In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.impute import SimpleImputer
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
data = pd.read_csv("DMV Dataset/data.csv", encoding="cp1252")
data
```

Out[2]:

	stn_code	sampling_date	state	location	agency	type	so2	no2	rspm	sp
0	150.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	4.8	17.4	NaN	N
1	151.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	3.1	7.0	NaN	N
2	152.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.2	28.5	NaN	N
3	150.0	March - M031990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.3	14.7	NaN	N
4	151.0	March - M031990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	4.7	7.5	NaN	N
...	...	...	...	...	...	...	...	...	...	
435737	SAMP	24-12-15	West Bengal	ULUBERIA	West Bengal State Pollution Control Board	RIRUO	22.0	50.0	143.0	N
435738	SAMP	29-12-15	West Bengal	ULUBERIA	West Bengal State Pollution Control Board	RIRUO	20.0	46.0	171.0	N
435739	NaN	NaN	andaman-and-nicobar-islands	NaN	NaN	NaN	NaN	NaN	NaN	N
435740	NaN	NaN	Lakshadweep	NaN	NaN	NaN	NaN	NaN	NaN	N
435741	NaN	NaN	Tripura	NaN	NaN	NaN	NaN	NaN	NaN	N

435742 rows × 13 columns

In [3]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 435742 entries, 0 to 435741
Data columns (total 13 columns):
 #   Column                        Non-Null Count   Dtype  
---  ------                        --------------   -----  
 0   stn_code                      291665 non-null  object 
 1   sampling_date                 435739 non-null  object 
 2   state                         435742 non-null  object 
 3   location                      435739 non-null  object 
 4   agency                        286261 non-null  object 
 5   type                          430349 non-null  object 
 6   so2                           401096 non-null  float64
 7   no2                           419509 non-null  float64
 8   rspm                          395520 non-null  float64
 9   spm                           198355 non-null  float64
 10  location_monitoring_station   408251 non-null  object 
 11  pm2_5                         9314 non-null    float64
 12  date                          435735 non-null  object 
dtypes: float64(5), object(8)
memory usage: 43.2+ MB
```

In [4]:
```
data.state = data.state.replace({'Uttaranchal':'Uttarakhand'})
data.state[data.location == "Jamshedpur"] = data.state[data.location == 'Jamshedpur'].replace
```

In [5]:
```
types = {
    "Residential": "R",
    "Residential and others": "RO",
    "Residential, Rural and other Areas": "RRO",
    "Industrial Area": "I",
    "Industrial Areas": "I",
    "Industrial": "I",
    "Sensitive Area": "S",
    "Sensitive Areas": "S",
    "Sensitive": "S",
    np.nan: "RRO"
}

data.type = data.type.replace(types)
data.head()
```

Out[5]:

	stn_code	sampling_date	state	location	agency	type	so2	no2	rspm	spm	location_monit
0	150.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	RRO	4.8	17.4	NaN	NaN	
1	151.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	I	3.1	7.0	NaN	NaN	
2	152.0	February - M021990	Andhra Pradesh	Hyderabad	NaN	RRO	6.2	28.5	NaN	NaN	
3	150.0	March - M031990	Andhra Pradesh	Hyderabad	NaN	RRO	6.3	14.7	NaN	NaN	
4	151.0	March - M031990	Andhra Pradesh	Hyderabad	NaN	I	4.7	7.5	NaN	NaN	

In [6]:
```
VALUE_COLS = ['so2', 'no2', 'rspm', 'spm', 'pm2_5']
```

In [11]:
```
imputer = SimpleImputer(missing_values=np.nan, strategy='mean')
data[VALUE_COLS] = imputer.fit_transform(data[VALUE_COLS])
```

In [12]:
```
data.isnull().sum()
```

Out[12]:

```
stn_code                       144077
sampling_date                       3
state                               0
location                            3
agency                         149481
type                                0
so2                                 0
no2                                 0
rspm                                0
spm                                 0
location_monitoring_station     27491
pm2_5                               0
date                                7
dtype: int64
```

In [13]:

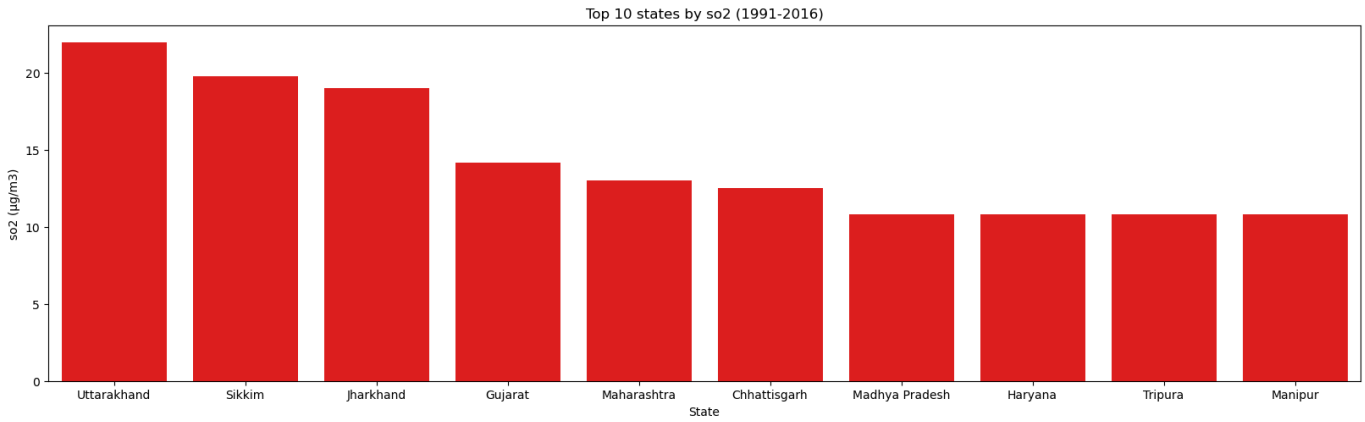
```
data.tail()
```

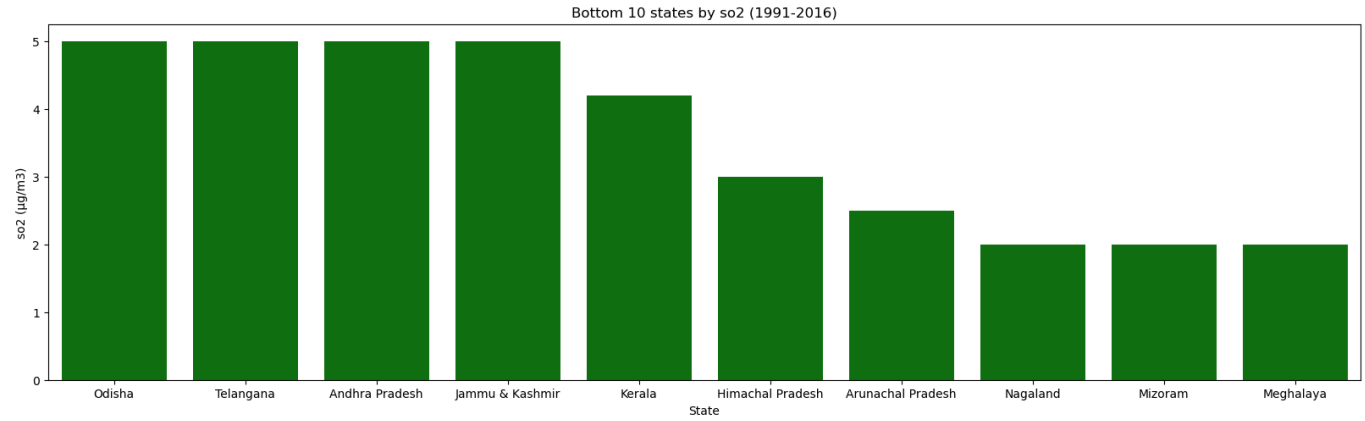
Out[13]:

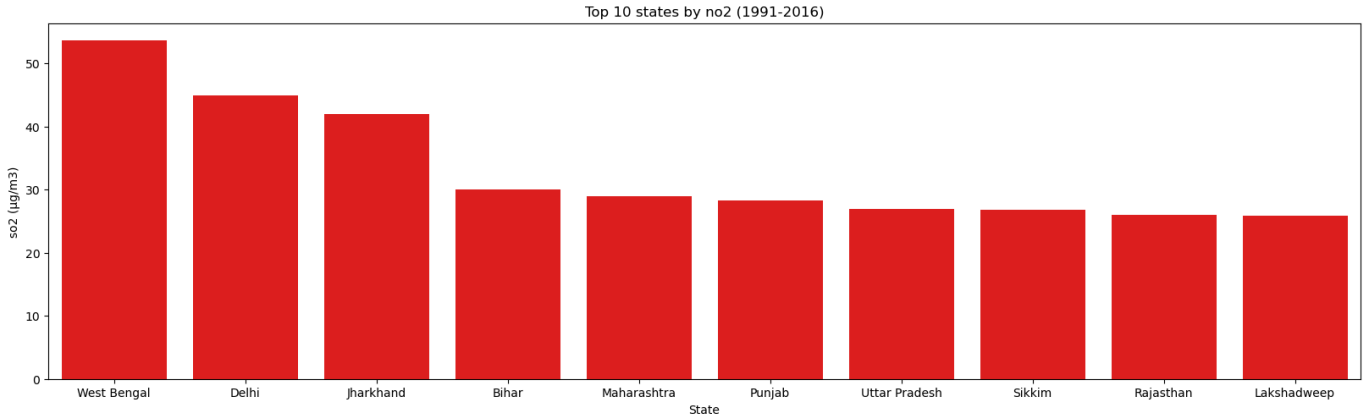
	stn_code	sampling_date	state	location	agency	type	so2	no2	
435737	SAMP	24-12-15	West Bengal	ULUBERIA	West Bengal State Pollution Control Board	RIRUO	22.000000	50.000000	143
435738	SAMP	29-12-15	West Bengal	ULUBERIA	West Bengal State Pollution Control Board	RIRUO	20.000000	46.000000	171
435739	NaN	NaN	andaman-and-nicobar-islands	NaN	NaN	RRO	10.829414	25.809623	108
435740	NaN	NaN	Lakshadweep	NaN	NaN	RRO	10.829414	25.809623	108
435741	NaN	NaN	Tripura	NaN	NaN	RRO	10.829414	25.809623	108

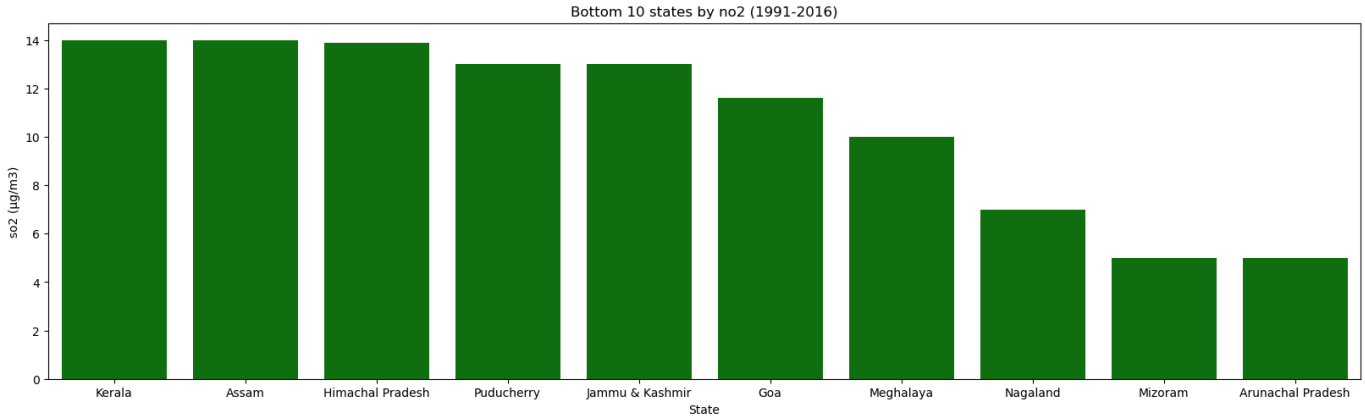
In [14]:

```
def top_and_bottom_10_states(indicator="so2"):
    fig, ax = plt.subplots(2,1, figsize=(20, 12))
    ind = data[[indicator, 'state']].groupby('state', as_index=False).median().sort_values(by
    top10 = sns.barplot(x='state', y=indicator, data=ind[:10], ax=ax[0], color='red')
    top10.set_title("Top 10 states by {} (1991-2016)".format(indicator))
    top10.set_ylabel("so2 (µg/m3)")
    top10.set_xlabel("State")
    bottom10 = sns.barplot(x='state', y=indicator, data=ind[-10:], ax=ax[1], color='green')
    bottom10.set_title("Bottom 10 states by {} (1991-2016)".format(indicator))
    bottom10.set_ylabel("so2 (µg/m3)")
    bottom10.set_xlabel("State")

top_and_bottom_10_states("so2")
top_and_bottom_10_states("no2")
```



