Your name:

Assignment Name: CA01 - Data Cleaning and Exploration of India Air Quality

Program Inititialization Section

Enter your import packages here

```
# import packages
import pandas as pd
import numpy as np
```

Data File Reading Section

Write code to read in data from external sources here

```
#read datasets
df = pd.read_csv('/content/data.csv',na_values=0, encoding='cp1252')
df
```

```
/usr/local/lib/python3.6/dist-packages/IPython/core/interactiveshell.py:2718: DtypeWarn
  interactivity=interactivity, compiler=compiler, result=result)
```

	stn_code	sampling_date	state	location	agency	type	so2	no
0	150	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	4.8	17.
1	151	February - M021990	Andhra Pradesh	Hyderabad	NaN	Industrial Area	3.1	7.
2	152	February - M021990	Andhra Pradesh	Hyderabad	NaN	Residential, Rural and other Areas	6.2	28.
						Residential		

Initial Data Investigation Section

Summarized details

Generate descriptive statistics that summarize the central tendency, dispersion, and shape of a dataset’s distribution, excluding NaN values.

Steps:

1. Statistical Description of data (data.describe)

2. Display number of total rows and columns of the dataset (data.shape)

3. Display number of non-null values for each column (data.count)

4. Display number of null values for each column (sum of data.isnull)

5. Display range, column, number of non-null objects of each column, datatype and memory usage (data.info)

6. Display Top 10 and Bottom 10 records (head and tail)

```
# Your code for this section here ...
df.describe()
```

	so2	no2	rspm	spm	pm2_5
count	400221.000000	418724.000000	394754.000000	197142.000000	9314.000000
mean	10.853091	25.858009	109.043969	222.141944	40.791467
std	11.177910	18.486613	74.791245	150.863648	30.832525

```
df.shape
```

```
(435742, 13)
```

50% 8.000000 22.000000 90.000000 188.000000 32.000000

```
df.count()
```

```
stn_code                       291665
sampling_date                  435739
state                          435742
location                       435739
agency                         286261
type                           430349
so2                            400221
no2                            418724
rspm                           394754
spm                            197142
location_monitoring_station    408251
pm2_5                            9314
date                           435735
dtype: int64
```

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

```
stn_code                       144077
sampling_date                       3
state                               0
location                            3
agency                         149481
type                             5393
so2                             35521
no2                             17018
rspm                            40988
spm                            238600
location_monitoring_station     27491
pm2_5                          426428
date                                7
dtype: int64
```

```
df.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                          400221 non-null  float64
 7   no2                          418724 non-null  float64
 8   rspm                         394754 non-null  float64
 9   spm                          197142 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
```

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

	stn_code	sampling_date	state	location	agency	type	so2	no2	rspm	spr
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
						Residential				

```
df.tail(10)
```

	stn_code	sampling_date	state	location	agency	type	so2	no2	r
435732	SAMP	09-12-15	West Bengal	ULUBERIA	West Bengal State Pollution Control Board	RIRUO	22.0	50.0	1
435733	SAMP	12-12-15	West Bengal	ULUBERIA	West Bengal State Pollution	RIRUO	34.0	61.0	1
435735	SAMP	18-12-15	West Bengal	ULUBERIA	[illegible]	RIRUO	17.0	44.0	1

Cleansing the dataset

Dropping of less valued columns:

1. stn_code, agency, sampling_date, location_monitoring_agency do not add much value to the dataset in terms of information. Therefore, we can drop those columns.
2. Dropping rows where no date is available.

```
# Cleaning up the data

#dropping columns that aren't required

# ... your code here

# dropping rows where no date is available

# ... your code here

#data columns:  stn_code  sampling_date state location  agency  type so2  no2 rspm  spm locat
#Drop stn_code, agency, sampling_date, location_monitoring_agency
#New data: state  location type so2 no2 rspm  spm pm2_5 date

df = df.loc[:,['state', 'location', 'type', 'so2',  'no2',  'rspm', 'spm', 'pm2_5', 'date']]


# displaying final columns (data.columns)
df.columns

# ... your code here
#Drop columns where there is NaN in Date

#Doesn't work
#df['date'].dropna(how = "all", axis = 0,inplace = True)
df = df[df['date'].notnull()]
```

```
#No null values in those columns
df.isnull().sum()
```

```
state            0
location         0
type          5390
so2          35518
no2          17015
rspm         40985
spm         238593
pm2_5       426421
date             0
dtype: int64
```

```
df
```

