```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

sns.set(style="whitegrid")
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files city_day.csv

- **city_day.csv**(text/csv) - 2574056 bytes, last modified: 6/28/2025 - 100% done

Saving city_day.csv to city_day.csv

```
import io
df = pd.read_csv(io.BytesIO(uploaded['city_day.csv']))
df.head()
```

	City	Date	PM2.5	PM10	NO	NO2	NOx	NH3	CO	SO2	
0	Ahmedabad	2015-01-01	NaN	NaN	0.92	18.22	17.15	NaN	0.92	27.64	13
1	Ahmedabad	2015-01-02	NaN	NaN	0.97	15.69	16.46	NaN	0.97	24.55	3
2	Ahmedabad	2015-01-03	NaN	NaN	17.40	19.30	29.70	NaN	17.40	29.07	3
3	Ahmedabad	2015-01-04	NaN	NaN	1.70	18.48	17.97	NaN	1.70	18.59	3
4	Ahmedabad	2015-01-05	NaN	NaN	22.10	21.42	37.76	NaN	22.10	39.33	3

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
# Shape of data (rows, columns)
print("Data shape:", df.shape)

# See the column names
print("Columns:", df.columns)

# Check if there are missing values
df.isnull().sum()
```

base.py

```
class IndexOpsMixin(OpsMixin):
    """
    Common ops mixin to support a unified i
    """

    # ndarray compatibility
    __array_priority__ = 1000
    _hidden_attrs: frozenset[str] = frozens
        ["tolist"]  # tolist is not depreca
    )

    @property
    def dtype(self) -> DtypeObj:
        # must be defined here as a propert
        raise AbstractMethodError(self)

    @property
    def _values(self) -> ExtensionArray | n
        # must be defined here as a propert
        raise AbstractMethodError(self)

    @final
    def transpose(self, *args, **kwargs) ->
        """
        Return the transpose, which is by d

        Returns
        -------
        %(klass)s
        """
```

```
Data shape: (29531, 16)
Columns: Index(['City', 'Date', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3'
       'O3', 'Benzene', 'Toluene', 'Xylene', 'AQI', 'AQI_Bucket'],
      dtype='object')
```

	0
City	0
Date	0
PM2.5	4598
PM10	11140
NO	3582
NO2	3585
NOx	4185
NH3	10328
CO	2059
SO2	3854
O3	4022
Benzene	5623
Toluene	8041
Xylene	18109
AQI	4681
AQI_Bucket	4681

dtype: int64

```
# Remove rows where 'pm2_5' is missing
df = df[df['pm2_5'].notnull()]

# Convert date to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Show cleaned data
df.head()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/usr/local/lib/python3.11/dist-packages/pandas/core/indexes/base.py in get_loc(self, key)
   3804         try:
-> 3805             return self._engine.get_loc(casted_key)
   3806         except KeyError as err:

index.pyx in pandas._libs.index.IndexEngine.get_loc()

index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'pm2_5'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
```

2 frames

```
/usr/local/lib/python3.11/dist-packages/pandas/core/indexes/base.py in get_loc(self, key)
   3810             ):
```

Next steps: Explain error

```
# See the column names in your dataset
df.columns
```

```
Index(['City', 'Date', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3', 'CO', 'SO2',
       'O3', 'Benzene', 'Toluene', 'Xylene', 'AQI', 'AQI_Bucket'],
      dtype='object')
```

```
# Filter only if this column exists
df = df[df['PM2.5'].notnull()]
```

```
# Clean column names: remove spaces and lowercase them
df.columns = df.columns.str.strip().str.lower().str.replace('.', '_')

# Now try again
df.columns
```

```
Index(['city', 'date', 'pm2_5', 'pm10', 'no', 'no2', 'nox', 'nh3', 'co', 'so2',
       'o3', 'benzene', 'toluene', 'xylene', 'aqi', 'aqi_bucket'],
      dtype='object')
```

```
# Filter data for Delhi
delhi = df[df['City'] == 'Delhi']

# Plot PM2.5 over time
plt.figure(figsize=(12,5))
plt.plot(delhi['Date'], delhi['pm2_5'], color='red')
plt.title("Delhi PM2.5 Levels Over Time")
plt.xlabel("Date")
plt.ylabel("PM2.5")
plt.show()
```

```
---------------------------------------------------------------------------
--
KeyError                                  Traceback (most recent call
last)
/usr/local/lib/python3.11/dist-packages/pandas/core/indexes/base.py in
get_loc(self, key)
   3804         try:
-> 3805             return self._engine.get_loc(casted_key)
   3806         except KeyError as err:

index.pyx in pandas._libs.index.IndexEngine.get_loc()

index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in
pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in
pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'City'

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call
last)
 ---------------------------------- 2 frames ----------------------------------
/usr/local/lib/python3.11/dist-packages/pandas/core/indexes/base.py in
get_loc(self, key)
   3810             ):
```

