Perform the following operations using Python on the Heart Disease data sets

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

In [1]:

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

In [4]:

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

In [5]:

```
df
```

Out[5]:

	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

a. Data cleaning

a.1 Removing Missing or Null Values:

In [6]:

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

a.2 Reading and Removing Duplicate Values

- Reading Duplicates:

In [7]:

```
df.duplicated(subset=['trestbps'])
```

Out[7]:

```
0       False
1       False
2       False
3       False
4       False
        ...  
1020     True
1021     True
1022     True
1023     True
1024     True
Length: 1025, dtype: bool
```

- Remove Duplicates:

In [9]:

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

Out[9]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	targe

a.3 Handling Outliers:

In [10]:

```
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[10]:

	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

b. Data integration

In [11]:

```
df1 = pd.read_csv('AirQualityUCI.csv',sep=';')
df1
```

Out[11]:

	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

In [12]:

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

Out[12]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	...	NOx(GT)
0	52.0	1.0	0.0	125.0	212.0	0.0	1.0	168.0	0.0	1.0	...	NaN
1	53.0	1.0	0.0	140.0	203.0	1.0	0.0	155.0	1.0	3.1	...	NaN
2	70.0	1.0	0.0	145.0	174.0	0.0	1.0	125.0	1.0	2.6	...	NaN
3	61.0	1.0	0.0	148.0	203.0	0.0	1.0	161.0	0.0	0.0	...	NaN
4	62.0	0.0	0.0	138.0	294.0	1.0	1.0	106.0	0.0	1.9	...	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
9466	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
9467	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
9468	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
9469	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
9470	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN

10496 rows × 31 columns

c. Data transformation

In [13]:

```
dt = df.groupby(['age','cp'])
dt.first()
```

Out[13]:

		sex	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	**cp**												
29	**1**	1	130	204	0	0	202	0	0.0	2	0	2	1
34	**1**	0	118	210	0	1	192	0	0.7	2	0	2	1
	3	1	118	182	0	0	174	0	0.0	2	0	2	1
35	**0**	1	120	198	0	1	130	1	1.6	1	0	3	0
	1	1	122	192	0	1	174	0	0.0	2	0	2	1
...	**...**	...	...	...	...	...	...	...	...	...	...	...	...
71	**1**	0	160	302	0	1	162	0	0.4	2	2	2	1
	2	0	110	265	1	0	130	0	0.0	2	1	2	1
74	**1**	0	120	269	0	0	121	1	0.2	2	1	2	1
76	**2**	0	140	197	0	2	116	0	1.1	1	0	2	1
77	**0**	1	125	304	0	0	162	1	0.0	2	3	2	0

108 rows × 12 columns

d. Error correcting

e. Data model building

In [14]:

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

In [15]:

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

```
Training Dataset: (768, 14)
```

In [16]:

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

```
Testing Dataset: (257, 14)
```

```
In [ ]:
```