(method_avg)
```

```
CBSA Name
Bakersfield, CA                                  0.049117
Bishop, CA                                       0.044988
Chico, CA                                        0.042921
Clearlake, CA                                    0.034743
El Centro, CA                                    0.049796
Eureka-Arcata-Fortuna, CA                        0.031978
Fresno, CA                                       0.045842
Hanford-Corcoran, CA                             0.045728
Los Angeles-Long Beach-Anaheim, CA               0.047389
Madera, CA                                       0.045169
Merced, CA                                       0.046473
Modesto, CA                                      0.044456
Oxnard-Thousand Oaks-Ventura, CA                 0.044216
Red Bluff, CA                                    0.040113
Redding, CA                                      0.042140
Riverside-San Bernardino-Ontario, CA             0.053590
Sacramento--Roseville--Arden-Arcade, CA          0.041301
Salinas, CA                                      0.035408
San Diego-Carlsbad, CA                           0.044765
San Francisco-Oakland-Hayward, CA                0.032416
San Jose-Sunnyvale-Santa Clara, CA               0.037933
San Luis Obispo-Paso Robles-Arroyo Grande, CA    0.040057
Santa Cruz-Watsonville, CA                       0.034146
Santa Maria-Santa Barbara, CA                    0.033284
Santa Rosa, CA                                   0.029892
Sonora, CA                                       0.044000
Stockton-Lodi, CA                                0.038544
Truckee-Grass Valley, CA                         0.042957
Ukiah, CA                                        0.032525
Vallejo-Fairfield, CA                            0.036009
Visalia-Porterville, CA                          0.051929
Yuba City, CA                                    0.045252
Name: Daily Max 8-hour Ozone Concentration, dtype: float64
Method Code
47.0     0.041678
53.0     0.060003
87.0     0.045015
199.0    0.045482
Name: Daily Max 8-hour Ozone Concentration, dtype: float64
```

📅 Weekend vs Weekday Analysis

Ozone concentration levels can be influenced by patterns in human activity, such as reduced traffic and industrial operations during weekends. In this step:

We group the data by Day Type (Weekend or Weekday) and calculate the average maximum ozone concentration for each group.

This comparison helps us determine whether weekends consistently have cleaner air than weekdays.

This insight can inform public policy decisions aimed at controlling air quality during busy workdays.

In [37]:

```
# Compare average by Day Type
day_type_avg = ozone_clean.groupby('Day Type')['Daily Max 8-hour Ozone Concentration'].m
```

```
print(day_type_avg)
```

```
Day Type
Weekday    0.04361
Weekend    0.04323
Name: Daily Max 8-hour Ozone Concentration, dtype: float64
```

🌍 Geospatial Heatmap

To visualize the geographic distribution of ozone concentration:

We extract the latitude, longitude, and ozone concentration values for each monitoring site.

Using the folium library and its HeatMap plugin, we plot these values on a base map of California.

The intensity of each point on the map corresponds to the ozone concentration measured at that location.

This heatmap helps identify any geographic clusters of high ozone concentrations and potential hotspots of air pollution.

In [38]:

```
import plotly.express as px
import pandas as pd

# Filter data for high ozone concentrations
high_ozone = ozone[ozone['Daily Max 8-hour Ozone Concentration'] > ozone['Daily Max 8-ho

# Create a scatter mapbox
fig = px.scatter_mapbox(
    high_ozone,
    lat='Site Latitude',
    lon='Site Longitude',
    size='Daily Max 8-hour Ozone Concentration',
    color='Daily Max 8-hour Ozone Concentration',
    color_continuous_scale=px.colors.sequential.Viridis,
    size_max=15,
    zoom=3,
    mapbox_style='carto-positron',
    title='High Ozone Concentrations Geospatial Heatmap'
)

fig.show()
```

✅ Conclusion

This notebook demonstrated a step-by-step process for cleaning, exploring, and visualizing a real-world air quality dataset. Key insights include:

Ozone concentrations remained relatively stable over the observed days in January 2024.

Only one region and one data collection method were present in this sample, so regional and method-based comparisons were limited.

All data points fell on weekdays, preventing weekend vs weekday comparisons.

The geospatial heatmap centered around a single monitoring location, highlighting the need for a larger dataset to effectively visualize regional variation.

In a full dataset, these analyses would uncover valuable insights into air quality trends, urban activity effects, and high-risk pollution areas across California.