	state	location	type	so2	no2	rspm	spm	pm2_5	date
0	Andhra Pradesh	Hyderabad	Residential, Rural and other Areas	4.8	17.4	NaN	NaN	NaN	1990-02-01
1	Andhra Pradesh	Hyderabad	Industrial Area	3.1	7.0	NaN	NaN	NaN	1990-02-01
2	Andhra Pradesh	Hyderabad	Residential, Rural and other Areas	6.2	28.5	NaN	NaN	NaN	1990-02-01
3	Andhra Pradesh	Hyderabad	Residential, Rural and other Areas	6.3	14.7	NaN	NaN	NaN	1990-03-01
4	Andhra Pradesh	Hyderabad	Industrial Area	4.7	7.5	NaN	NaN	NaN	1990-03-01
...	...	...	...	...	...	...	...	...	...
435734	West Bengal	ULUBERIA	RIRUO	20.0	44.0	148.0	NaN	NaN	2015-12-15
435735	West Bengal	ULUBERIA	RIRUO	17.0	44.0	131.0	NaN	NaN	2015-12-18

Changing the types to uniform format:

Notice that the ‘type’ column has values such as ‘Industrial Area’ and ‘Industrial Areas’—both actually mean the same, so let’s remove such type of stuff and make it uniform. Replace the 'type' values with standard codes as follows:

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)

```
# ... Your code here
#df['type'].replace({ "Residential": "R", "Residential and others": "RO", "Residential, Rural

types = { "Residential": "R", "Residential and others": "RO", "Residential, Rural and other A

df.type = df.type.replace(types)


df
```

	state	location	type	so2	no2	rspm	spm	pm2_5	date
0	Andhra Pradesh	Hyderabad	RRO	4.8	17.4	NaN	NaN	NaN	1990-02-01
1	Andhra Pradesh	Hyderabad	I	3.1	7.0	NaN	NaN	NaN	1990-02-01
2	Andhra Pradesh	Hyderabad	RRO	6.2	28.5	NaN	NaN	NaN	1990-02-01
3	Andhra Pradesh	Hyderabad	RRO	6.3	14.7	NaN	NaN	NaN	1990-03-01
4	Andhra Pradesh	Hyderabad	I	4.7	7.5	NaN	NaN	NaN	1990-03-01
...	...	...	...	...	...	...	...	...	...
435734	West Bengal	ULUBERIA	RIRUO	20.0	44.0	148.0	NaN	NaN	2015-12-15
435735	West Bengal	ULUBERIA	RIRUO	17.0	44.0	131.0	NaN	NaN	2015-12-18
435736	West Bengal	ULUBERIA	RIRUO	18.0	45.0	140.0	NaN	NaN	2015-12-21
435737	West Bengal	ULUBERIA	RIRUO	22.0	50.0	143.0	NaN	NaN	2015-12-24
435738	West Bengal	ULUBERIA	RIRUO	20.0	46.0	171.0	NaN	NaN	2015-12-29

435735 rows × 9 columns

```
# Display top 10 records after codification of 'types'
# ... Your code here
#
df.head(10)
```

	state	location	type	so2	no2	rspm	spm	pm2_5	date
0	Andhra Pradesh	Hyderabad	RRO	4.8	17.4	NaN	NaN	NaN	1990-02-01

Creating a year column

To view the trend over a period of time, we need year values for each row and also when you see in most of the values in date column only has ‘year’ value. So, let’s create a new column holding year values. Convert the column to 'datetime' type and extract the year to populate the new column. Display Top 5 records after the conversion.

