PANDAS LIBRARY:

Pandas allows users to manipulate and analyze data.

Pandas provides two data structures that shape data into a readable form:

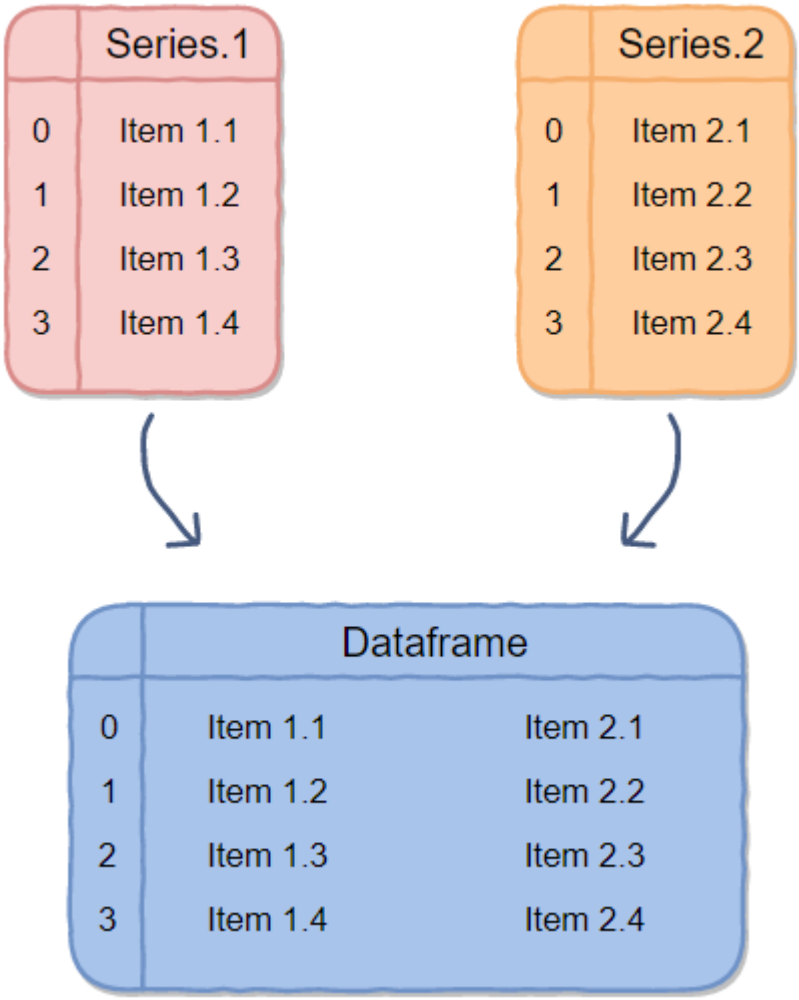
1) Series 2) Data frame

SERIES: A pandas series is a one-dimensional data structure that comprises of a key-value pair. It is similar to a python dictionary. To initialize a series, use pandas.Series():

	Series.1
0	Item 1.1
1	Item 1.2
2	Item 1.3
3	Item 1.4

	Series.2
0	Item 2.1
1	Item 2.2
2	Item 2.3
3	Item 2.4

DATAFRAME: A pandas dataframe is a two-dimensional data-structure that can be thought of as a spreadsheet. A dataframe can also be thought of as a combination of two or more series. To initialize a dataframe, use pandas.DataFrame:



In [61]:
```
#Dataframe
import pandas as pd
data = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],'Age':[28,34,29,42]}

data = pd.DataFrame(data) #creating dataframe
print(data.shape)  #gives dimensions
data
```

```
(4, 2)
```

Out[61]:

	Name	Age
0	Tom	28
1	Jack	34
2	Steve	29
3	Ricky	42

In [62]:
```
#Series
import pandas as pd
data = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],'Age':[28,34,29,42]}
data = pd.Series(data) #creating series
print(data.shape) #gives dimensions
data
```

```
(2,)
```

Out[62]:
```
Name    [Tom, Jack, Steve, Ricky]
Age              [28, 34, 29, 42]
dtype: object
```

In [63]:
```
# how to read excel or csv file

import pandas as pd
xls = pd.read_csv(r'C:\Users\sohail\Desktop\heart.csv')
xls
```

