Perform the following operations using Python on the Air quality data sets

a. Data cleaning b. Data integration c. Data transformation d. Error correcting e. Data model building

In [2]:

```
import pandas as pd
import numpy as np
```

In [5]:

```
df = pd.read_csv('AirQualityUCI.csv',sep=';')
```

In [6]:

```
df
```

Out[6]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx
0	10/03/2004	18.00.00	2,6	1360.0	150.0	11,9	1046.0	1
1	10/03/2004	19.00.00	2	1292.0	112.0	9,4	955.0	1
2	10/03/2004	20.00.00	2,2	1402.0	88.0	9,0	939.0	1
3	10/03/2004	21.00.00	2,2	1376.0	80.0	9,2	948.0	1
4	10/03/2004	22.00.00	1,6	1272.0	51.0	6,5	836.0	1
...	...	...	...	...	...	...	...	
9466	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9467	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9468	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9469	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9470	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

9471 rows × 17 columns

a. Data cleaning

a.1 Removing Missing or Null Values:

In [10]:

```
df.dropna(axis=0,how='any')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9471 entries, 0 to 9470
Data columns (total 17 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Date           9357 non-null   object 
 1   Time           9357 non-null   object 
 2   CO(GT)         9357 non-null   object 
 3   PT08.S1(CO)    9357 non-null   float64
 4   NMHC(GT)       9357 non-null   float64
 5   C6H6(GT)       9357 non-null   object 
 6   PT08.S2(NMHC)  9357 non-null   float64
 7   NOx(GT)        9357 non-null   float64
 8   PT08.S3(NOx)   9357 non-null   float64
 9   NO2(GT)        9357 non-null   float64
 10  PT08.S4(NO2)   9357 non-null   float64
 11  PT08.S5(O3)    9357 non-null   float64
 12  T              9357 non-null   object 
 13  RH             9357 non-null   object 
 14  AH             9357 non-null   object 
 15  Unnamed: 15    0 non-null      float64
 16  Unnamed: 16    0 non-null      float64
dtypes: float64(10), object(7)
memory usage: 1.2+ MB
```

a.2 Reading and Removing Duplicate Values

- Reading Duplicates:

In [20]:

```
df.duplicated(subset=['CO(GT)'])
```

Out[20]:

```
0       False
1       False
2       False
3        True
4       False
        ...
9466     True
9467     True
9468     True
9469     True
9470     True
Length: 9471, dtype: bool
```

- Remove Duplicates:

In [24]:

```
df.drop_duplicates(keep=False)
```

Out[24]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx
0	10/03/2004	18.00.00	2,6	1360.0	150.0	11,9	1046.0	1
1	10/03/2004	19.00.00	2	1292.0	112.0	9,4	955.0	1
2	10/03/2004	20.00.00	2,2	1402.0	88.0	9,0	939.0	1
3	10/03/2004	21.00.00	2,2	1376.0	80.0	9,2	948.0	1
4	10/03/2004	22.00.00	1,6	1272.0	51.0	6,5	836.0	1
...	...	...	...	...	...	...	...	
9352	04/04/2005	10.00.00	3,1	1314.0	-200.0	13,5	1101.0	4
9353	04/04/2005	11.00.00	2,4	1163.0	-200.0	11,4	1027.0	3
9354	04/04/2005	12.00.00	2,4	1142.0	-200.0	12,4	1063.0	2
9355	04/04/2005	13.00.00	2,1	1003.0	-200.0	9,5	961.0	2
9356	04/04/2005	14.00.00	2,2	1071.0	-200.0	11,9	1047.0	2

9357 rows × 17 columns

a.3 Handling Outliers:

In [18]:

```
def remove_outliers(df,columns,n_std):
    for col in columns:
        print('Working on coloumn: {}'.format(col))

        mean = df[col].mean()
        sd = df[col].std()

        df = df[(df[col] <= mean+(n_std*sd))]
    return df
df
```

Out[18]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx
0	10/03/2004	18.00.00	2,6	1360.0	150.0	11,9	1046.0	1
1	10/03/2004	19.00.00	2	1292.0	112.0	9,4	955.0	1
2	10/03/2004	20.00.00	2,2	1402.0	88.0	9,0	939.0	1
3	10/03/2004	21.00.00	2,2	1376.0	80.0	9,2	948.0	1
4	10/03/2004	22.00.00	1,6	1272.0	51.0	6,5	836.0	1
...	...	...	...	...	...	...	...	
9466	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9467	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9468	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9469	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
9470	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

9471 rows × 17 columns

b. Data integration

In [26]:

```
df1 = pd.read_csv('heart.csv')
df1
```

Out[26]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	t
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	

1025 rows × 14 columns

In [27]:

```
pd.concat([df,df1])
```

Out[27]:

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx
0	10/03/2004	18.00.00	2,6	1360.0	150.0	11,9	1046.0	1
1	10/03/2004	19.00.00	2	1292.0	112.0	9,4	955.0	1
2	10/03/2004	20.00.00	2,2	1402.0	88.0	9,0	939.0	1
3	10/03/2004	21.00.00	2,2	1376.0	80.0	9,2	948.0	1
4	10/03/2004	22.00.00	1,6	1272.0	51.0	6,5	836.0	1
...	...	...	...	...	...	...	...	
1020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1021	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1023	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1024	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

10496 rows × 31 columns

c. Data transformation

In [30]:

```
dt = df.groupby(['CO(GT)','PT08.S1(CO)'])
dt.first()
```

Out[30]:

		Date	Time	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)
CO(GT)	**PT08.S1(CO)**						
-200	**-200.0**	25/05/2004	19.00.00	-200.0	-200,0	-200.0	-200.0
	681.0	15/11/2004	04.00.00	-200.0	0,4	428.0	24.0
	704.0	13/10/2004	03.00.00	-200.0	0,5	444.0	-200.0
	709.0	13/10/2004	04.00.00	-200.0	0,6	449.0	-200.0
	711.0	22/08/2004	15.00.00	-200.0	2,8	640.0	-200.0
...	**...**	...	...	...	...	...	...
9,2	**1778.0**	02/11/2004	20.00.00	-200.0	48,2	1935.0	859.0
9,3	**-200.0**	14/12/2004	18.00.00	-200.0	-200,0	-200.0	1310.0
9,4	**1816.0**	02/12/2004	19.00.00	-200.0	43,9	1851.0	1184.0
9,5	**1908.0**	26/10/2004	18.00.00	-200.0	52,1	2007.0	952.0
9,9	**1881.0**	13/12/2004	18.00.00	-200.0	50,8	1983.0	1479.0

6574 rows × 15 columns

d. Error correcting

e. Data model building

In [31]:

```
from sklearn.model_selection import train_test_split
train,test=train_test_split(df,random_state=0,test_size=.25)
```

In [32]:

```
print("Training Dataset:",train.shape)
```

```
Training Dataset: (7103, 17)
```

In [33]:

```
print("Testing Dataset:",test.shape)
```

```
Testing Dataset: (2368, 17)
```

In [ ]:

```
```