6 Andhra Pradesh Hyderabad RRO 5.4 17.1 NaN NaN NaN 1990-04-01

```
# ... Your code here
#convert to datetime
df['date']= pd.to_datetime(df['date'])
#Extract year to populate new column year
df['year'] = df['date'].dt.year
```

```
df
```

	state	location	type	so2	no2	rspm	spm	pm2_5	date	year
0	Andhra Pradesh	Hyderabad	RRO	4.8	17.4	NaN	NaN	NaN	1990-02-01	1990
1	Andhra Pradesh	Hyderabad	I	3.1	7.0	NaN	NaN	NaN	1990-02-01	1990
2	Andhra Pradesh	Hyderabad	RRO	6.2	28.5	NaN	NaN	NaN	1990-02-01	1990
3	Andhra Pradesh	Hyderabad	RRO	6.3	14.7	NaN	NaN	NaN	1990-03-01	1990
4	Andhra Pradesh	Hyderabad	I	4.7	7.5	NaN	NaN	NaN	1990-03-01	1990
...	...	...	...	...	...	...	...	...	...	...
435734	West Bengal	ULUBERIA	RIRUO	20.0	44.0	148.0	NaN	NaN	2015-12-15	2015
435735	West Bengal	ULUBERIA	RIRUO	17.0	44.0	131.0	NaN	NaN	2015-12-18	2015

Handling Missing Values

The column such as SO2, NO2, rspm, spm, pm2_5 are the ones which contribute much to our analysis. So, we need to remove null from those columns to avoid inaccuracy in the prediction. We use the Imputer from sklearn.preprocessing to fill the missing values in every column with the

```
# define columns of importance, which shall be used reguarly (COLS = ....)
# invoke SimpleImputer to fill missing values using 'mean' as the replacement strategy
# Display data.info after the transformation
# Display that there are no more missing values in the dataset
```

```
# ... your code here
#remove null values from SO2, NO2, rspm, spm, pm2_5
#fill missing value with imputer from sklearn
from sklearn.impute import SimpleImputer
#imp = SimpleImputer(missing_values=-1, strategy='mean')
dfnew = df.loc[:,['so2','no2', 'rspm', 'spm', 'pm2_5']]
imp = SimpleImputer(missing_values = np.nan, strategy='mean')
#Fit then transform
imp.fit(dfnew)
dftransformed = imp.transform(dfnew)
dftransformednew = imp.transform(dftransformed)
#Dataframe turned into nested list arrays
#now convert it back into it being a dataframe
dfnew = pd.DataFrame(dftransformednew)
dfnew.columns = ['so2','no2', 'rspm', 'spm', 'pm2_5']

#Drop 'so2','no2', 'rspm', 'spm', 'pm2_5' in original dataframe
df = df.loc[:,['state', 'location', 'type', 'date', 'year']]
result = pd.concat([df, dfnew], axis=1)

#Figure out a way to do this without dropping columns !!! DO LATER
#result2 = pd.merge(df, dfnew, how='right')
```

Statewise Grouping of so2, no2, rspm, spm values

Calculate median values of so2, no2, rspm, spm for each state and display in (a) as table (b) bar chart, with values sorted in ascending order. Separate section for each of the component. Use matplotlib().

```
result
check = result.groupby('state')['so2'].agg(['median'])
```

