```
print("Hi")
```

```
Hi
```

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

```
df=pd.read_csv("/content/owid_co2_clean_v1.csv")
```

```
df.head()
```

	iso_code	country	year	co2	co2_per_capita	gdp	population	coal_co2	oil_co2	gas_co2
0	AFG	Afghanistan	1949	0.015	0.002	NaN	7624058.0	0.015	NaN	NaN
1	AFG	Afghanistan	1950	0.084	0.011	9.421400e+09	7752117.0	0.021	0.063	NaN
2	AFG	Afghanistan	1951	0.092	0.012	9.692280e+09	7840151.0	0.026	0.066	NaN
3	AFG	Afghanistan	1952	0.092	0.012	1.001732e+10	7935996.0	0.032	0.060	NaN
4	AFG	Afghanistan	1953	0.106	0.013	1.063052e+10	8039684.0	0.038	0.068	NaN

```
df.columns
```

```
Index(['iso_code', 'country', 'year', 'co2', 'co2_per_capita', 'gdp',
       'population', 'coal_co2', 'oil_co2', 'gas_co2'],
      dtype='object')
```

```
print(df.dtypes)
```

```
iso_code           object
country            object
year                int64
co2               float64
co2_per_capita    float64
gdp               float64
population        float64
coal_co2          float64
oil_co2           float64
gas_co2           float64
dtype: object
```

```
df.shape
```

```
(23949, 10)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23949 entries, 0 to 23948
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   iso_code        21112 non-null  object
 1   country         23949 non-null  object
 2   year            23949 non-null  int64
 3   co2             23949 non-null  float64
 4   co2_per_capita  23356 non-null  float64
 5   gdp             13426 non-null  float64
 6   population      22101 non-null  float64
 7   coal_co2        17188 non-null  float64
 8   oil_co2         20539 non-null  float64
 9   gas_co2         8845 non-null   float64
dtypes: float64(7), int64(1), object(2)
memory usage: 1.8+ MB
```

```
df.describe()
```

	year	co2	co2_per_capita	gdp	population	coal_co2	oil_co2	gas_co2
count	23949.000000	23949.000000	23356.000000	1.342600e+04	2.210100e+04	17188.000000	20539.000000	8845.000000
mean	1954.800869	267.861942	4.162061	2.897320e+11	7.209688e+07	175.358171	106.254381	108.750774
std	52.398931	1521.680894	14.897772	2.189050e+12	3.852525e+08	786.106838	602.683622	441.064563
min	1750.000000	0.000000	0.000000	5.543200e+07	1.490000e+03	0.000000	0.000000	0.000000
25%	1928.000000	0.528000	0.250000	9.859887e+09	1.349373e+06	0.322000	0.311000	0.385000
50%	1968.000000	4.857000	1.241000	3.044132e+10	5.111371e+06	3.981000	2.100000	4.199000
75%	1995.000000	42.818000	4.646250	1.286544e+11	1.829461e+07	35.532750	17.369000	30.830000
max	2020.000000	36702.503000	748.639000	1.136302e+14	7.794799e+09	15062.902000	12229.642000	7553.394000

```
df.isnull().sum()
```

	0
iso_code	2837
country	0
year	0
co2	0
co2_per_capita	593
gdp	10523
population	1848
coal_co2	6761
oil_co2	3410
gas_co2	15104

dtype: int64

```
df.duplicated().sum()
```

```
np.int64(0)
```

```
df.drop_duplicates(inplace=True)
```

```
df.dropna(how="all",inplace=True)
```

```
df.columns=df.columns.str.strip().str.lower().str.replace(" ","_")
```

```
df.columns
```

```
Index(['iso_code', 'country', 'year', 'co2', 'co2_per_capita', 'gdp',
       'population', 'coal_co2', 'oil_co2', 'gas_co2'],
      dtype='object')
```

```
numeric_cols=df.select_dtypes(include=["object"]).columns
for col in numeric_cols:
  try:
    df[col]=pd.to_numeric(df[col],errors="ignore")
  except:
    pass
```

```
/tmp/ipython-input-2844254157.py:4: FutureWarning: errors='ignore' is deprecated and will raise in a future version. Use to_numeric
  df[col]=pd.to_numeric(df[col],errors="ignore")
```

```
print("Cleaned data Info ")
print(df.info())
```

```
Cleaned data Info
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23949 entries, 0 to 23948
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   iso_code        21112 non-null  object
```

```
 1   country         23949 non-null  object
 2   year            23949 non-null  int64
 3   co2             23949 non-null  float64
 4   co2_per_capita  23356 non-null  float64
 5   gdp             13426 non-null  float64
 6   population      22101 non-null  float64
 7   coal_co2        17188 non-null  float64
 8   oil_co2         20539 non-null  float64
 9   gas_co2         8845 non-null   float64
dtypes: float64(7), int64(1), object(2)
memory usage: 1.8+ MB
None
```

```
df.isnull().sum()
```