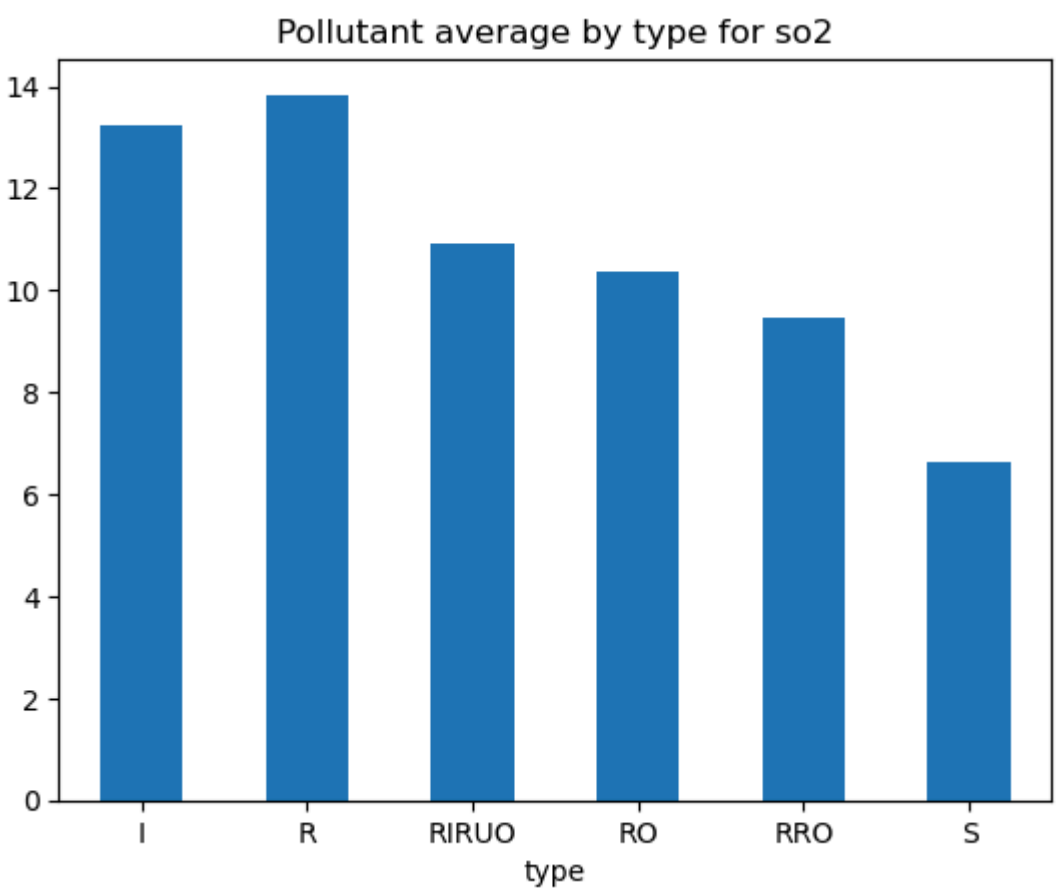




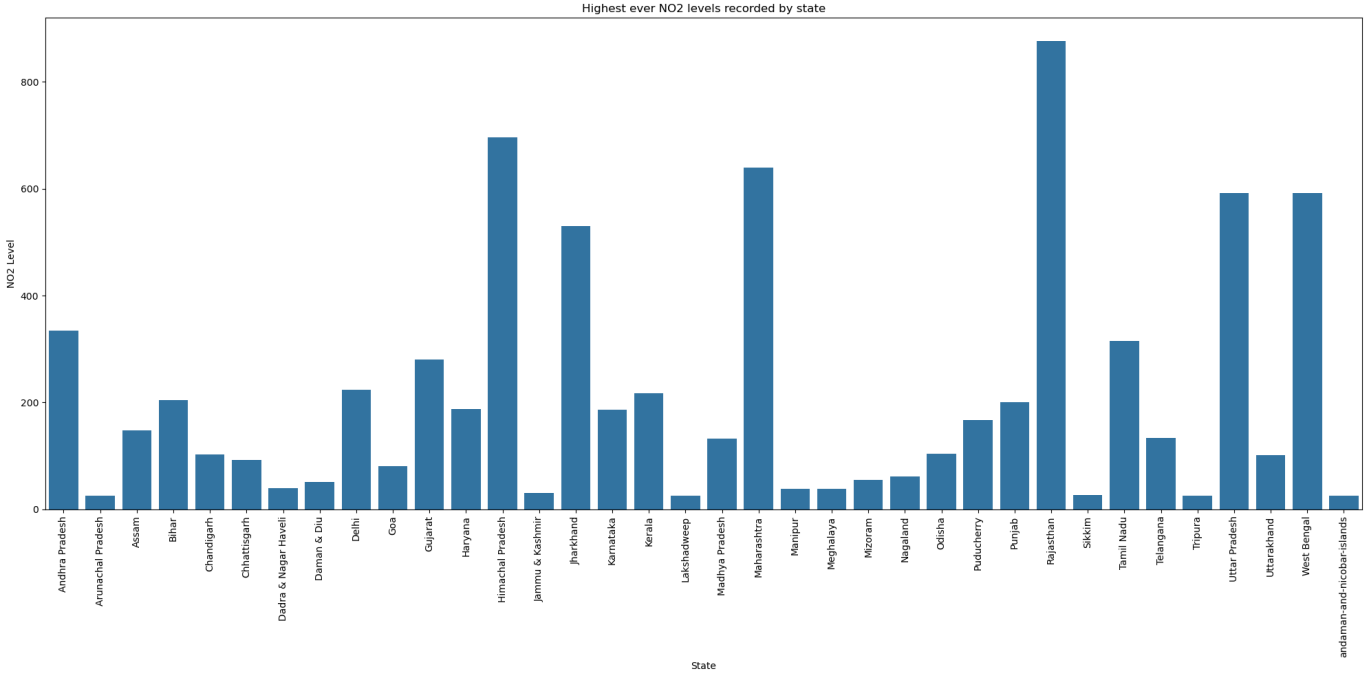


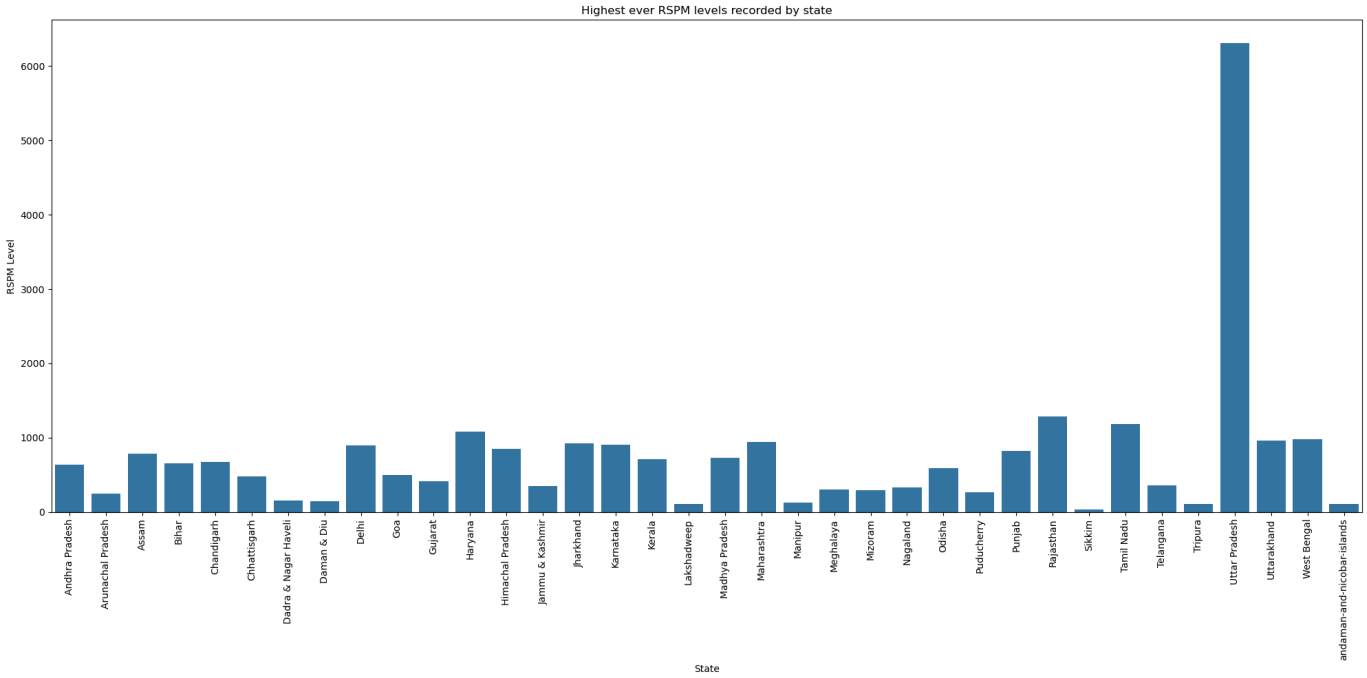
In [19]:

```
def type_avg(indicator=""):
    type_avg = data[VALUE_COLS + ['type', 'date']].groupby("type").mean(numeric_only=True)

    if indicator:
        t = type_avg[indicator].plot(kind='bar')
        plt.xticks(rotation=0)
        plt.title(f"Pollutant average by type for {indicator}")
    else:
        t = type_avg.plot(kind='bar')
        plt.xticks(rotation=0)
        plt.title("Pollutant average by type")

type_avg('so2')
```



In [20]:
```