so2 status

```
# ... Your code here
import matplotlib.pyplot as plt
dfso2 = result.groupby('state')['so2'].agg(['median'])
```

```
#Sort ascending table then barplot
dfso2 = dfso2.sort_values(by='median')
dfso2['state'] = dfso2.index
dfso2.state.count()

n = 34
ind = np.arange(n)
width = 0.35
dfso2med = dfso2['median']

plt.bar(ind, dfso2med, width)

plt.xticks(ind+width / 2, dfso2.index, rotation=90)
```

```
([<matplotlib.axis.XTick at 0x7f911dd87630>,
  <matplotlib.axis.XTick at 0x7f911dd875f8>,
  <matplotlib.axis.XTick at 0x7f911dd874a8>,
  <matplotlib.axis.XTick at 0x7f911dcfe400>,
  <matplotlib.axis.XTick at 0x7f911dcfe9e8>,
  <matplotlib.axis.XTick at 0x7f911dcfee80>,
  <matplotlib.axis.XTick at 0x7f911dcfe6d8>,
  <matplotlib.axis.XTick at 0x7f911dd09128>,
  <matplotlib.axis.XTick at 0x7f911dd09898>,
  <matplotlib.axis.XTick at 0x7f911dd09d30>,
  <matplotlib.axis.XTick at 0x7f911dc93208>,
  <matplotlib.axis.XTick at 0x7f911dc936a0>,
  <matplotlib.axis.XTick at 0x7f911dc93b38>,
  <matplotlib.axis.XTick at 0x7f911dc93d68>,
  <matplotlib.axis.XTick at 0x7f911dc9a4a8>,
  <matplotlib.axis.XTick at 0x7f911dc9a940>,
  <matplotlib.axis.XTick at 0x7f911dc93b00>,
  <matplotlib.axis.XTick at 0x7f911dd09860>,
  <matplotlib.axis.XTick at 0x7f911dc9a400>,
  <matplotlib.axis.XTick at 0x7f911dc9ad68>,
  <matplotlib.axis.XTick at 0x7f911dca24a8>,
  <matplotlib.axis.XTick at 0x7f911dca2940>,
  <matplotlib.axis.XTick at 0x7f911dca2dd8>,
  <matplotlib.axis.XTick at 0x7f911dcab2b0>,
  <matplotlib.axis.XTick at 0x7f911dcab748>,
  <matplotlib.axis.XTick at 0x7f911dca2898>,
  <matplotlib.axis.XTick at 0x7f911dc9ae48>,
  <matplotlib.axis.XTick at 0x7f911dcaba90>,
  <matplotlib.axis.XTick at 0x7f911dcab208>,
  <matplotlib.axis.XTick at 0x7f911dcb4320>,
  <matplotlib.axis.XTick at 0x7f911dcb47b8>,
  <matplotlib.axis.XTick at 0x7f911dcb4c50>,
  <matplotlib.axis.XTick at 0x7f911dcbe160>,
  <matplotlib.axis.XTick at 0x7f911dcbe5c0>],
 [Text(0, 0, 'Nagaland'),
  Text(0, 0, 'Mizoram'),
  Text(0, 0, 'Meghalaya'),
  Text(0, 0, 'Arunachal Pradesh'),
  Text(0, 0, 'Himachal Pradesh'),
  Text(0, 0, 'Kerala'),
  Text(0, 0, 'Telangana'),
  Text(0, 0, 'Odisha'),
  Text(0, 0, 'Jammu & Kashmir'),
  Text(0, 0, 'Andhra Pradesh'),
  Text(0, 0, 'Goa'),
  Text(0, 0, 'Rajasthan'),
  Text(0, 0, 'Delhi'),
  Text(0, 0, 'Assam'),
  Text(0, 0, 'Puducherry'),
  Text(0, 0, 'Daman & Diu'),
  Text(0, 0, 'Chandigarh'),
  Text(0, 0, 'West Bengal'),
  Text(0, 0, 'Dadra & Nagar Haveli'),
  Text(0, 0, 'Karnataka'),
```

