Intro to DataFrame

2 dimensional labeled because its have row and column with column name.

In [2]:

```
#lets create with the help of dictionary
dict1={"Name":["Ram","Shyam","Lakshman"], "Highest-Marks":[77,82,95],"Subject":["Science","Math","English"]}
```

In [3]:

```
type(dict1)
```

Out[3]:

```
dict
```

In [4]:

```
#converting it into dataframe
import pandas as pd
df=pd.DataFrame(dict1)
```

In [5]:

```
df
```

Out[5]:

	Name	Highest-Marks	Subject
0	Ram	77	Science
1	Shyam	82	Math
2	Lakshman	95	English

Some Inbuilt Panadas DF function

In [6]:

```
df.head(2)
```

Out[6]:

	Name	Highest-Marks	Subject
0	Ram	77	Science
1	Shyam	82	Math

In [7]:

```
#lets use other file so that we can look its function very clearly
nba=pd.read_csv(r"C:\Users\USER\Downloads\nba.csv")
```

In [8]:

```
nba.head()       #First 5 row
```

Out[8]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0

In [9]:

```
nba.tail()        #last 5 row
```

Out[9]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	24.0	6-1	179.0	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21.0	C	26.0	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	Kansas	947276.0
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [10]:

```
nba.describe()
```

Out[10]:

	Number	Age	Weight	Salary
count	457.000000	457.000000	457.000000	4.460000e+02
mean	17.678337	26.938731	221.522976	4.842684e+06
std	15.966090	4.404016	26.368343	5.229238e+06
min	0.000000	19.000000	161.000000	3.088800e+04
25%	5.000000	24.000000	200.000000	1.044792e+06
50%	13.000000	26.000000	220.000000	2.839073e+06
75%	25.000000	30.000000	240.000000	6.500000e+06
max	99.000000	40.000000	307.000000	2.500000e+07

In [11]:

```
nba.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 458 entries, 0 to 457
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Name      457 non-null    object 
 1   Team      457 non-null    object 
 2   Number    457 non-null    float64
 3   Position  457 non-null    object 
 4   Age       457 non-null    float64
 5   Height    457 non-null    object 
 6   Weight    457 non-null    float64
 7   College   373 non-null    object 
 8   Salary    446 non-null    float64
dtypes: float64(4), object(5)
memory usage: 32.3+ KB
```

In [12]:

```
nba.shape              #number of rows and columns.
```

Out[12]:

```
(458, 9)
```

iloc[]

In [13]:

```
#Index location
nba.iloc[1:4,2:4]   #left side of comma is corresponding to row and right side is for column
```

Out[13]:

	Number	Position
1	99.0	SF
2	30.0	SG
3	28.0	SG

In [14]:

```
#Similarly for loc but here we can give index for row because row does not have name but for column we have to 
nba.loc[1:4,("Number","Position")]
```

Out[14]:

	Number	Position
1	99.0	SF
2	30.0	SG
3	28.0	SG
4	8.0	PF

In [15]:

```
#look the difference iloc does not inculde 4th row but loc is including 1 to 4th row.
```

Dropping Column

In [16]:

```
nba.drop('Age',axis=1) #1 means column and 0 means row
```

Out[16]:

	Name	Team	Number	Position	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	6-10	231.0	NaN	5000000.0
...	...	...	...	...	...	...	...	...
453	Shelvin Mack	Utah Jazz	8.0	PG	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	6-1	179.0	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21.0	C	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	7-0	231.0	Kansas	947276.0
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

458 rows × 8 columns

It got dropped, but it dropped permanently, no. This behavior is designed to prevent unintentional modifications to the original data.