Out[63]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows × 14 columns

In [ ]:
```
# this is just for your information
#You can read multiple sheets of single excel file in this way :

import pandas as pd
xls = pd.ExcelFile(r'C:\Users\sohail\Desktop\heart.csv')
df1 = pd.read_excel(xls, 'Sheet1')
df2 = pd.read_excel(xls, 'Sheet2')
print(df1)
print(df2)
```

Working on Dataset (heart.csv)

Now we have a dataset named 'heart.csv' that gives following information of patient :

age: The person's age in years

sex: The person's sex (1 = male, 0 = female)

cp: The chest pain experienced (Value 0: typical angina, Value 1: atypical angina, Value 2: non-anginal pain, Value 3: asymptomatic)

trestbps: The person's resting blood pressure (mm Hg on admission to the hospital)

chol: The person's cholesterol measurement in mg/dl

fbs: The person's fasting blood sugar (> 120 mg/dl, 1 = true; 0 = false)

restecg: Resting electrocardiographic measurement (0 = normal, 1 = having ST-T wave abnormality, 2 = showing probable or definite left ventricular hypertrophy by Estes' criteria)

thalach: The person's maximum heart rate achieved

exang: Exercise induced angina (1 = yes; 0 = no)

oldpeak: ST depression induced by exercise relative to rest ('ST' relates to positions on the ECG plot. See more here)

slope: the slope of the peak exercise ST segment (Value 1: upsloping, Value 2: flat, Value 3: downsloping)

ca: The number of major vessels (0-3)

thal: A blood disorder called thalassemia (3 = normal; 6 = fixed defect; 7 = reversable defect)

target: Heart disease (0 = no, 1 = yes)

Diagnosis: The diagnosis of heart disease is done on a combination of clinical signs and test results. The types of tests run will be chosen on the basis of what the physician thinks is going on 1, ranging from electrocardiograms and cardiac computerized tomography (CT) scans, to blood tests and exercise stress tests 2.

Looking at information of heart disease risk factors led me to the following: high cholesterol, high blood pressure, diabetes, weight, family history and smoking 3. According to another source 4, the major factors that can't be changed are: increasing age, male gender and heredity. Note that thalassemia, one of the variables in this dataset, is heredity. Major factors that can be modified are: Smoking, high cholesterol, high blood pressure, physical inactivity, and being overweight and having diabetes. Other factors include stress, alcohol and poor diet/nutrition.

I can see no reference to the 'number of major vessels', but given that the definition of heart disease is "...what happens when your heart's blood supply is blocked or interrupted by a build-up of fatty substances in the coronary arteries", it seems logical the more major vessels is a good thing, and therefore will reduce the probability of heart disease.

Given the above, I would hypothesis that, if the model has some predictive ability, we'll see these factors standing out as most important

In [64]:
```
# csv (comma separated file) is automatically loaded as a dataframe
import pandas as pd
data = pd.read_csv(r'C:\Users\sohail\Desktop\heart.csv')


data
```

Out[64]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows × 14 columns

In [65]:
```
#it gives first five rows by default.

data.head()

# However you can type number of rows in parenthesis that you want eg. data.head(
```

Out[65]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

In [11]:
```
#it gives last five rows by default.

data.tail()

# However, you can type number of rows in parenthesis that you want eg. data.tail
```

Out[11]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

In [67]:
```
#changing name of all columns ot make it more readable
data.columns = ['age', 'sex', 'chest_pain_type', 'resting_blood_pressure', 'chole
       'exercise_induced_angina', 'st_depression', 'st_slope', 'num_major_vessels
data
```

Out[67]:

	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	rest_ecg
0	63	1	3	145	233	1	0
1	37	1	2	130	250	0	1
2	41	0	1	130	204	0	0
3	56	1	1	120	236	0	1
4	57	0	0	120	354	0	1
...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1
299	45	1	3	110	264	0	1
300	68	1	0	144	193	1	1
301	57	1	0	130	131	0	1
302	57	0	1	130	236	0	0