no2 status

```
       Text(0, 0, 'Manipur'),
```

```
# ... Your code here
dfno2 = result.groupby('state')['no2'].agg(['median'])
#Sort ascending table then barplot
dfno2 = dfno2.sort_values(by='median')
dfno2['state'] = dfno2.index
dfno2.state.count()

n = 34
ind = np.arange(n)
width = 0.35
dfno2med = dfno2['median']

plt.bar(ind, dfno2med, width)

plt.xticks(ind+width / 2, dfno2.index, rotation=90)
```

```
([<matplotlib.axis.XTick at 0x7f911dc74400>,
  <matplotlib.axis.XTick at 0x7f911dc744e0>,
  <matplotlib.axis.XTick at 0x7f911dc74550>,
  <matplotlib.axis.XTick at 0x7f911dbf9b38>,
  <matplotlib.axis.XTick at 0x7f911dbf9e80>,
  <matplotlib.axis.XTick at 0x7f911dc06470>,
  <matplotlib.axis.XTick at 0x7f911dc06908>,
  <matplotlib.axis.XTick at 0x7f911dc06da0>,
  <matplotlib.axis.XTick at 0x7f911db8e278>,
  <matplotlib.axis.XTick at 0x7f911db8e710>,
  <matplotlib.axis.XTick at 0x7f911db8eba8>,
  <matplotlib.axis.XTick at 0x7f911db970f0>,
  <matplotlib.axis.XTick at 0x7f911db97518>,
  <matplotlib.axis.XTick at 0x7f911db8e668>,
  <matplotlib.axis.XTick at 0x7f911dc069b0>,
  <matplotlib.axis.XTick at 0x7f911db97898>,
  <matplotlib.axis.XTick at 0x7f911db97ba8>,
  <matplotlib.axis.XTick at 0x7f911dba10f0>,
  <matplotlib.axis.XTick at 0x7f911dba1518>,
  <matplotlib.axis.XTick at 0x7f911dba19b0>,
  <matplotlib.axis.XTick at 0x7f911dba1e48>,
  <matplotlib.axis.XTick at 0x7f911dba9320>,
  <matplotlib.axis.XTick at 0x7f911dba97b8>,
  <matplotlib.axis.XTick at 0x7f911dba1f28>,
  <matplotlib.axis.XTick at 0x7f911db97400>,
  <matplotlib.axis.XTick at 0x7f911dba9860>,
  <matplotlib.axis.XTick at 0x7f911dba9e48>,
  <matplotlib.axis.XTick at 0x7f911dbb2320>,
  <matplotlib.axis.XTick at 0x7f911dbb27b8>,
  <matplotlib.axis.XTick at 0x7f911dbb2c50>,
  <matplotlib.axis.XTick at 0x7f911dbba160>,
  <matplotlib.axis.XTick at 0x7f911dbba5c0>,
  <matplotlib.axis.XTick at 0x7f911dbbaa58>,
  <matplotlib.axis.XTick at 0x7f911dbb2ac8>],
 [Text(0, 0, 'Sikkim'),
  Text(0, 0, 'Arunachal Pradesh'),
  Text(0, 0, 'Mizoram'),
  Text(0, 0, 'Nagaland'),
  Text(0, 0, 'Meghalaya'),
  Text(0, 0, 'Goa'),
  Text(0, 0, 'Puducherry'),
  Text(0, 0, 'Jammu & Kashmir'),
  Text(0, 0, 'Himachal Pradesh'),
  Text(0, 0, 'Kerala'),
  Text(0, 0, 'Assam'),
  Text(0, 0, 'Odisha'),
  Text(0, 0, 'Daman & Diu'),
  Text(0, 0, 'Dadra & Nagar Haveli'),
  Text(0, 0, 'Chandigarh'),
  Text(0, 0, 'Manipur'),
  Text(0, 0, 'Madhya Pradesh'),
  Text(0, 0, 'Andhra Pradesh'),
  Text(0, 0, 'Tamil Nadu'),
  Text(0, 0, 'Karnataka'),
  Text(0, 0, 'Chhattisgarh'),
  Text(0, 0, 'Telangana')
```

```
        Text(0, 0, 'Uttarakhand'),
        Text(0, 0, 'Rajasthan'),
```