In [17]:

```
nba.head()          #we can again see age, orignal data
```

Out[17]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0

In [18]:

```
#lets drop permanently
nba.drop('Age',axis=1, inplace=True)
```

In [19]:

```
nba.head()
```

Out[19]:

	Name	Team	Number	Position	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	6-10	231.0	NaN	5000000.0

Drop row

In [20]:

```
nba.drop([1,2,3],axis=0)  #see below, 1, 2 and 3rd row has been dropped temporarly.
```

Out[20]:

	Name	Team	Number	Position	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	6-2	180.0	Texas	7730337.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	6-9	240.0	NaN	12000000.0
6	Jordan Mickey	Boston Celtics	55.0	PF	6-8	235.0	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41.0	C	7-0	238.0	Gonzaga	2165160.0
...	...	...	...	...	...	...	...	...
453	Shelvin Mack	Utah Jazz	8.0	PG	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	6-1	179.0	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21.0	C	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	7-0	231.0	Kansas	947276.0
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

455 rows × 8 columns

Seeing some statistical with individual

In [21]:

```
nba.mean()
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_652\3862783939.py:1: FutureWarning: The default value of numeric_only in DataFrame.mean is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=None' is deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
  nba.mean()
```

Out[21]:

```
Number    1.767834e+01
Weight    2.215230e+02
Salary    4.842684e+06
dtype: float64
```

In [22]:

```
#Above mean() is showing mean of numeric values only. Note this.
```

In [23]:

```
nba.max()
```

```
C:\Users\USER\AppData\Local\Temp\ipykernel_652\129215889.py:1: FutureWarning: The default value of numeric_only in DataFrame.max is deprecated. In a future version, it will default to False. In addition, specifying 'numeric_only=None' is deprecated. Select only valid columns or specify the value of numeric_only to silence this warning.
  nba.max()
```

Out[23]:

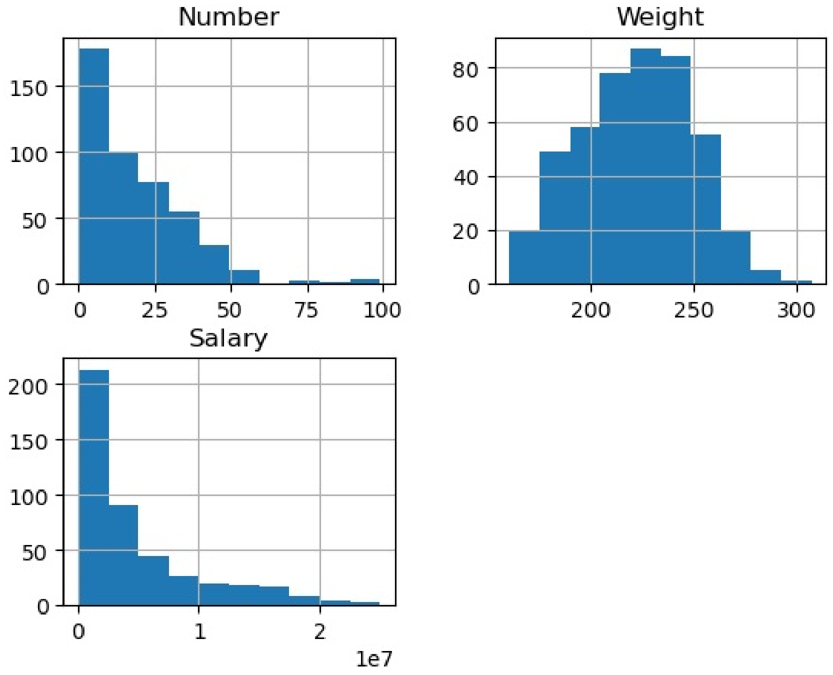
```
Number          99.0
Weight         307.0
Salary    25000000.0
dtype: float64
```

In [24]:

```
nba.hist()          #just checking graph of all numeric value
```

Out[24]:

```
array([[<Axes: title={'center': 'Number'}>,
        <Axes: title={'center': 'Weight'}>],
       [<Axes: title={'center': 'Salary'}>, <Axes: >]], dtype=object)
```



By seeing above graph - Can we say,Number and Salary is rightly skewed and Weight is somewhat (not perfect) but it is Gussian curve.