303 rows × 14 columns

In [68]:
```
data['sex'][data['sex'] == 0] = 'female'
data['sex'][data['sex'] == 1] = 'male'

data['chest_pain_type'][data['chest_pain_type'] == 0] = 'typical angina'
data['chest_pain_type'][data['chest_pain_type'] == 1] = 'atypical angina'
data['chest_pain_type'][data['chest_pain_type'] == 2] = 'non-anginal pain'
data['chest_pain_type'][data['chest_pain_type'] == 3] = 'asymptomatic'
data['fasting_blood_sugar'][data['fasting_blood_sugar'] == 0] = 'lower than 120mg
data['fasting_blood_sugar'][data['fasting_blood_sugar'] == 1] = 'greater than 120

data['rest_ecg'][data['rest_ecg'] == 0] = 'normal'
data['rest_ecg'][data['rest_ecg'] == 1] = 'ST-T wave abnormality'
data['rest_ecg'][data['rest_ecg'] == 2] = 'left ventricular hypertrophy'

data['exercise_induced_angina'][data['exercise_induced_angina'] == 0] = 'no'
data['exercise_induced_angina'][data['exercise_induced_angina'] == 1] = 'yes'

data['st_slope'][data['st_slope'] == 1] = 'upsloping'
data['st_slope'][data['st_slope'] == 2] = 'flat'
data['st_slope'][data['st_slope'] == 3] = 'downsloping'

data['thalassemia'][data['thalassemia'] == 1] = 'normal'
data['thalassemia'][data['thalassemia'] == 2] = 'fixed defect'
data['thalassemia'][data['thalassemia'] == 3] = 'reversable defect'
```

```
<ipython-input-68-48b7b8026ebc>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data['sex'][data['sex'] == 0] = 'female'
<ipython-input-68-48b7b8026ebc>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data['chest_pain_type'][data['chest_pain_type'] == 0] = 'typical angina'
<ipython-input-68-48b7b8026ebc>:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data['fasting_blood_sugar'][data['fasting_blood_sugar'] == 0] = 'lower than
120mg/ml'
<ipython-input-68-48b7b8026ebc>:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
```

```
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  data['fasting_blood_sugar'][data['fasting_blood_sugar'] == 1] = 'greater th
an 120mg/ml'
<ipython-input-68-48b7b8026ebc>:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  data['rest_ecg'][data['rest_ecg'] == 0] = 'normal'
<ipython-input-68-48b7b8026ebc>:15: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  data['exercise_induced_angina'][data['exercise_induced_angina'] == 0] = 'n
o'
<ipython-input-68-48b7b8026ebc>:16: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  data['exercise_induced_angina'][data['exercise_induced_angina'] == 1] = 'ye
s'
<ipython-input-68-48b7b8026ebc>:18: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  data['st_slope'][data['st_slope'] == 1] = 'upsloping'
<ipython-input-68-48b7b8026ebc>:22: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  data['thalassemia'][data['thalassemia'] == 1] = 'normal'
<ipython-input-68-48b7b8026ebc>:23: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/s
table/user_guide/indexing.html#returning-a-view-versus-a-copy (https://panda
s.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
sus-a-copy)
  data['thalassemia'][data['thalassemia'] == 2] = 'fixed defect'
<ipython-input-68-48b7b8026ebc>:24: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data['thalassemia'][data['thalassemia'] == 3] = 'reversable defect'
```

In [69]:
```
data = data.head(10)
data
```

Out[69]:

	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	rest_e
0	63	male	asymptomatic	145	233	greater than 120mg/ml	norn
1	37	male	non-anginal pain	130	250	lower than 120mg/ml	ST-T wa abnorma
2	41	female	atypical angina	130	204	lower than 120mg/ml	norn
3	56	male	atypical angina	120	236	lower than 120mg/ml	ST-T wa abnorma
4	57	female	typical angina	120	354	lower than 120mg/ml	ST-T wa abnorma
5	57	male	typical angina	140	192	lower than 120mg/ml	ST-T wa abnorma
6	56	female	atypical angina	140	294	lower than 120mg/ml	norn
7	44	male	atypical angina	120	263	lower than 120mg/ml	ST-T wa abnorma
8	52	male	non-anginal pain	172	199	greater than 120mg/ml	ST-T wa abnorma
9	57	male	non-anginal pain	150	168	lower than 120mg/ml	ST-T wa abnorma