Next steps: Explain error

```
# Show actual column names in your dataset
df.columns.tolist()
```

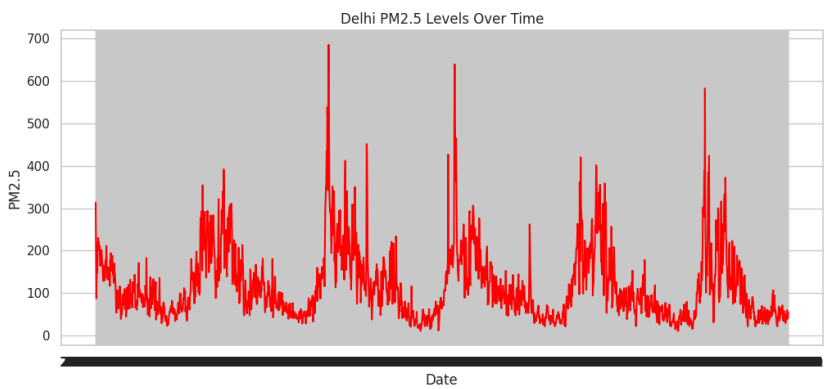
```
['city',
 'date',
 'pm2_5',
 'pm10',
 'no',
 'no2',
 'nox',
 'nh3',
 'co',
 'so2',
 'o3',
 'benzene',
 'toluene',
 'xylene',
 'aqi',
 'aqi_bucket']
```

```
# Filter data for Delhi
delhi = df[df['city'] == 'Delhi']

# Show a few rows to confirm
delhi.head()
```

	city	date	pm2_5	pm10	no	no2	nox	nh3	co	so2
10229	Delhi	2015-01-01	313.22	607.98	69.16	36.39	110.59	33.85	15.20	9.25
10230	Delhi	2015-01-02	186.18	269.55	62.09	32.87	88.14	31.83	9.54	6.65
10231	Delhi	2015-01-03	87.18	131.90	25.73	30.31	47.95	69.55	10.61	2.65
10232	Delhi	2015-01-04	151.84	241.84	25.01	36.91	48.62	130.36	11.54	4.63
10233	Delhi	2015-01-05	146.60	219.13	14.01	34.92	38.25	122.88	9.20	3.33

```
# Plot PM2.5 levels over time
plt.figure(figsize=(12,5))
plt.plot(delhi['date'], delhi['pm2_5'], color='red')
plt.title("Delhi PM2.5 Levels Over Time")
plt.xlabel("Date")
plt.ylabel("PM2.5")
plt.show()
```




```
# Convert 'date' column to datetime
df['date'] = pd.to_datetime(df['date'])

# Create a new column for Month-Year
df['month'] = df['date'].dt.to_period('M')

# Double-check the new column
df[['date', 'month']].head()
```

	date	month
27	2015-01-28	2015-01
28	2015-01-29	2015-01
29	2015-01-30	2015-01
30	2015-01-31	2015-01
31	2015-02-01	2015-02