In [25]:

```
#value_counts()

nba["Team"].value_counts()            # its shows how many times team name is there in data.
```

Out[25]:

```
New Orleans Pelicans      19
Memphis Grizzlies         18
New York Knicks           16
Milwaukee Bucks           16
Boston Celtics            15
Brooklyn Nets             15
Portland Trail Blazers    15
Oklahoma City Thunder     15
Denver Nuggets            15
Washington Wizards        15
Miami Heat                15
Charlotte Hornets         15
Atlanta Hawks             15
San Antonio Spurs         15
Houston Rockets           15
Dallas Mavericks          15
Indiana Pacers            15
Detroit Pistons           15
Cleveland Cavaliers       15
Chicago Bulls             15
Sacramento Kings          15
Phoenix Suns              15
Los Angeles Lakers        15
Los Angeles Clippers      15
Golden State Warriors     15
Toronto Raptors           15
Philadelphia 76ers        15
Utah Jazz                 15
Orlando Magic             14
Minnesota Timberwolves    14
Name: Team, dtype: int64
```

In [26]:

```
#sorting dataframe according to team
nba.sort_values(by="Team")
```

Out[26]:

	Name	Team	Number	Position	Height	Weight	College	Salary
317	Lamar Patterson	Atlanta Hawks	13.0	SG	6-5	225.0	Pittsburgh	525093.0
309	Kent Bazemore	Atlanta Hawks	24.0	SF	6-5	201.0	Old Dominion	2000000.0
310	Tim Hardaway Jr.	Atlanta Hawks	10.0	SG	6-6	205.0	Michigan	1304520.0
311	Kirk Hinrich	Atlanta Hawks	12.0	SG	6-4	190.0	Kansas	2854940.0
312	Al Horford	Atlanta Hawks	15.0	C	6-10	245.0	Florida	12000000.0
...	...	...	...	...	...	...	...	...
369	Bradley Beal	Washington Wizards	3.0	SG	6-5	207.0	Florida	5694674.0
368	Alan Anderson	Washington Wizards	6.0	SG	6-6	220.0	Michigan State	4000000.0
382	John Wall	Washington Wizards	2.0	PG	6-4	195.0	Kentucky	15851950.0
370	Jared Dudley	Washington Wizards	1.0	SF	6-7	225.0	Boston College	4375000.0
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

458 rows × 8 columns

In [27]:

```
#What sorting_values is doing?
# - Its sorting dataframe according to minimum to maximum if it is numerical, or ascending to descending for ca
```

In [28]:

```
#lets look null values if there is in table
nba.isnull()
```

Out[28]:

	Name	Team	Number	Position	Height	Weight	College	Salary
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	True	False
...	...	...	...	...	...	...	...	...
453	False	False	False	False	False	False	False	False
454	False	False	False	False	False	False	True	False
455	False	False	False	False	False	False	True	False
456	False	False	False	False	False	False	False	False
457	True	True	True	True	True	True	True	True

458 rows × 8 columns

In [29]:

```
#lets see how many null values are there
nba.isnull().sum()
```

Out[29]:

```
Name         1
Team         1
Number       1
Position     1
Height       1
Weight       1
College     85
Salary      12
dtype: int64
```

In [30]:

```
#We check for null values but lets check if there is NaN file, not available
nba.isna()
```

Out[30]:

	Name	Team	Number	Position	Height	Weight	College	Salary
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	True
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	True	False
...	...	...	...	...	...	...	...	...
453	False	False	False	False	False	False	False	False
454	False	False	False	False	False	False	True	False
455	False	False	False	False	False	False	True	False
456	False	False	False	False	False	False	False	False
457	True	True	True	True	True	True	True	True

458 rows × 8 columns

In [31]:
```
nba.isna().sum()
```

Out[31]:
```
Name         1
Team         1
Number       1
Position     1
Height       1
Weight       1
College     85
Salary      12
dtype: int64
```

During data analysis, we do not want empty or not available value either we can drop it or replace it with mean, median and mode, whichever is effective.