In [71]:
```
#changing name of specific columns
data=data.rename(columns = {'rest_ecg':'ecg','sex':'gender'})
data.head()
```

Out[71]:

	age	gender	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	e
0	63	male	asymptomatic	145	233	greater than 120mg/ml	nor
1	37	male	non-anginal pain	130	250	lower than 120mg/ml	ST-T wa abnorma
2	41	female	atypical angina	130	204	lower than 120mg/ml	nor
3	56	male	atypical angina	120	236	lower than 120mg/ml	ST-T wa abnorma
4	57	female	typical angina	120	354	lower than 120mg/ml	ST-T wa abnorma

In [80]:
```
#Reloading original file
import pandas as pd
data = pd.read_csv(r'C:\Users\sohail\Desktop\heart.csv')
#it gives you lot of information about

data.describe()
```

Out[80]:

	age	sex	cp	trestbps	chol	fbs	restecg	tha
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.00
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.64
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.90
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.00
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.50
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.00
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.00
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.00

In [76]:
```
#Inserting columns
data = data.head(10)
names = ['Ali', 'Salman', 'Sohail', 'Mohsin','Waqas','Zeshan','Babar','John','Elo
data.insert(0,'name',names)
data
```

Out[76]:

	name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	Ali	63	1	3	145	233	1	0	150	0	2.3	0	0	1
1	Salman	37	1	2	130	250	0	1	187	0	3.5	0	0	2
2	Sohail	41	0	1	130	204	0	0	172	0	1.4	2	0	2
3	Mohsin	56	1	1	120	236	0	1	178	0	0.8	2	0	2
4	Waqas	57	0	0	120	354	0	1	163	1	0.6	2	0	2
5	Zeshan	57	1	0	140	192	0	1	148	0	0.4	1	0	1
6	Babar	56	0	1	140	294	0	0	153	0	1.3	1	0	2
7	John	44	1	1	120	263	0	1	173	0	0.0	2	0	3
8	Elon	52	1	2	172	199	1	1	162	0	0.5	2	0	3
9	Michael	57	1	2	150	168	0	1	174	0	1.6	2	0	2

In [83]:
```
#Make new dataframe of first 5 rows
data = data.head()

#Then select specific columns
data = data[['age','sex','cp','target']]
data
```

Out[83]:

	age	sex	cp	target
0	63	1	3	1
1	37	1	2	1
2	41	0	1	1
3	56	1	1	1
4	57	0	0	1

In [84]:
```
# inplace=True changes the original dataframe while inplace=False makes the copy

#dropping column
data.drop(labels=['target'],axis=1,inplace=True)
data
```

Out[84]:

	age	sex	cp
0	63	1	3
1	37	1	2
2	41	0	1
3	56	1	1
4	57	0	0

In [85]:
```
#Adding row
new_row = {'age':25, 'sex':1, 'cp':3}
data = data.append(new_row, ignore_index=True) #ignore_index by-default=False. If
data
```

Out[85]:

	age	sex	cp
0	63	1	3
1	37	1	2
2	41	0	1
3	56	1	1
4	57	0	0
5	25	1	3

In [86]:
```
#Adding row at a specific index

data.loc[7] = [23,1,4]
data
```

Out[86]:

	age	sex	cp
0	63	1	3
1	37	1	2
2	41	0	1
3	56	1	1
4	57	0	0
5	25	1	3
7	23	1	4

In [87]:
```
#Dropping rows

data.drop(labels=[5,7],axis=0,inplace=True)
data
```

Out[87]:

	age	sex	cp
0	63	1	3
1	37	1	2
2	41	0	1
3	56	1	1
4	57	0	0

In [88]:
```
#Resetting index

data.reset_index(drop=True,inplace=True)
data

#If you set drop = False , reset_index will create another column of indexes
```

Out[88]:

	age	sex	cp
0	63	1	3
1	37	1	2
2	41	0	1
3	56	1	1
4	57	0	0

In [89]:
```
#loading original file
data = pd.read_csv(r'C:\Users\sohail\Desktop\heart.csv')
data
```