	0
iso_code	2837
country	0
year	0
co2	0
co2_per_capita	593
gdp	10523
population	1848
coal_co2	6761
oil_co2	3410
gas_co2	15104

dtype: int64

```
# Fill numeric columns with median value per country
numeric_cols = ['gdp', 'population', 'co2_per_capita', 'coal_co2', 'oil_co2', 'gas_co2']
for col in numeric_cols:
    df[col] = df.groupby('country')[col].transform(lambda x: x.fillna(x.median()))

# Drop rows where iso_code is missing
df = df.dropna(subset=['iso_code'])

# Check again
print(df.isnull().sum())
```

```
iso_code             0
country              0
year                 0
co2                  0
co2_per_capita      48
gdp               3044
population          48
coal_co2          3775
oil_co2              0
gas_co2           5615
dtype: int64
```

```
# 1. Drop rows with missing iso_code
df = df.dropna(subset=['iso_code'])

# 2. Fill emission-related nulls with 0
df[['co2_per_capita', 'coal_co2', 'oil_co2', 'gas_co2']] = \
df[['co2_per_capita', 'coal_co2', 'oil_co2', 'gas_co2']].fillna(0)

# 3. Fill GDP nulls with median GDP per country
df['gdp'] = df.groupby('country')['gdp'].transform(lambda x: x.fillna(x.median()))

# 4. Fill Population nulls with median per country
df['population'] = df.groupby('country')['population'].transform(lambda x: x.fillna(x.median()))

# Check nulls again
print(df.isnull().sum())
```

```
iso_code             0
country              0
year                 0
co2                  0
co2_per_capita       0
```

```
gdp               3044
population          48
coal_co2             0
oil_co2              0
gas_co2              0
dtype: int64
```

```
# Fill string column
df['iso_code'] = df['iso_code'].fillna('Unknown')

# Fill numeric columns
df['co2_per_capita'] = df['co2_per_capita'].fillna(df['co2_per_capita'].mean())
df['gdp'] = df['gdp'].fillna(df['gdp'].median())
df['population'] = df['population'].fillna(method='ffill')

# Fill CO₂ sources with 0
df['coal_co2'] = df['coal_co2'].fillna(0)
df['oil_co2'] = df['oil_co2'].fillna(0)
df['gas_co2'] = df['gas_co2'].fillna(0)
```

```
/tmp/ipython-input-2567358954.py:7: FutureWarning: Series.fillna with 'method' is deprecated and will raise in a future version. Use
  df['population'] = df['population'].fillna(method='ffill')
```

```
df.isnull().sum()
```

	0
iso_code	0
country	0
year	0
co2	0
co2_per_capita	0
gdp	0
population	0
coal_co2	0
oil_co2	0
gas_co2	0