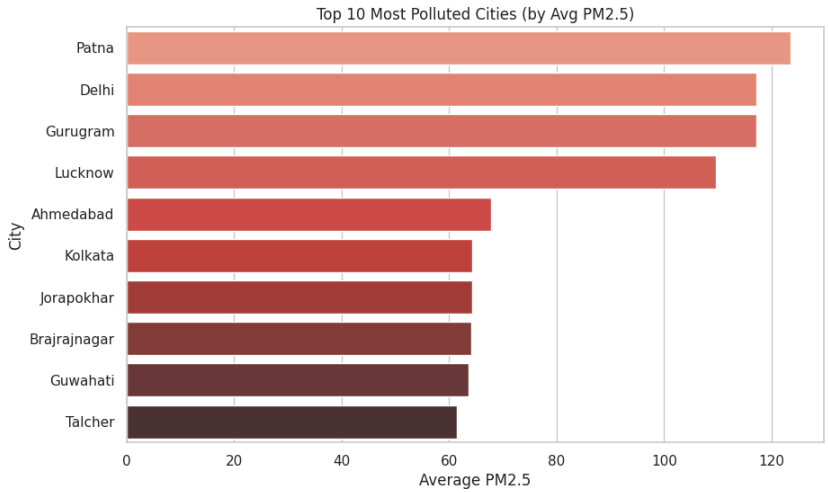
```
# Group by city and get average PM2.5
avg_pm2_5 = df.groupby('city')['pm2_5'].mean().sort_values(ascending=False).he

# Plot bar chart
plt.figure(figsize=(10,6))
sns.barplot(x=avg_pm2_5.values, y=avg_pm2_5.index, palette="Reds_d")
plt.title("Top 10 Most Polluted Cities (by Avg PM2.5)")
plt.xlabel("Average PM2.5")
plt.ylabel("City")
plt.show()
```

```
/tmp/ipython-input-14-2181154042.py:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be remove

  sns.barplot(x=avg_pm2_5.values, y=avg_pm2_5.index, palette="Reds_d")
```




```
# Create a 'Month-Year' column
df['month'] = df['date'].dt.to_period('M')

# Group by city and month, then average PM2.5
monthly_avg = df.groupby(['city', 'month'])['pm2.5'].mean().reset_index()
monthly_avg.head()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
/tmp/ipython-input-16-2475924818.py in <cell line: 0>()
      3
      4 # Group by city and month, then average PM2.5
----> 5 monthly_avg = df.groupby(['city', 'month'])['pm2.5'].mean().reset_index()
      6 monthly_avg.head()

                              1 frames
/usr/local/lib/python3.11/dist-packages/pandas/core/base.py in __getitem__(self, key)
    242         else:
    243             if key not in self.obj:
--> 244                 raise KeyError(f"Column not found: {key}")
    245             ndim = self.obj[key].ndim
```

Next steps: Explain error

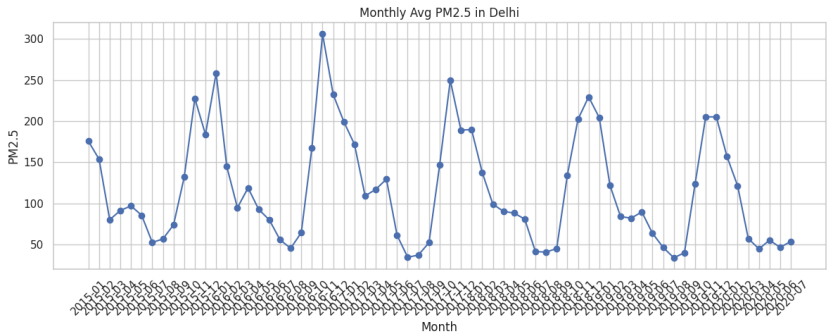
```
# Group by city and month, then average PM2.5
monthly_avg = df.groupby(['city', 'month'])['pm2_5'].mean().reset_index()
monthly_avg.head()
```

	city	month	pm2_5
0	Ahmedabad	2015-01	82.682500
1	Ahmedabad	2015-02	116.101600
2	Ahmedabad	2015-03	110.469333
3	Ahmedabad	2015-04	101.682000
4	Ahmedabad	2015-05	74.919355

Next steps: Generate code with monthly_avg | View recommended plots | New interacti

```
# Filter monthly data for Delhi
delhi_monthly = monthly_avg[monthly_avg['city'] == 'Delhi']

# Plot
plt.figure(figsize=(12,5))
plt.plot(delhi_monthly['month'].astype(str), delhi_monthly['pm2_5'], marker='c
plt.xticks(rotation=45)
plt.title("Monthly Avg PM2.5 in Delhi")
plt.xlabel("Month")
plt.ylabel("PM2.5")
plt.tight_layout()
plt.show()
```