Out[89]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows × 14 columns

In [90]:
```
#loading csv file to concatenate its rows with our dataframe
data1 = pd.read_csv(r'C:\Users\sohail\Desktop\heart2.csv')
data1
```

Out[90]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
1	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
2	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
3	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

In [91]:
```
#concatenating rows of two csv files
result = pd.concat([data, data1],ignore_index =True)
result
```

Out[91]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0
303	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
304	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
305	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
306	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

307 rows × 14 columns

In [92]:
```
#loading csv file to concatenate its columns with our dataframe
data2 = pd.read_csv(r'C:\Users\sohail\Desktop\heart3.csv')
data2
```

Out[92]:

	Name
0	Ali
1	Ali
2	Ali
3	Ali
4	Ali
...	...
302	Ali
303	Ali
304	Ali
305	Ali
306	Ali

307 rows × 1 columns

In [93]:
```
#concatenating columns of two csv files
result3 = pd.concat([result, data2],axis=1,ignore_index = False)
result3
```

Out[93]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0
303	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
304	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
305	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
306	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

307 rows × 15 columns

In [103]:
```
# Selection of rows and columns and particular value
import pandas as pd
data = pd.read_csv(r'C:\Users\sohail\Desktop\heart.csv')
data = data.head(10)

data
```

Out[103]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
7	44	1	1	120	263	0	1	173	0	0.0	2	0	3	1
8	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
9	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1

In [95]:
```
#lets start Extraction of particular columns, rows and values
data[['age']]
```

Out[95]:

	age
0	63
1	37
2	41
3	56
4	57
5	57
6	56
7	44
8	52
9	57

In [96]:
```
data[['age','sex']]
```

Out[96]:

	age	sex
0	63	1
1	37	1
2	41	0
3	56	1
4	57	0
5	57	1
6	56	0
7	44	1
8	52	1
9	57	1

In [104]:
```
data.head(10)
```

Out[104]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
7	44	1	1	120	263	0	1	173	0	0.0	2	0	3	1
8	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
9	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1

In [100]:
```
# check each command one by one by un-commenting
# iloc access rows and columns by indexes

# data.iloc[starting index of row:ending index of row(exclusive):step size , star

#data.iloc[2:5 , 2:4]        #iloc[starting row:ending row , starting column:endi

#data.iloc[0:4 , 5:6]

#data.iloc[2: , :5]

#data.iloc[: , :]

#data.iloc[0:5:2 , 2:4:2]      #iloc[starting row:ending row:stepSize , starting

data.iloc[::-1 , ::-1]
#data.head(10)
```

Out[100]:

	target	thal	ca	slope	oldpeak	exang	thalach	restecg	fbs	chol	trestbps	cp	sex	age
9	1	2	0	2	1.6	0	174	1	0	168	150	2	1	57
8	1	3	0	2	0.5	0	162	1	1	199	172	2	1	52
7	1	3	0	2	0.0	0	173	1	0	263	120	1	1	44
6	1	2	0	1	1.3	0	153	0	0	294	140	1	0	56
5	1	1	0	1	0.4	0	148	1	0	192	140	0	1	57
4	1	2	0	2	0.6	1	163	1	0	354	120	0	0	57
3	1	2	0	2	0.8	0	178	1	0	236	120	1	1	56
2	1	2	0	2	1.4	0	172	0	0	204	130	1	0	41
1	1	2	0	0	3.5	0	187	1	0	250	130	2	1	37
0	1	1	0	0	2.3	0	150	0	1	233	145	3	1	63

In [101]:
```
#data
data.iloc[0][0]
#data.iloc[2][3]
```

Out[101]:
```
63.0
```

In [105]:
```
# "loc" is designed to access values by labels(categorical values). So first we w

names = ['Ali', 'Salman', 'Sameer', 'Ozair','Omar','Zeshan','Babar','John','Elon'
data.insert(0,'name',names)
data
```