dtype: int64

```
# Fill missing values with 0 for numerical columns
df['co2_per_capita'] = df['co2_per_capita'].fillna(0)
df['gdp'] = df['gdp'].fillna(0)
df['population'] = df['population'].fillna(0)
df['coal_co2'] = df['coal_co2'].fillna(0)
df['oil_co2'] = df['oil_co2'].fillna(0)
df['gas_co2'] = df['gas_co2'].fillna(0)

# iso_code can be left as NaN if it’s just metadata
```

```
print(df.describe())
print(df.info())
```

```
                year           co2  co2_per_capita           gdp    population  \
count  21112.000000  21112.000000    21112.000000  2.111200e+04  2.111200e+04
mean    1959.451686    158.682005        4.342292  4.341760e+11  5.509382e+07
std       47.862368   1387.880029       15.711641  2.578594e+12  3.504665e+08
min     1750.000000      0.000000        0.000000  5.543200e+07  1.490000e+03
25%     1935.000000      0.462000        0.234000  1.730216e+10  1.251012e+06
50%     1970.000000      3.671500        1.189500  3.897846e+10  4.739270e+06
75%     1996.000000     28.088250        4.742250  1.141529e+11  1.525414e+07
max     2020.000000  36702.503000      748.639000  1.136302e+14  7.794799e+09

            coal_co2       oil_co2       gas_co2
count  21112.000000  21112.000000  21112.000000
mean      74.746437     62.187632     33.386326
std      595.533541    520.258997    246.664897
min        0.000000      0.000000      0.000000
25%        0.004000      0.389750      0.000000
50%        0.443000      2.455000      1.242000
75%        7.409500     14.525500      9.284000
max    15062.902000  12229.642000   7553.394000
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 21112 entries, 0 to 23948
Data columns (total 10 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   iso_code        21112 non-null  object
 1   country         21112 non-null  object
 2   year            21112 non-null  int64
 3   co2             21112 non-null  float64
 4   co2_per_capita  21112 non-null  float64
 5   gdp             21112 non-null  float64
 6   population      21112 non-null  float64
 7   coal_co2        21112 non-null  float64
 8   oil_co2         21112 non-null  float64
 9   gas_co2         21112 non-null  float64
dtypes: float64(7), int64(1), object(2)
memory usage: 1.8+ MB
None
```

```
print("Total countries:", df['country'].nunique())
print("Year range:", df['year'].min(), "to", df['year'].max())
```

```
Total countries: 219
Year range: 1750 to 2020
```

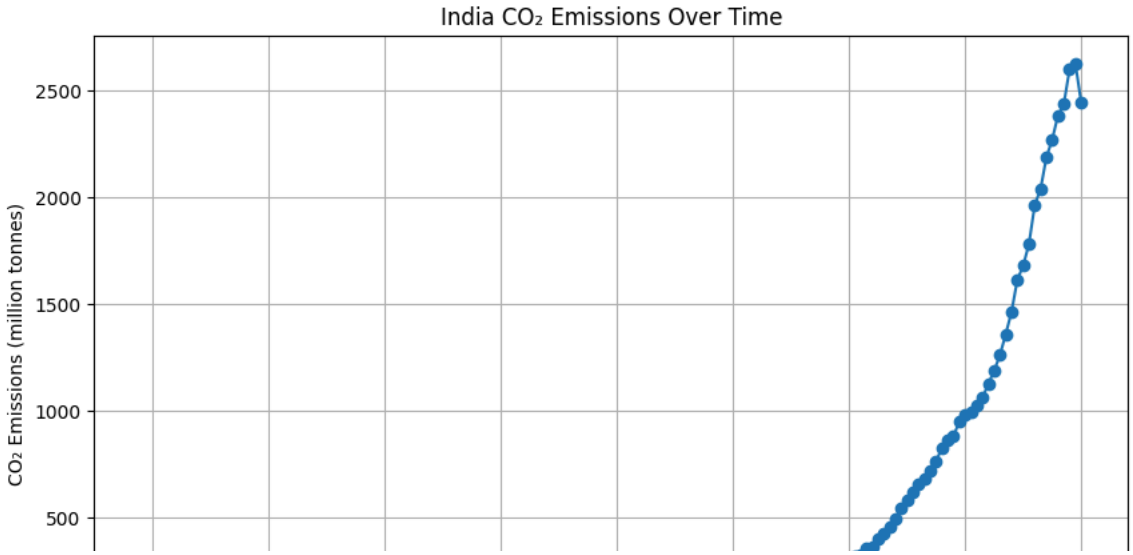
```
latest_year = df['year'].max()
top_emitters = df[df['year'] == latest_year].sort_values(by='co2', ascending=False).head(10)

print(top_emitters[['country', 'co2']])
```

```
              country        co2
23688           World  34807.259
4536            China  10667.887
22841  United States   4712.771
10473           India   2441.792
18003          Russia   1577.136
11466           Japan   1030.775
10787            Iran    745.035
9123          Germany    644.310
18613   Saudi Arabia    625.508
19907    South Korea    597.605
```

```
india = df[df['country'] == "India"]

plt.figure(figsize=(10,6))
plt.plot(india['year'], india['co2'], marker='o')
plt.title("India CO₂ Emissions Over Time")
plt.xlabel("Year")
plt.ylabel("CO₂ Emissions (million tonnes)")
plt.grid(True)
plt.show()
```


India CO₂ Emissions Over Time
CO₂ Emissions (million tonnes)
2500
2000
1500
1000
500