In [1]:
```
#problem 10
# DATA ANALYTICS ON AIR QUALITY INDEX
# Q3 :-> NO DATA WAS AVALABLE OF YEAR 2021 I USE 2001
```

In [2]:
```
import pandas as pd
import numpy as np
```

In [3]:
```
#read csv file
air_quality=pd.read_csv("data.csv",encoding="ISO-8859-1")
```

```
C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:314
6: DtypeWarning: Columns (0) have mixed types.Specify dtype option on import or
set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

In [4]:
```
air_quality.head()
```

Out[4]:

	stn_code	sampling_date	state	location	agency	type	so2	no2	rspm	spm	locat
0	150	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	4.8	17.4	NaN	NaN	
1	151	February - M021990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	3.1	7.0	NaN	NaN	
2	152	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.2	28.5	NaN	NaN	
3	150	March - M031990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.3	14.7	NaN	NaN	
4	151	March - M031990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	4.7	7.5	NaN	NaN	

In [5]:

```
air_quality.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 [6]:

```
air_quality.isnull().sum()
```

Out[6]:

```
stn_code                       144077
sampling_date                       3
state                               0
location                            3
agency                         149481
type                             5393
so2                             34646
no2                             16233
rspm                            40222
spm                            237387
location_monitoring_station     27491
pm2_5                          426428
date                                7
dtype: int64
```

In [7]:

```
np.unique(air_quality['state'])
```

Out[7]:

```
array(['Andhra Pradesh', 'Arunachal Pradesh', 'Assam', 'Bihar',
       'Chandigarh', 'Chhattisgarh', 'Dadra & Nagar Haveli',
       'Daman & Diu', 'Delhi', 'Goa', 'Gujarat', 'Haryana',
       'Himachal Pradesh', 'Jammu & Kashmir', 'Jharkhand', 'Karnataka',
       'Kerala', 'Lakshadweep', 'Madhya Pradesh', 'Maharashtra',
       'Manipur', 'Meghalaya', 'Mizoram', 'Nagaland', 'Odisha',
       'Puducherry', 'Punjab', 'Rajasthan', 'Sikkim', 'Tamil Nadu',
       'Telangana', 'Tripura', 'Uttar Pradesh', 'Uttarakhand',
       'Uttaranchal', 'West Bengal', 'andaman-and-nicobar-islands'],
      dtype=object)
```

In [8]:
```
#removing unwanted columns
air_quality.drop(['agency','type','stn_code','location_monitoring_station'],axis=
```

In [9]:
```
air_quality.describe()
```

Out[9]:

	so2	no2	rspm	spm	pm2_5
count	401096.000000	419509.000000	395520.000000	198355.000000	9314.000000
mean	10.829414	25.809623	108.832784	220.783480	40.791467
std	11.177187	18.503086	74.872430	151.395457	30.832525
min	0.000000	0.000000	0.000000	0.000000	3.000000
25%	5.000000	14.000000	56.000000	111.000000	24.000000
50%	8.000000	22.000000	90.000000	187.000000	32.000000
75%	13.700000	32.200000	142.000000	296.000000	46.000000
max	909.000000	876.000000	6307.033333	3380.000000	504.000000

In [10]:
```
air_quality.columns
```

Out[10]:
```
Index(['sampling_date', 'state', 'location', 'so2', 'no2', 'rspm', 'spm',
       'pm2_5', 'date'],
      dtype='object')
```

In [11]:
```
air_quality.isnull().sum()
```

Out[11]:
```
sampling_date         3
state                 0
location              3
so2               34646
no2               16233
rspm              40222
spm              237387
pm2_5            426428
date                  7
dtype: int64
```

In [12]:
```
#fixing null values
air_quality["so2"].fillna(air_quality["so2"].mean(),inplace=True)
air_quality["no2"].fillna(air_quality["no2"].mean(),inplace=True)
air_quality["rspm"].fillna(air_quality["rspm"].mean(),inplace=True)
air_quality["spm"].fillna(air_quality["spm"].mean(),inplace=True)
air_quality["pm2_5"].fillna(air_quality["pm2_5"].mean(),inplace=True)
```