Out[105]:

	name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	Ali	63	1	3	145	233	1	0	150	0	2.3	0	0	1
1	Salman	37	1	2	130	250	0	1	187	0	3.5	0	0	2
2	Sameer	41	0	1	130	204	0	0	172	0	1.4	2	0	2
3	Ozair	56	1	1	120	236	0	1	178	0	0.8	2	0	2
4	Omar	57	0	0	120	354	0	1	163	1	0.6	2	0	2
5	Zeshan	57	1	0	140	192	0	1	148	0	0.4	1	0	1
6	Babar	56	0	1	140	294	0	0	153	0	1.3	1	0	2
7	John	44	1	1	120	263	0	1	173	0	0.0	2	0	3
8	Elon	52	1	2	172	199	1	1	162	0	0.5	2	0	3
9	Michael	57	1	2	150	168	0	1	174	0	1.6	2	0	2

In [106]:
```
data = data.set_index('name')
data
```

Out[106]:

name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	tar
Ali	63	1	3	145	233	1	0	150	0	2.3	0	0	1	
Salman	37	1	2	130	250	0	1	187	0	3.5	0	0	2	
Sameer	41	0	1	130	204	0	0	172	0	1.4	2	0	2	
Ozair	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
Omar	57	0	0	120	354	0	1	163	1	0.6	2	0	2	
Zeshan	57	1	0	140	192	0	1	148	0	0.4	1	0	1	
Babar	56	0	1	140	294	0	0	153	0	1.3	1	0	2	
John	44	1	1	120	263	0	1	173	0	0.0	2	0	3	
Elon	52	1	2	172	199	1	1	162	0	0.5	2	0	3	
Michael	57	1	2	150	168	0	1	174	0	1.6	2	0	2	

In [107]:
```
## check each command one by one by un-commenting
#USE LABELS FOR LOC
print(data.loc['Salman']['trestbps'])
#print(data.loc['Sameer':'Babar','trestbps'])
```

```
130.0
```

In [48]:
```
#will print only rows that have age values greater than 50
data.loc[data['age']>50]
```

Out[48]:

name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	tar
Ali	63	1	3	145	233	1	0	150	0	2.3	0	0	1	
Ozair	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
Omar	57	0	0	120	354	0	1	163	1	0.6	2	0	2	
Zeshan	57	1	0	140	192	0	1	148	0	0.4	1	0	1	
Babar	56	0	1	140	294	0	0	153	0	1.3	1	0	2	
Elon	52	1	2	172	199	1	1	162	0	0.5	2	0	3	
Michael	57	1	2	150	168	0	1	174	0	1.6	2	0	2	

In [108]:
```
data.sort_values('age',inplace=True)
data
```

Out[108]:

name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	tar
Salman	37	1	2	130	250	0	1	187	0	3.5	0	0	2	
Sameer	41	0	1	130	204	0	0	172	0	1.4	2	0	2	
John	44	1	1	120	263	0	1	173	0	0.0	2	0	3	
Elon	52	1	2	172	199	1	1	162	0	0.5	2	0	3	
Ozair	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
Babar	56	0	1	140	294	0	0	153	0	1.3	1	0	2	
Omar	57	0	0	120	354	0	1	163	1	0.6	2	0	2	
Zeshan	57	1	0	140	192	0	1	148	0	0.4	1	0	1	
Michael	57	1	2	150	168	0	1	174	0	1.6	2	0	2	
Ali	63	1	3	145	233	1	0	150	0	2.3	0	0	1	

In [109]:
```
data.at['Elon','chol']
```

Out[109]: 199

In [110]:
```
#changing value at particular position
data.at['Elon','chol'] = 120
data
```

Out[110]:

name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	tar
Salman	37	1	2	130	250	0	1	187	0	3.5	0	0	2	
Sameer	41	0	1	130	204	0	0	172	0	1.4	2	0	2	
John	44	1	1	120	263	0	1	173	0	0.0	2	0	3	
Elon	52	1	2	172	120	1	1	162	0	0.5	2	0	3	
Ozair	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
Babar	56	0	1	140	294	0	0	153	0	1.3	1	0	2	
Omar	57	0	0	120	354	0	1	163	1	0.6	2	0	2	
Zeshan	57	1	0	140	192	0	1	148	0	0.4	1	0	1	
Michael	57	1	2	150	168	0	1	174	0	1.6	2	0	2	
Ali	63	1	3	145	233	1	0	150	0	2.3	0	0	1	

