PYTHON INTERNSHIP MANUAL 5(A)

Batch A - 2025 | Week#9 (Introduction to Pamdas)

Important Instructions:

- Read the manual carefully and understand every topic before starting the assignment.
- Follow the deadlines properly; marks will be deducted for late submissions.
- Avoid to use AI Tools and make your own logic.
- Completing all tasks in the assignment is compulsory to receive full marks.

- **Manual Posted Date**:

  30th April, 2025 (Wednesday)

- **Assignment-5 Deadline:**

  5th April, 2025 (Monday 11:59pm)

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

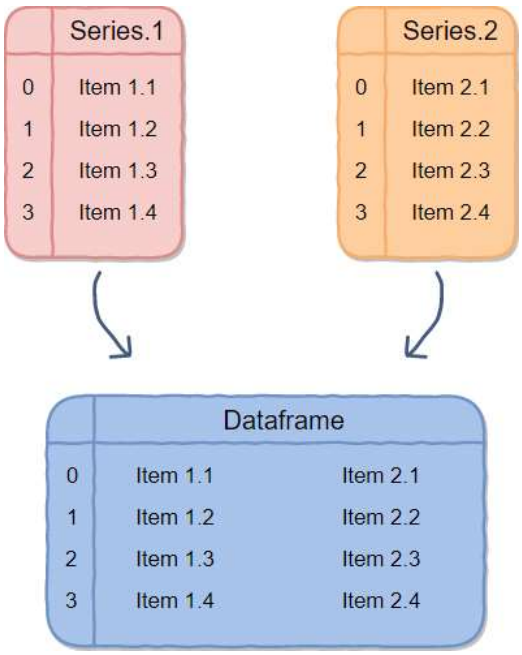
A pandas series is a one-dimensional data structure that comprises of a key-value pair. It is similar to a python dictionary. To initialize a series, use pandas.Series():

	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

Data Frame

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:




```
#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)
```

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

```
#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,)

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

```
#You can read multiple sheets of excel file in this way :

import pandas as pd
xls = pd.ExcelFile(r'myFile.xlsx')
df1 = pd.read_excel(xls, 'Sheet1')
df2 = pd.read_excel(xls, 'Sheet2')
print(df1)
print(df2)
```

```
     Name  Marks  Age
0     Ali     20   25
1   Sadiq     30   78
2  Hammad     50   45
3  Romaan     70   69
  Animals   Country
0     Cow  Pakistan
1    Goat     India
```

Example:

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 1: typical angina, Value 2: atypical angina, Value 3: non-anginal pain, Value 4: 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

```
# csv (comma separated file) is automatically loaded as a dataframe
import pandas as pd
data = pd.read_csv(r'heart.csv')


data
```

	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

```
#it gives first five rows by default.

data.head()

# However you can type number of rows in parenthesis that you want eg. data.head(10) will pritn first 10 rows
```

	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

```
#it gives last five rows by default.

data.tail()

# However, you can type number of rows in parenthesis that you want eg. data.tail(10) will print last 10 rows
```

	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

```
#changing name of all columns ot make it more readable
data.columns = ['age', 'sex', 'chest_pain_type', 'resting_blood_pressure', 'cholesterol', 'fasting_blood_sugar', 'rest_ecg', 'ma
       'exercise_induced_angina', 'st_depression', 'st_slope', 'num_major_vessels', 'thalassemia', 'target']
data
```

	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	rest_ecg	max_heart_rate_achieved	exercise_induced_angina	st_depr
0	63	1	3	145	233	1	0	150	0	
1	37	1	2	130	250	0	1	187	0	
2	41	0	1	130	204	0	0	172	0	
3	56	1	1	120	236	0	1	178	0	
4	57	0	0	120	354	0	1	163	1	
...	...	...	...	...	...	...	...	...	...	
298	57	0	0	140	241	0	1	123	1	
299	45	1	3	110	264	0	1	132	0	
300	68	1	0	144	193	1	1	141	0	
301	57	1	0	130	131	0	1	115	1	
302	57	0	1	130	236	0	0	174	0	

303 rows × 14 columns