rspm status

```
# ... Your code here
dfrspm = result.groupby('state')['rspm'].agg(['median'])
#Sort ascending table then barplot
dfrspm = dfrspm.sort_values(by='median')
dfrspm['state'] = dfrspm.index
dfrspm.state.count()

n = 34
ind = np.arange(n)
width = 0.35
dfrspmmed = dfrspm['median']

plt.bar(ind, dfrspmmed, width)

plt.xticks(ind+width / 2, dfrspm.index, rotation=90)
```

```
([<matplotlib.axis.XTick at 0x7f911db75748>,
  <matplotlib.axis.XTick at 0x7f911db755f8>,
  <matplotlib.axis.XTick at 0x7f911db75908>,
  <matplotlib.axis.XTick at 0x7f911e9fec50>,
  <matplotlib.axis.XTick at 0x7f911e9fe208>,
  <matplotlib.axis.XTick at 0x7f911e99fba8>,
  <matplotlib.axis.XTick at 0x7f911e99f630>,
  <matplotlib.axis.XTick at 0x7f911e99f4a8>,
  <matplotlib.axis.XTick at 0x7f911e9ad828>,
  <matplotlib.axis.XTick at 0x7f911e9adc18>,
  <matplotlib.axis.XTick at 0x7f911e99f128>,
  <matplotlib.axis.XTick at 0x7f911e9ad390>,
  <matplotlib.axis.XTick at 0x7f911e9ada20>,
  <matplotlib.axis.XTick at 0x7f911e9b85c0>,
  <matplotlib.axis.XTick at 0x7f911e9b8630>,
  <matplotlib.axis.XTick at 0x7f911e9b87f0>,
  <matplotlib.axis.XTick at 0x7f911ea2e940>,
  <matplotlib.axis.XTick at 0x7f911ea2e278>,
  <matplotlib.axis.XTick at 0x7f911ea2ef60>,
  <matplotlib.axis.XTick at 0x7f911ea2ea20>,
  <matplotlib.axis.XTick at 0x7f911e9b8128>,
  <matplotlib.axis.XTick at 0x7f911e99f390>,
  <matplotlib.axis.XTick at 0x7f911e9d86a0>,
  <matplotlib.axis.XTick at 0x7f911e9d89e8>,
  <matplotlib.axis.XTick at 0x7f911ea12198>,
  <matplotlib.axis.XTick at 0x7f911ea120f0>,
  <matplotlib.axis.XTick at 0x7f911ea122b0>,
  <matplotlib.axis.XTick at 0x7f911ea73518>,
  <matplotlib.axis.XTick at 0x7f911ea73eb8>,
  <matplotlib.axis.XTick at 0x7f911ea12a90>,
  <matplotlib.axis.XTick at 0x7f911e9ad7f0>,
  <matplotlib.axis.XTick at 0x7f911ea738d0>,
  <matplotlib.axis.XTick at 0x7f911ea48e48>,
  <matplotlib.axis.XTick at 0x7f911ea48710>],
 [Text(0, 0, 'Mizoram'),
  Text(0, 0, 'Puducherry'),
  Text(0, 0, 'Kerala'),
  Text(0, 0, 'Meghalaya'),
  Text(0, 0, 'Goa'),
  Text(0, 0, 'Tamil Nadu'),
  Text(0, 0, 'Manipur'),
  Text(0, 0, 'Karnataka'),
  Text(0, 0, 'Arunachal Pradesh'),
  Text(0, 0, 'Andhra Pradesh'),
  Text(0, 0, 'Assam'),
  Text(0, 0, 'Nagaland'),
  Text(0, 0, 'Telangana'),
  Text(0, 0, 'Odisha'),
  Text(0, 0, 'Chandigarh'),
  Text(0, 0, 'Himachal Pradesh'),
  Text(0, 0, 'Dadra & Nagar Haveli'),
  Text(0, 0, 'Maharashtra'),
  Text(0, 0, 'Gujarat'),
  Text(0, 0, 'West Bengal'),
  Text(0, 0, 'Chhattisgarh'),
  Text(0, 0, 'Daman & Diu'),
  Text(0, 0, 'Bihar'),
       Text(0, 0, 'Sikkim'),
```