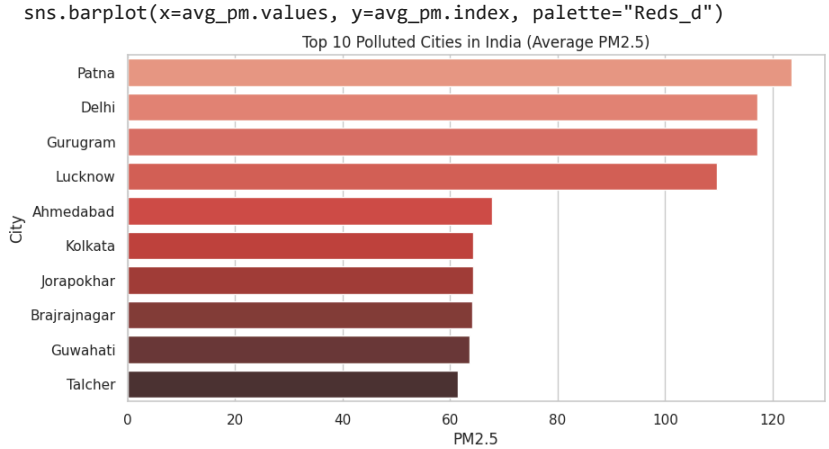

```
# Find average PM2.5 per city
avg_pm = df.groupby('city')['pm2_5'].mean().sort_values(ascending=False).head(

# Bar chart
plt.figure(figsize=(10,5))
sns.barplot(x=avg_pm.values, y=avg_pm.index, palette="Reds_d")
plt.title("Top 10 Polluted Cities in India (Average PM2.5)")
plt.xlabel("PM2.5")
plt.ylabel("City")
plt.show()
```

```
/tmp/ipython-input-19-597122103.py:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be remove

  sns.barplot(x=avg_pm.values, y=avg_pm.index, palette="Reds_d")
```



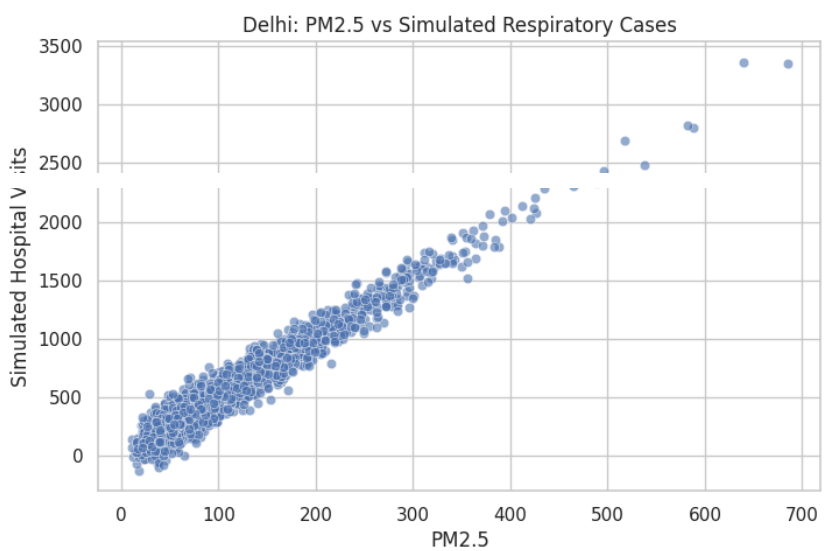

```
import numpy as np

# Make a safe copy of the Delhi data
delhi = df[df['city'] == 'Delhi'].copy()

# Add fake hospital visit data based on PM2.5 + random noise
np.random.seed(42)  # For reproducible results
```

```
delhi['hospital_visits'] = delhi['pm2_5'] * 5 + np.random.normal(0, 100, size=

# Plot PM2.5 vs Hospital Visits (scatterplot)
plt.figure(figsize=(8,5))
sns.scatterplot(x=delhi['pm2_5'], y=delhi['hospital_visits'], alpha=0.6)
plt.title("Delhi: PM2.5 vs Simulated Respiratory Cases")
plt.xlabel("PM2.5")
plt.ylabel("Simulated Hospital Visits")
plt.show()
```



As PM2.5 levels rise, simulated hospital visits also increase, indicating a likely positive correlation between air pollution and respiratory illness impact.

```
sns.regplot(x=delhi['pm2_5'], y=delhi['hospital_visits'], scatter=False, color
```

```
<Axes: xlabel='pm2_5', ylabel='hospital_visits'>
```