```
data['sex'][data['sex'] == 0] = 'female'
data['sex'][data['sex'] == 1] = 'male'

data['chest_pain_type'][data['chest_pain_type'] == 1] = 'typical angina'
data['chest_pain_type'][data['chest_pain_type'] == 2] = 'atypical angina'
data['chest_pain_type'][data['chest_pain_type'] == 3] = 'non-anginal pain'
data['chest_pain_type'][data['chest_pain_type'] == 4] = 'asymptomatic'
data['fasting_blood_sugar'][data['fasting_blood_sugar'] == 0] = 'lower than 120mg/ml'
data['fasting_blood_sugar'][data['fasting_blood_sugar'] == 1] = 'greater than 120mg/ml'

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'
```

```
data = data.head(10)
data
```

	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	rest_ecg	max_heart_rate_achieved	exercise_induced_angina	st_de
0	63	male	non-anginal pain	145	233	greater than 120mg/ml	normal	150	no	
1	37	male	atypical angina	130	250	lower than 120mg/ml	ST-T wave abnormality	187	no	
2	41	female	typical angina	130	204	lower than 120mg/ml	normal	172	no	
3	56	male	typical angina	120	236	lower than 120mg/ml	ST-T wave abnormality	178	no	
4	57	female	0	120	354	lower than 120mg/ml	ST-T wave abnormality	163	yes	
5	57	male	0	140	192	lower than 120mg/ml	ST-T wave abnormality	148	no	
6	56	female	typical angina	140	294	lower than 120mg/ml	normal	153	no	
7	44	male	typical angina	120	263	lower than 120mg/ml	ST-T wave abnormality	173	no	
8	52	male	atypical angina	172	199	greater than 120mg/ml	ST-T wave abnormality	162	no	
9	57	male	atypical angina	150	168	lower than 120mg/ml	ST-T wave abnormality	174	no	

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

	age	gender	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	ecg	max_hea
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	

```
#Reloading original file
import pandas as pd
data = pd.read_csv(r'heart.csv')
#it gives you lot of information about

data.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0(
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	2.31
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	0.61
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.0(
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	2.0(
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	2.0(
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.0(
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.0(

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

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

```
#Make new dataframe of first 5 rows
data = data.head()

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

	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

```
# inplace=True changes the original dataframe while inplace=False makes the copy

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

	age	sex	cp
0	63	1	3
1	37	1	2
2	41	0	1

```
#Adding row
new_row = {'names':"Bill",'age':25, 'sex':1, 'cp':3}
data = data.append(new_row, ignore_index=True)
#ignore_index by-default=False.
#If True, index values are not used along concatenation axis and the resulting axis will be labeled 0,1,..,n-1.
#Cannot append dictionary if ignore_index=False
data
```

	names	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	Ali	63.0	1.0	3.0	145.0	233.0	1.0	0.0	150.0	0.0	2.3	0.0	0.0	1.0
1	Salman	37.0	1.0	2.0	130.0	250.0	0.0	1.0	187.0	0.0	3.5	0.0	0.0	2.0
2	Sohail	41.0	0.0	1.0	130.0	204.0	0.0	0.0	172.0	0.0	1.4	2.0	0.0	2.0
3	Mohsin	56.0	1.0	1.0	120.0	236.0	0.0	1.0	178.0	0.0	0.8	2.0	0.0	2.0
4	Waqas	57.0	0.0	0.0	120.0	354.0	0.0	1.0	163.0	1.0	0.6	2.0	0.0	2.0
5	NaN	25.0	1.0	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	Bill	25.0	1.0	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
#Adding row at a specific index