spm status

```
       Text(0, 0, 'Jammu & Kashmir'),
```

```
# ... Your code here
dfspm = result.groupby('state')['spm'].agg(['median'])
#Sort ascending table then barplot
dfspm = dfspm.sort_values(by='median')
dfspm['state'] = dfspm.index
dfspm.state.count()

n = 34
ind = np.arange(n)
width = 0.35
dfspmmed = dfspm['median']

plt.bar(ind, dfspmmed, width)

plt.xticks(ind+width / 2, dfspm.index, rotation=90)
```

```
([<matplotlib.axis.XTick at 0x7f911ebdab70>,
  <matplotlib.axis.XTick at 0x7f911ebdabe0>,
  <matplotlib.axis.XTick at 0x7f911ebda908>,
  <matplotlib.axis.XTick at 0x7f911ed12da0>,
  <matplotlib.axis.XTick at 0x7f911ed124a8>,
  <matplotlib.axis.XTick at 0x7f911ed12518>,
  <matplotlib.axis.XTick at 0x7f911ed85d68>,
  <matplotlib.axis.XTick at 0x7f911ed60c88>,
  <matplotlib.axis.XTick at 0x7f911ed60080>,
  <matplotlib.axis.XTick at 0x7f911ed60ac8>,
  <matplotlib.axis.XTick at 0x7f911ee0a080>,
  <matplotlib.axis.XTick at 0x7f911ed60908>,
  <matplotlib.axis.XTick at 0x7f911ed12a20>,
  <matplotlib.axis.XTick at 0x7f911ee0a1d0>,
  <matplotlib.axis.XTick at 0x7f911ee0a0f0>,
  <matplotlib.axis.XTick at 0x7f911ee0a860>,
  <matplotlib.axis.XTick at 0x7f911ee02ef0>,
  <matplotlib.axis.XTick at 0x7f911ee02588>,
  <matplotlib.axis.XTick at 0x7f911ee02cc0>,
  <matplotlib.axis.XTick at 0x7f911edb6198>,
  <matplotlib.axis.XTick at 0x7f911edb6dd8>,
  <matplotlib.axis.XTick at 0x7f911ee02240>,
  <matplotlib.axis.XTick at 0x7f911ee0a160>,
  <matplotlib.axis.XTick at 0x7f911edb6b70>,
  <matplotlib.axis.XTick at 0x7f911edb6320>,
  <matplotlib.axis.XTick at 0x7f911edad518>,
  <matplotlib.axis.XTick at 0x7f911edad898>,
  <matplotlib.axis.XTick at 0x7f911edad208>,
  <matplotlib.axis.XTick at 0x7f911ede5d30>,
  <matplotlib.axis.XTick at 0x7f911ede50b8>,
  <matplotlib.axis.XTick at 0x7f911edadf28>,
  <matplotlib.axis.XTick at 0x7f911edb6828>,
  <matplotlib.axis.XTick at 0x7f911ede54a8>,
  <matplotlib.axis.XTick at 0x7f911ede5f28>],
 [Text(0, 0, 'Manipur'),
  Text(0, 0, 'Puducherry'),
  Text(0, 0, 'Daman & Diu'),
  Text(0, 0, 'Sikkim'),
  Text(0, 0, 'Andhra Pradesh'),
  Text(0, 0, 'Meghalaya'),
  Text(0, 0, 'Mizoram'),
  Text(0, 0, 'Nagaland'),
  Text(0, 0, 'Odisha'),
  Text(0, 0, 'Punjab'),
  Text(0, 0, 'Rajasthan'),
  Text(0, 0, 'Tamil Nadu'),
  Text(0, 0, 'Telangana'),
  Text(0, 0, 'Uttar Pradesh'),
  Text(0, 0, 'Uttarakhand'),
  Text(0, 0, 'Maharashtra'),
  Text(0, 0, 'Madhya Pradesh'),
  Text(0, 0, 'Kerala'),
  Text(0, 0, 'Karnataka'),
  Text(0, 0, 'Jharkhand'),
  Text(0, 0, 'Jammu & Kashmir'),
  Text(0, 0, 'Himachal Pradesh'),
  Text(0, 0, 'Haryana'),
  Text(0, 0, 'Gujarat'),
  Text(0, 0, 'Goa'),
  Text(0, 0, 'Dadra & Nagar Haveli'),
  Text(0, 0, 'Chhattisgarh'),
```

What is the yearly trend in a particular state, say ‘Andhra Pradesh’?