In [32]:
```
#First of all lets look how to drop
nba.dropna()
```

Out[32]:

	Name	Team	Number	Position	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	6-6	235.0	Marquette	6796117.0
3	R.J. Hunter	Boston Celtics	28.0	SG	6-5	185.0	Georgia State	1148640.0
6	Jordan Mickey	Boston Celtics	55.0	PF	6-8	235.0	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41.0	C	7-0	238.0	Gonzaga	2165160.0
...	...	...	...	...	...	...	...	...
449	Rodney Hood	Utah Jazz	5.0	SG	6-8	206.0	Duke	1348440.0
451	Chris Johnson	Utah Jazz	23.0	SF	6-6	206.0	Dayton	981348.0
452	Trey Lyles	Utah Jazz	41.0	PF	6-10	234.0	Kentucky	2239800.0
453	Shelvin Mack	Utah Jazz	8.0	PG	6-3	203.0	Butler	2433333.0
456	Jeff Withey	Utah Jazz	24.0	C	7-0	231.0	Kansas	947276.0

364 rows × 8 columns

In [33]:
```
#see row gets decreased because all not available value got dropped.
```

In [35]:
```
nba.shape       #we all know until we placed inplace, it will not get dropped permanently, now lets practice re
```

Out[35]:
```
(458, 8)
```

In [37]:
```
nba['Number'].fillna(nba['Number'].mean(), inplace=True)
```

In [38]:
```
nba.isna().sum()
```

Out[38]:
```
Name         1
Team         1
Number       0
Position     1
Height       1
Weight       1
College     85
Salary      12
dtype: int64
```

In [39]:
```
#see Number it showing zero - there is no Not available value because we just replace it with fillna function.
```

In [40]:

```
nba.isnull().sum()
```

Out[40]:

```
Name         1
Team         1
Number       0
Position     1
Height       1
Weight       1
College     85
Salary      12
dtype: int64
```

Conditional statement

In [44]:

```
nba[nba['Weight']>180]      #it give output with records for people with weight above 180
```

Out[44]:

	Name	Team	Number	Position	Height	Weight	College	Salary
1	Jae Crowder	Boston Celtics	99.0	SF	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	6-9	240.0	NaN	12000000.0
...	...	...	...	...	...	...	...	...
451	Chris Johnson	Utah Jazz	23.0	SF	6-6	206.0	Dayton	981348.0
452	Trey Lyles	Utah Jazz	41.0	PF	6-10	234.0	Kentucky	2239800.0
453	Shelvin Mack	Utah Jazz	8.0	PG	6-3	203.0	Butler	2433333.0
455	Tibor Pleiss	Utah Jazz	21.0	C	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	7-0	231.0	Kansas	947276.0

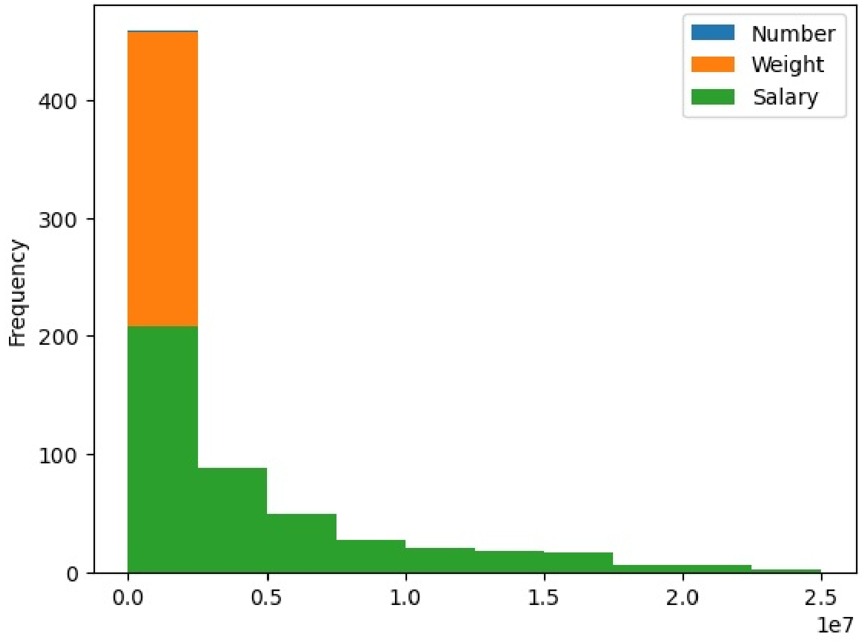
431 rows × 8 columns

In [46]:

```
#Like previously, we see histogram of all numerical in 3 different graph, if want to see in one?
nba.plot(kind='hist')
```

Out[46]:

```
<Axes: ylabel='Frequency'>
```



Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js