data.loc[7] = ["Ali",23,1,4,145.0,235.0,1.0,1.0,152.25,0.0,2.3,0.0,0.0,1.0]
data
```

	names	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	Ali	63.0	1.0	3.0	145.0	233.0	1.0	0.0	150.00	0.0	2.3	0.0	0.0	1.0
1	Salman	37.0	1.0	2.0	130.0	250.0	0.0	1.0	187.00	0.0	3.5	0.0	0.0	2.0
2	Sohail	41.0	0.0	1.0	130.0	204.0	0.0	0.0	172.00	0.0	1.4	2.0	0.0	2.0
3	Mohsin	56.0	1.0	1.0	120.0	236.0	0.0	1.0	178.00	0.0	0.8	2.0	0.0	2.0
4	Waqas	57.0	0.0	0.0	120.0	354.0	0.0	1.0	163.00	1.0	0.6	2.0	0.0	2.0
5	NaN	25.0	1.0	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
6	Bill	25.0	1.0	3.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
7	Ali	23.0	1.0	4.0	145.0	235.0	1.0	1.0	152.25	0.0	2.3	0.0	0.0	1.0

```
#dropping rows

import pandas as pd

data = {
  "name": ["Sally", "Mary", "John"],
  "age": [50, 40, 30],
  "qualified": [True, False, False]
}

df = pd.DataFrame(data)

df
```

	name	age	qualified
0	Sally	50	True
1	Mary	40	False
2	John	30	False

```
newdf = df.drop(labels=[0,2], axis='rows')

print(newdf)
```

```
   name  age  qualified
1  Mary   40      False
```

```
#dropping columns

newdf = df.drop("qualified", axis='columns')

print(newdf)
```

```
    name  age
0  Sally   50
1   Mary   40
2   John   30
```

```
#Resetting index

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

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

	index	age	sex	cp
0	0	63	1	3
1	1	37	1	2

Concatenating data frames

```
#loading original file
data = pd.read_csv(r'heart.csv')
data
```

	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

```
#loading another csv file to concatenate its rows with our dataframe
data1 = pd.read_csv(r'heart2.csv')
data1
```

	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

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

	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

```
#loading csv file to concatenate its columns with our dataframe
data2 = pd.read_csv(r'heart3.csv')
data2
```

	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

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

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

307 rows × 15 columns

Selection of rows and columns and particular value

```
# Selection of rows and columns and particular value
import pandas as pd
data = pd.read_csv(r'heart.csv')
data = data.head(10)

data
```

	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

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

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

```
data[['age','sex']]
```

	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

```
# iloc access rows and columns by indexes

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

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

#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 column:ending column:stepSize]

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

```
data.iloc[1][3]
```

```
130.0
```

```
#loc is designed to access values through labels

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

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

```
data = data.set_index('name')
data
```

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

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

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

Sorting Data

```
data.sort_values('age',inplace=True)
data
```

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

Changing a value at a particular position

```
data.at['Elon','chol']
```

```
199
```

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

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

Making derived columns

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

name	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	new_column
Salman	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1	188
Sameer	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1	172
John	44	1	1	120	263	0	1	173	0	0.0	2	0	3	1	174
Elon	52	1	2	172	120	1	1	162	0	0.5	2	0	3	1	163
Ozair	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1	179
Babar	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1	153
Omar	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1	164
Zeshan	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1	149
Michael	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1	175
Ali	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1	150

Cleaning Data

```
data = pd.read_csv(r'clean.csv')
data
```

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	rest_ecg	max_heart_rate_achieved	exercise_induced_ang
0	12/29/2019 .	63	male	non-anginal pain	145	233.0	1	NaN	150	
1	11/18/2019 - 0,124	37	male	atypical angina	130	250.0	0	NaN	187	
2	12/05/2019 - 25	41	fe---	typical angina	130	204.0	0	NaN	172	
3	01/01/2019 -- 0?126	56	male	typical angina	120	236.0	0	NaN	165	
4	04/09/2019 - 0,,127	?	fe---	typical angina	135	NaN	0	NaN	155	
5	11/02/2019 99s	57	male	typical angina	142	192.0	0	NaN	125	
6	12/12/2019 - 0	56	fe---	typical angina	140	294.0	0	NaN	145	
7	12/12/2019 - 01	/	male	typical angina	120	NaN	0	NaN	168	
8	12/12/2019,,?	52	male	atypical angina	172	199.0	1	NaN	153	
9	12/12/2019..	57	male	atypical angina	150	168.0	0	NaN	173	