Create a new dataframe containing the NO2, SO2, rspm, and spm data regarding state ‘Andhra Pradesh’ only and group it by ‘year’. Display top 5 records after.

```
        Text(0, 0, 'Uttaranchal')])
```

```
result
```

	state	location	type	date	year	so2	no2	rspm	spm	
0	Andhra Pradesh	Hyderabad	RRO	1990-02-01	1990.0	4.8	17.4	109.044278	222.141944	40.
1	Andhra Pradesh	Hyderabad	I	1990-02-01	1990.0	3.1	7.0	109.044278	222.141944	40.
2	Andhra Pradesh	Hyderabad	RRO	1990-02-01	1990.0	6.2	28.5	109.044278	222.141944	40.
3	Andhra Pradesh	Hyderabad	RRO	1990-03-01	1990.0	6.3	14.7	109.044278	222.141944	40.
4	Andhra Pradesh	Hyderabad	I	1990-03-01	1990.0	4.7	7.5	109.044278	222.141944	40.
...	...	...	...	...	...	...	...	...	...	
435734	West Bengal	ULUBERIA	RIRUO	2015-12-15	2015.0	20.0	46.0	171.000000	222.141944	40.
435735	West Bengal	ULUBERIA	RIRUO	2015-12-18	2015.0	NaN	NaN	NaN	NaN	

```
dfandhranew
```

	no2	so2	rspm	spm	year
	mean	mean	mean	mean	
year					
1990.0	15.296552	8.868966	109.044278	165.984875	1990.0
1991.0	18.827778	10.811111	109.044278	154.503943	1991.0
1992.0	36.013978	19.766738	109.044278	210.643678	1992.0
1993.0	15.961111	8.544444	109.044278	222.141944	1993.0
1994.0	18.273016	10.760317	109.044278	178.677526	1994.0
1995.0	33.531868	17.225275	109.044278	167.293513	1995.0
1996.0	36.589482	19.861099	109.044278	180.345095	1996.0
1997.0	38.163330	20.339507	109.044278	177.785714	1997.0
1998.0	26.409459	11.491892	109.044278	175.351351	1998.0
1999.0	19.887963	14.384259	109.044278	197.509259	1999.0
2000.0	23.882019	13.856973	109.044278	140.935185	2000.0
2001.0	27.105714	12.000952	109.044278	142.790476	2001.0
2002.0	24.976190	7.506667	109.044278	85.866667	2002.0
2004.0	31.430507	7.338767	85.384298	170.814559	2004.0
2005.0	30.081095	6.370926	79.468608	220.257282	2005.0

```
# ... Your code here
dfandhra = result.loc[result.state == 'Andhra Pradesh', ['no2', 'so2', 'rspm', 'spm','year']]
dfandhranew = dfandhra.groupby('year').agg(['mean'])


# Display yearly trend graph (year vs. value) in pairs: (a) so2 and no2 (b) rspm and spm.
# So, you will display TWO graphs altogether.

# ... Your code here

so2mean = dfandhranew['so2']
no2mean = dfandhranew['no2']

plt.plot(so2mean, ls= '--',color='red',marker='s', markerfacecolor= 'white',label='so2')
plt.plot(no2mean, ls= '-.', color='g', marker='o', markerfacecolor='grey',label='no2')
plt.legend()
plt.title('Mean NO2 vs SO2 for Andhra Pradesh')
plt.xlabel('Year')
plt.ylabel('Gas Level')
plt.show()
```