In [111]:
```
#making derived columns
data['new_column'] = data['restecg']+data['thalach']
data
```

Out[111]:

name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	tar
Salman	37	1	2	130	250	0	1	187	0	3.5	0	0	2	
Sameer	41	0	1	130	204	0	0	172	0	1.4	2	0	2	
John	44	1	1	120	263	0	1	173	0	0.0	2	0	3	
Elon	52	1	2	172	120	1	1	162	0	0.5	2	0	3	
Ozair	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
Babar	56	0	1	140	294	0	0	153	0	1.3	1	0	2	
Omar	57	0	0	120	354	0	1	163	1	0.6	2	0	2	
Zeshan	57	1	0	140	192	0	1	148	0	0.4	1	0	1	
Michael	57	1	2	150	168	0	1	174	0	1.6	2	0	2	
Ali	63	1	3	145	233	1	0	150	0	2.3	0	0	1	

Cleaning

In [112]:
```
data = pd.read_csv(r'C:\Users\sohail\Desktop\clean.csv')
data
```

Out[112]:

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sug
0	12/29/2019 .	63	male	non-anginal pain	145	233.0	
1	11/18/2019 - 0,124	37	male	atypical angina	130	250.0	
2	12/05/2019 - 25	41	fe---	typical angina	130	204.0	
3	01/01/2019 -- 0?126	56	male	typical angina	120	236.0	
4	04/09/2019 - 0,,127	?	fe---	typical angina	135	NaN	
5	11/02/2019 99s	57	male	typical angina	142	192.0	
6	12/12/2019 - 0	56	fe---	typical angina	140	294.0	
7	12/12/2019 - 01	/	male	typical angina	120	NaN	
8	12/12/2019,,?	52	male	atypical angina	172	199.0	
9	12/12/2019..	57	male	atypical angina	150	168.0	

In [113]:
```
#drop columns having all values as null values
data = data.dropna(axis=1,how='all')
data
```

Out[113]:

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sug
0	12/29/2019 .	63	male	non-anginal pain	145	233.0	
1	11/18/2019 - 0,124	37	male	atypical angina	130	250.0	
2	12/05/2019 - 25	41	fe---	typical angina	130	204.0	
3	01/01/2019 -- 0?126	56	male	typical angina	120	236.0	
4	04/09/2019 - 0,,127	?	fe---	typical angina	135	NaN	
5	11/02/2019 99s	57	male	typical angina	142	192.0	
6	12/12/2019 - 0	56	fe---	typical angina	140	294.0	
7	12/12/2019 - 01	/	male	typical angina	120	NaN	
8	12/12/2019,,?	52	male	atypical angina	172	199.0	
9	12/12/2019..	57	male	atypical angina	150	168.0	

In [114]:
```
#replacing male and female with 0 and 1
data['sex'].replace({'fe---' : 0, 'male' : 1}, inplace = True)
data
```

```
C:\Users\sohail\anaconda3\lib\site-packages\pandas\core\series.py:4563: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  return super().replace(
```

Out[114]:

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sug
0	12/29/2019 .	63	1	non-anginal pain	145	233.0	
1	11/18/2019 - 0,124	37	1	atypical angina	130	250.0	
2	12/05/2019 - 25	41	0	typical angina	130	204.0	
3	01/01/2019 -- 0?126	56	1	typical angina	120	236.0	
4	04/09/2019 - 0,,127	?	0	typical angina	135	NaN	
5	11/02/2019 99s	57	1	typical angina	142	192.0	
6	12/12/2019 - 0	56	0	typical angina	140	294.0	
7	12/12/2019 - 01	/	1	typical angina	120	NaN	
8	12/12/2019,,?	52	1	atypical angina	172	199.0	
9	12/12/2019..	57	1	atypical angina	150	168.0	

In [56]:
```
#In exercise induced angina, replacing yes and no with 1 and 0
data['exercise_induced_angina'].replace({'no' : 0, 'yes' : 1}, inplace = True)
data
```

Out[56]:

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_suga
0	12/29/2019 .	63	1	non-anginal pain	145	233.0	
1	11/18/2019 - 0,124	37	1	atypical angina	130	250.0	
2	12/05/2019 - 25	41	0	typical angina	130	204.0	
3	01/01/2019 -- 0?126	56	1	typical angina	120	236.0	
4	04/09/2019 - 0,,127	?	0	typical angina	135	NaN	
5	11/02/2019 99s	57	1	typical angina	142	192.0	
6	12/12/2019 - 0	56	0	typical angina	140	294.0	
7	12/12/2019 - 01	/	1	typical angina	120	NaN	
8	12/12/2019,,?	52	1	atypical angina	172	199.0	
9	12/12/2019..	57	1	atypical angina	150	168.0	

In [57]:
```
#fill in all missing values with mean of their column values
data = data.fillna(data.mean())
data
```

Out[57]:

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_suga
0	12/29/2019 .	63	1	non-anginal pain	145	233.0	
1	11/18/2019 - 0,124	37	1	atypical angina	130	250.0	
2	12/05/2019 - 25	41	0	typical angina	130	204.0	
3	01/01/2019 -- 0?126	56	1	typical angina	120	236.0	
4	04/09/2019 - 0,,127	?	0	typical angina	135	222.0	
5	11/02/2019 99s	57	1	typical angina	142	192.0	
6	12/12/2019 - 0	56	0	typical angina	140	294.0	
7	12/12/2019 - 01	/	1	typical angina	120	222.0	
8	12/12/2019,,?	52	1	atypical angina	172	199.0	
9	12/12/2019..	57	1	atypical angina	150	168.0	

In [58]:
```
data['chest_pain_type'].replace({'non-anginal pain' : 1, 'typical angina' : 2, 'a
data
```

Out[58]:

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sug
0	12/29/2019 .	63	1	1	145	233.0	
1	11/18/2019 - 0,124	37	1	3	130	250.0	
2	12/05/2019 - 25	41	0	2	130	204.0	
3	01/01/2019 -- 0?126	56	1	2	120	236.0	
4	04/09/2019 - 0,,127	?	0	2	135	222.0	
5	11/02/2019 99s	57	1	2	142	192.0	
6	12/12/2019 - 0	56	0	2	140	294.0	
7	12/12/2019 - 01	/	1	2	120	222.0	
8	12/12/2019,,?	52	1	3	172	199.0	
9	12/12/2019..	57	1	3	150	168.0	

In [ ]:
```
#converting impossible values to 0
data['age'].replace(['?','/'],0,inplace=True)

#converting data type of column to int64 so that mean could be taken
data['age']=data.age.astype('int64')

#replacing 0 with mean of column
data['age'].replace({0 : round(data['age'].mean())}, inplace = True)

data
```

In [59]:
```
#converting cholesterol column to integer type
data['cholesterol']=data.cholesterol.astype('int64')
data
```

Out[59]:

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_suga
0	12/29/2019 .	63	1	1	145	233	
1	11/18/2019 - 0,124	37	1	3	130	250	
2	12/05/2019 - 25	41	0	2	130	204	
3	01/01/2019 -- 0?126	56	1	2	120	236	
4	04/09/2019 - 0,,127	?	0	2	135	222	
5	11/02/2019 99s	57	1	2	142	192	
6	12/12/2019 - 0	56	0	2	140	294	
7	12/12/2019 - 01	/	1	2	120	222	
8	12/12/2019,,?	52	1	3	172	199	
9	12/12/2019..	57	1	3	150	168	

In [60]:
```
#get the first 10 characters of date
data['Date'] = data['Date'].str[:10]
data
```

Out[60]:

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar
0	12/29/2019	63	1	1	145	233	1
1	11/18/2019	37	1	3	130	250	0
2	12/05/2019	41	0	2	130	204	0
3	01/01/2019	56	1	2	120	236	0
4	04/09/2019	?	0	2	135	222	0
5	11/02/2019	57	1	2	142	192	0
6	12/12/2019	56	0	2	140	294	0
7	12/12/2019	/	1	2	120	222	0
8	12/12/2019	52	1	3	172	199	1
9	12/12/2019	57	1	3	150	168	0

In [ ]:
```
data.to_csv (r'Desktop\Final_Data.csv', index = False, header=True) # index=true
```

In [ ]:
```

```