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

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	max_heart_rate_achieved	exercise_induced_angina	st_de
0	12/29/2019 .	63	male	non-anginal pain	145	233.0	1	150	no	
1	11/18/2019 - 0.124	37	male	atypical angina	130	250.0	0	187	no	
2	12/05/2019 - 25	41	fe---	typical angina	130	204.0	0	172	no	
3	01/01/2019 -- 07126	56	male	typical angina	120	236.0	0	165	no	
4	04/09/2019 - 0_127	?	fe---	typical angina	135	NaN	0	155	yes	
5	11/02/2019 99s	57	male	typical angina	142	192.0	0	125	no	
6	12/12/2019 - 0	56	fe---	typical angina	140	294.0	0	145	no	
7	12/12/2019 - 01	/	male	typical angina	120	NaN	0	168	no	
8	12/12/2019_?	52	male	atypical angina	172	199.0	1	153	no	
9	12/12/2019.	57	male	atypical angina	150	168.0	0	173	no	

Parameter	Value	Description
axis	0 1 'index' 'columns'	Optional, default 0. 0 and 'index'removes ROWS that contains NULL values 1 and 'columns' removes COLUMNS that contains NULL values
how	'all' 'any'	Optional, default 'any'. Specifies whether to remove the row or column when ALL values are NULL, or if ANY value is NULL.

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

```
C:\Users\Fast\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
  return super().replace(
```

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	max_heart_rate_achieved	exercise_induced_angina	st_dep
0	12/29/2019 .	63	1	non-anginal pain	145	233.0	1	150	no	
1	11/18/2019 - 0.124	37	1	atypical angina	130	250.0	0	187	no	
2	12/05/2019 - 25	41	0	typical angina	130	204.0	0	172	no	
3	01/01/2019 -- 07126	56	1	typical angina	120	236.0	0	165	no	
4	04/09/2019 - 0_127	?	0	typical angina	135	NaN	0	155	yes	
5	11/02/2019 99s	57	1	typical angina	142	192.0	0	125	no	
6	12/12/2019 - 0	56	0	typical angina	140	294.0	0	145	no	
7	12/12/2019 - 01	/	1	typical angina	120	NaN	0	168	no	
8	12/12/2019_?	52	1	atypical angina	172	199.0	1	153	no	
9	12/12/2019.	57	1	atypical angina	150	168.0	0	173	no	

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

```
C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py:6610: 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
  return self._update_inplace(result)
```

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	max_heart_rate_achieved	exercise_induced_angina	st_dep
0	12/29/2019	63	1	non-anginal pain	145	233.0	1	150	0	
1	11/18/2019 - 0.124	37	1	atypical angina	130	250.0	0	187	0	
2	12/05/2019 - 25	41	0	typical angina	130	204.0	0	172	0	
3	01/01/2019 – 07126	56	1	typical angina	120	236.0	0	165	0	
4	04/09/2019 - 0.,127	?	0	typical angina	135	NaN	0	155	1	
5	11/02/2019 99s	57	1	typical angina	142	192.0	0	125	0	
6	12/12/2019 - 0	56	0	typical angina	140	294.0	0	145	0	
7	12/12/2019 - 01	/	1	typical angina	120	NaN	0	168	0	
8	12/12/2019..?	52	1	atypical angina	172	199.0	1	153	0	
9	12/12/2019	57	1	atypical angina	150	168.0	0	173	0	

```
#fill in missing values with mean of their column values
data = data.fillna(data.mean())
data
```

```
<ipython-input-152-640270f96ac8>:2: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  data = data.fillna(data.mean())
```