In [13]:
```
air_quality['date']=pd.to_datetime(air_quality['date'])
```

In [14]:
```
air_quality['year']=air_quality['date'].dt.year
```

In [15]:
```
air_quality['year'].fillna(method='ffill',inplace=True)
```

In [16]:

```
air_quality['year']=air_quality['year'].astype(int)
```

In [17]:

```
data=air_quality
```

In [18]:

```
def calculate_si(so2):
    si=0
    if (so2<=40):
     si= so2*(50/40)
    if (so2>40 and so2<=80):
     si= 50+(so2-40)*(50/40)
    if (so2>80 and so2<=380):
     si= 100+(so2-80)*(100/300)
    if (so2>380 and so2<=800):
     si= 200+(so2-380)*(100/800)
    if (so2>800 and so2<=1600):
     si= 300+(so2-800)*(100/800)
    if (so2>1600):
     si= 400+(so2-1600)*(100/800)
    return si
data['si']=data['so2'].apply(calculate_si)
df= data[['so2','si']]
```

In [19]:

```
def calculate_ni(no2):
    ni=0
    if(no2<=40):
     ni= no2*50/40
    elif(no2>40 and no2<=80):
     ni= 50+(no2-14)*(50/40)
    elif(no2>80 and no2<=180):
     ni= 100+(no2-80)*(100/100)
    elif(no2>180 and no2<=280):
     ni= 200+(no2-180)*(100/100)
    elif(no2>280 and no2<=400):
     ni= 300+(no2-280)*(100/120)
    else:
     ni= 400+(no2-400)*(100/120)
    return ni
data['ni']=data['no2'].apply(calculate_ni)
df= data[['no2','ni']]
```

In [20]:

```
def calculate_(rspm):
    rpi=0
    if(rpi<=30):
     rpi=rpi*50/30
    elif(rpi>30 and rpi<=60):
     rpi=50+(rpi-30)*50/30
    elif(rpi>60 and rpi<=90):
     rpi=100+(rpi-60)*100/30
    elif(rpi>90 and rpi<=120):
     rpi=200+(rpi-90)*100/30
    elif(rpi>120 and rpi<=250):
     rpi=300+(rpi-120)*(100/130)
    else:
     rpi=400+(rpi-250)*(100/130)
    return rpi
data['rpi']=data['rspm'].apply(calculate_si)
df= data[['rspm','rpi']]
```

In [21]:

```
def calculate_spi(spm):
    spi=0
    if(spm<=50):
     spi=spm
    if(spm<50 and spm<=100):
     spi=spm
    elif(spm>100 and spm<=250):
     spi= 100+(spm-100)*(100/150)
    elif(spm>250 and spm<=350):
     spi=200+(spm-250)
    elif(spm>350 and spm<=450):
     spi=300+(spm-350)*(100/80)
    else:
     spi=400+(spm-430)*(100/80)
    return spi
data['spi']=data['spm'].apply(calculate_spi)
df= data[['spm','spi']]
```

In [22]:

```
def calculate_aqi(si,ni,spi,rpi):
    aqi=0
    if(si>ni and si>spi and si>rpi):
     aqi=si
    if(spi>si and spi>ni and spi>rpi):
     aqi=spi
    if(ni>si and ni>spi and ni>rpi):
     aqi=ni
    if(rpi>si and rpi>ni and rpi>spi):
     aqi=rpi
    return aqi
data['AQI']=data.apply(lambda x:calculate_aqi(x['si'],x['ni'],x['spi'],x['rpi']),
df= data[['sampling_date','state','si','ni','rpi','spi','year','AQI']]
df.head()
```