def highest_levels_recorded(indicator="so2"):
    plt.figure(figsize=(20, 10))
    ind = data[['state', 'location', 'date', indicator]].copy()
    ind = ind.groupby('state', as_index=False)[indicator].max()

    sns.barplot(x='state', y=indicator, data=ind)
    plt.title(f"Highest ever {indicator.upper()} levels recorded by state")
    plt.xticks(rotation=90)
    plt.xlabel("State")
    plt.ylabel(f"{indicator.upper()} Level")
    plt.tight_layout()
    plt.show()

highest_levels_recorded("no2")
highest_levels_recorded("rspm")
```





In [22]:

```
def location_avgs(state, indicator="so2"):
    locs = data[VALUE_COLS + ['state', 'location', 'date']].groupby(['state', 'location']).me

    if state not in locs.index.get_level_values(0):
        print(f"State '{state}' not found in data.")
        return
    state_avgs = locs.loc[state].reset_index()

    plt.figure(figsize=(12, 6))
    sns.barplot(x='location', y=indicator, data=state_avgs)
    plt.title(f"Location-wise average for {indicator.upper()} in {state}")
    plt.xticks(rotation=90)
    plt.xlabel("Location")
    plt.ylabel(f"{indicator.upper()} Level")
    plt.tight_layout()
    plt.show()
location_avgs("Bihar", "no2")
```