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	max_heart_rate_achieved	exercise_induced_angina	st_dep
0	12/29/2019	63	1	non-anginal pain	145	233.0	1	150	0	
1	11/18/2019 - 0.124	37	1	atypical angina	130	250.0	0	187	0	
2	12/05/2019 - 25	41	0	typical angina	130	204.0	0	172	0	
3	01/01/2019 – 07126	56	1	typical angina	120	236.0	0	165	0	
4	04/09/2019 - 0.,127	?	0	typical angina	135	222.0	0	155	1	
5	11/02/2019 99s	57	1	typical angina	142	192.0	0	125	0	
6	12/12/2019 - 0	56	0	typical angina	140	294.0	0	145	0	
7	12/12/2019 - 01	/	1	typical angina	120	222.0	0	168	0	
8	12/12/2019..?	52	1	atypical angina	172	199.0	1	153	0	
9	12/12/2019	57	1	atypical angina	150	168.0	0	173	0	

```
#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
```

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	max_heart_rate_achieved	exercise_induced_angina	st_dep
0	12/29/2019	63	1	1	145	233.0	1	150	0	
1	11/18/2019 - 0,124	37	1	3	130	250.0	0	187	0	
2	12/05/2019 - 25	41	0	2	130	204.0	0	172	0	
3	01/01/2019 – 0?126	56	1	2	120	236.0	0	165	0	
4	04/09/2019 - 0,,127	42	0	2	135	222.0	0	155	1	
5	11/02/2019 99s	57	1	2	142	192.0	0	125	0	
6	12/12/2019 - 0	56	0	2	140	294.0	0	145	0	
7	12/12/2019 - 01	42	1	2	120	222.0	0	168	0	
8	12/12/2019,,?	52	1	3	172	199.0	1	153	0	
9	12/12/2019	57	1	3	150	168.0	0	173	0	

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

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	max_heart_rate_achieved	exercise_induced_angina	st_dep
0	12/29/2019	63	1	1	145	233	1	150	0	
1	11/18/2019 - 0,124	37	1	3	130	250	0	187	0	
2	12/05/2019 - 25	41	0	2	130	204	0	172	0	
3	01/01/2019 – 0?126	56	1	2	120	236	0	165	0	
4	04/09/2019 - 0,,127	42	0	2	135	222	0	155	1	
5	11/02/2019 99s	57	1	2	142	192	0	125	0	
6	12/12/2019 - 0	56	0	2	140	294	0	145	0	
7	12/12/2019 - 01	42	1	2	120	222	0	168	0	
8	12/12/2019,,?	52	1	3	172	199	1	153	0	
9	12/12/2019	57	1	3	150	168	0	173	0	

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

	Date	age	sex	chest_pain_type	resting_blood_pressure	cholesterol	fasting_blood_sugar	max_heart_rate_achieved	exercise_induced_angina	st_depre
0	12/29/2019	63	1	1	145	233	1	150	0	
1	11/18/2019	37	1	3	130	250	0	187	0	
2	12/05/2019	41	0	2	130	204	0	172	0	
3	01/01/2019	56	1	2	120	236	0	165	0	
4	04/09/2019	42	0	2	135	222	0	155	1	
5	11/02/2019	57	1	2	142	192	0	125	0	
6	12/12/2019	56	0	2	140	294	0	145	0	
7	12/12/2019	42	1	2	120	222	0	168	0	
8	12/12/2019	52	1	3	172	199	1	153	0	
9	12/12/2019	57	1	3	150	168	0	173	0	

Writing all modifications

```
data.to_csv(r'Final_Data.csv', index = False, header=True)
# index=true will write row names and header=true will write column names
```