Out[22]:

	sampling_date	state	si	ni	rpi	spi	year	AQI
0	February - M021990	Andhra Pradesh	6.000	21.750	109.610928	180.52232	1990	180.52232
1	February - M021990	Andhra Pradesh	3.875	8.750	109.610928	180.52232	1990	180.52232
2	February - M021990	Andhra Pradesh	7.750	35.625	109.610928	180.52232	1990	180.52232
3	March - M031990	Andhra Pradesh	7.875	18.375	109.610928	180.52232	1990	180.52232
4	March - M031990	Andhra Pradesh	5.875	9.375	109.610928	180.52232	1990	180.52232

In [23]:

```
df=df.sort_values(by="AQI",ascending=False)
```

In [24]:

```
year=df['year'].unique()
```

In [25]:

```
year=np.sort(year)
```

In [26]:

```
year=year.tolist()
```

In [27]:

```
df.head(10)
```

Out[27]:

	sampling_date	state	si	ni	rpi	spi	year	AQI
319916	April - M042001	Rajasthan	24.500000	120.000000	109.610928	4087.50	2001	4087.50
321441	29-04-04	Rajasthan	2.500000	23.750000	241.375000	3125.00	2004	3125.00
332783	27-05-10	Rajasthan	7.500000	83.312500	303.500000	2968.75	2010	2968.75
414008	June - M061996	West Bengal	14.000000	34.750000	109.610928	2820.00	1996	2820.00
102830	14-10-05	Haryana	13.536768	32.262029	101.666667	2628.75	2005	2628.75
332779	13-05-10	Rajasthan	7.350000	85.187500	167.333333	2495.00	2010	2495.00
332899	26-05-10	Rajasthan	7.900000	87.400000	232.125000	2381.25	2010	2381.25
332145	27-05-10	Rajasthan	5.000000	20.000000	243.500000	2303.75	2010	2303.75
67998	23-06-04	Delhi	13.536768	32.262029	183.333333	2218.75	2004	2218.75
320270	6/5/2003	Rajasthan	11.250000	20.000000	109.610928	2175.00	2003	2175.00

In [28]:
```
def calu(temp,i):
    temp1=temp.loc[(df["year"]==i)]
    temp1=temp1[["state","year","AQI"]].groupby(["state"]).median().sort_values(b
    print(temp1)
```

In [29]:
```
# Answer
for i in year:
    print("\n ****TOP 10 STATE IN YEAR %d ****"%(i))
    calu(df,i)
```

```
 ****TOP 10 STATE IN YEAR 1988 ****
                  year         AQI
state
Punjab            1988  336.250000
Uttar Pradesh     1988  313.125000
West Bengal       1988  275.500000
Haryana           1988  271.000000
Bihar             1988  267.000000
Himachal Pradesh  1988  209.000000
Gujarat           1988  197.333333
Madhya Pradesh    1988  197.333333
Delhi             1988  180.522320
Maharashtra       1988  180.522320

 ****TOP 10 STATE IN YEAR 1989 ****
                  year         AQI
state
Punjab            1989  476.250000
Uttar Pradesh     1989  302.500000
```

In [30]:
```
#problem 2
#monthwise Air quality of Delhi in year 2001
#no data fund for year 2021
air_quality["year"].unique()
```

Out[30]:
```
array([1990, 1991, 1992, 1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000,
       2001, 2002, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012,
       2013, 2014, 2015, 1987, 1988, 1989, 2003])
```

In [31]:
```
air_quality.head()
```

Out[31]:

	sampling_date	state	location	so2	no2	rspm	spm	pm2_5	date	year	
0	February - M021990	Andhra Pradesh	Hyderabad	4.8	17.4	108.832784	220.78348	40.791467	1990-02-01	1990	6
1	February - M021990	Andhra Pradesh	Hyderabad	3.1	7.0	108.832784	220.78348	40.791467	1990-02-01	1990	3
2	February - M021990	Andhra Pradesh	Hyderabad	6.2	28.5	108.832784	220.78348	40.791467	1990-02-01	1990	7
3	March - M031990	Andhra Pradesh	Hyderabad	6.3	14.7	108.832784	220.78348	40.791467	1990-03-01	1990	7
4	March - M031990	Andhra Pradesh	Hyderabad	4.7	7.5	108.832784	220.78348	40.791467	1990-03-01	1990	5

In [32]:
```
delhidata=air_quality.iloc[:,[0,1,3,4,5,6,7,8,9,-1]]
delhidata.columns
```

Out[32]:
```
Index(['sampling_date', 'state', 'so2', 'no2', 'rspm', 'spm', 'pm2_5', 'date',
       'year', 'AQI'],
      dtype='object')
```

In [33]:
```
delhidata=delhidata[delhidata["state"]=="Delhi"]
```

In [35]:
```
delhidata=delhidata.loc[(df["year"]==2001)]
```

In [38]:
```
delhidata['month']=delhidata['date'].dt.month
```

In [39]:
```
delhidata.columns
```

Out[39]:
```
Index(['sampling_date', 'state', 'so2', 'no2', 'rspm', 'spm', 'pm2_5', 'date',
       'year', 'AQI', 'month'],
      dtype='object')
```

In [40]:
```
delhidata.head(100)
```

Out[40]:

	sampling_date	state	so2	no2	rspm	spm	pm2_5	date	year	AQI	month
66987	January - M012001	Delhi	16.3	35.9	108.832784	278.0	40.791467	2001-01-01	2001	228.00	1
66988	February - M022001	Delhi	18.1	44.3	108.832784	367.0	40.791467	2001-02-01	2001	321.25	2
66989	March - M032001	Delhi	17.7	35.1	108.832784	280.0	40.791467	2001-03-01	2001	230.00	3
66990	April - M042001	Delhi	16.3	39.8	108.832784	342.0	40.791467	2001-04-01	2001	292.00	4
66991	May - M052001	Delhi	16.5	33.6	108.832784	285.0	40.791467	2001-05-01	2001	235.00	5
...	...	...	...	...	...	...	...	...	...	...	...
67081	August - M082001	Delhi	6.2	32.9	108.832784	307.0	40.791467	2001-08-01	2001	257.00	8
67082	September - M092001	Delhi	6.7	63.0	108.832784	522.0	40.791467	2001-09-01	2001	515.00	9
67083	October - M102001	Delhi	9.3	70.3	108.832784	652.0	40.791467	2001-10-01	2001	677.50	10
67084	November - M112001	Delhi	17.0	82.2	108.832784	790.0	40.791467	2001-11-01	2001	850.00	11
67085	December - M122001	Delhi	20.6	74.5	108.832784	506.0	40.791467	2001-12-01	2001	495.00	12

99 rows × 11 columns

In [41]:
```
temp1=delhidata[["so2","no2","rspm","spm","pm2_5","AQI","month"]].groupby(["month
```

In [42]:

```
#answer
temp1
```

Out[42]:

	so2	no2	rspm	spm	pm2_5	AQI
month						
1	15.30	35.90	108.832784	323.0	40.791467	273.000000
2	16.80	43.60	108.832784	365.0	40.791467	318.750000
3	14.80	35.50	108.832784	370.0	40.791467	325.000000
4	11.60	39.80	108.832784	428.0	40.791467	397.500000
5	10.75	35.00	108.832784	313.0	40.791467	263.000000
6	12.00	30.40	108.832784	205.0	40.791467	170.000000
7	11.90	25.35	108.832784	190.0	40.791467	160.000000
8	12.75	28.95	108.832784	231.0	40.791467	190.500000
9	12.25	33.15	108.832784	272.5	40.791467	223.333333
10	13.40	36.05	108.832784	333.5	40.791467	283.500000
11	14.50	40.45	108.832784	478.0	40.791467	460.000000
12	11.00	34.10	108.832784	383.0	40.791467	342.000000

In [ ]:

```
```