In [11]:

```
import pandas as pd
df = pd.DataFrame(columns=['name','age','percentage'])
df
```

Out[11]:

name	age	percentage

In [12]:

```
# Q1. Write a Python program to create a dataframe containing columns name, a
# rows to the dataframe. View the dataframe.
df.loc[0] = ['Aishwarya',20,99.00]
df.loc[1] = ['Mona',20,80.00]
df.loc[2] = ['Sayali',17,90.00]
df.loc[3] = ['Sita',18,95.00]
df.loc[4] = ['Sona',21,80.00]
df.loc[5] = ['Rita',15,85.00]
df.loc[6] = ['Rutuja',16,77.00]
df.loc[7] = ['Siddhi',17,78.00]
df.loc[8] = ['Riya',20,91.00]
df.loc[9] = ['Pinky',21,75.00]
df.loc[10] = ['Soniya',17,68.00]
df
```

Out[12]:

	name	age	percentage
0	Aishwarya	20	99.0
1	Mona	20	80.0
2	Sayali	17	90.0
3	Sita	18	95.0
4	Sona	21	80.0
5	Rita	15	85.0
6	Rutuja	16	77.0
7	Siddhi	17	78.0
8	Riya	20	91.0
9	Pinky	21	75.0
10	Soniya	17	68.0

In [13]:

```
# Q2. Write a Python program to print the shape, number of rows-columns, data
# the description of the data
print('shape: ',df.shape)
print('number of rows: ',df.shape[0])
print('number of columns: ',df.shape[1])
print('datatypes of all columns: ',df.dtypes)
print('Information of data:\n ',df.info())
print('Describing the data:\n ',df.describe())
```

```
shape:  (11, 3)
number of rows:  11
number of columns:  3
datatypes of all columns:  name           object
age            object
percentage    float64
dtype: object
<class 'pandas.core.frame.DataFrame'>
Int64Index: 11 entries, 0 to 10
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   name        11 non-null     object 
 1   age         11 non-null     object 
 2   percentage  11 non-null     float64
dtypes: float64(1), object(2)
memory usage: 352.0+ bytes
Information of data:
  None
Describing the data:
          percentage
count   11.000000
mean    83.454545
std      9.395357
min     68.000000
25%     77.500000
50%     80.000000
75%     90.500000
max     99.000000
```

In [14]:

```
# Q3. Write a Python program to view basic statistical details of the data.
df.describe()
```

Out[14]:

	percentage
count	11.000000
mean	83.454545
std	9.395357
min	68.000000
25%	77.500000
50%	80.000000
75%	90.500000
max	99.000000

In [15]:

```
# Q4. Write a Python program to Add 5 rows with duplicate values and missing
# ‘remarks’ with empty values. Display the data

import pandas as pd
# add in a table duplicate values and missing values
df = pd.DataFrame(columns=['name','age','percentage'])
df.loc[0] = ['Aishwarya',20,99]
df.loc[1] = ['Mona',20,None]
df.loc[2] = ['Sayali',None,90]
df.loc[3] = ['Sita',18,95]
df.loc[4] = ['Sayali',None,80]
df
# adding empty column
df["remarks"] = None
df
```

Out[15]:

	name	age	percentage	remarks
0	Aishwarya	20	99	None
1	Mona	20	None	None
2	Sayali	None	90	None
3	Sita	18	95	None
4	Sayali	None	80	None

In [16]:

```
import numpy as np
import pandas as pd
# created a table with duplicate values and missing values
data = np.array([['Aishwarya',20,99],['Rita',20,98],['Sita',21,85],['Priya',N
df = pd.DataFrame(data,columns=['name','age','percentage'])
df
# adding empty column
df["remarks"] = None
df
```

Out[16]:

	name	age	percentage	remarks
0	Aishwarya	20	99	None
1	Rita	20	98	None
2	Sita	21	85	None
3	Priya	None	87	None
4	Sita	23	None	None

In [17]:

```
# import numpy as nm
# You can use nm.nan as missing values
import numpy as np
df = pd.DataFrame(data,columns=['name','age','percentage'])
df.loc[0] = ['Aishwarya',20,99]
df.loc[1] = ['Mona',20,None]
df.loc[2] = ['Sayali',np.nan,90]
df.loc[3] = ['Sita',18,95]
df.loc[4] = ['Sayali',None,80]
df.loc[len(df.index)]=['Rita',np.nan,None]
df
```

Out[17]:

	name	age	percentage
0	Aishwarya	20	99
1	Mona	20	None
2	Sayali	NaN	90
3	Sita	18	95
4	Sayali	None	80
5	Rita	NaN	None

In [18]:

```python
# Q5. Write a Python program to get the number of observations, missing value
import numpy as np
df = pd.DataFrame(data,columns=['name','age','percentage'])
df.loc[0] = ['Aishwarya',20,99]
df.loc[1] = ['Mona',20,None]
df.loc[2] = ['Sayali',np.nan,80]
df.loc[3] = ['Sita',18,95]
df.loc[4] = ['Sayali',np.nan,80]
df.loc[5] = ['Sita',18,95]
df.loc[len(df.index)]=['Rita',np.nan,None]
df
print('number of observation: ',len(df.index))
DuplicateValues = df[df.duplicated()]
print('Duplicate rows are: \n')
DuplicateValues
```

```
number of observation:  7
Duplicate rows are: 
```

Out[18]:

	name	age	percentage
4	Sayali	NaN	80
5	Sita	18	95

In [19]:

```python
MissingValues = df[df['age'].isnull()]
print('Missing values rows are: \n')
MissingValues
```

```
Missing values rows are: 
```

Out[19]:

	name	age	percentage
2	Sayali	NaN	80
4	Sayali	NaN	80
6	Rita	NaN	None

In [20]:

```
# Q6. Write a Python program to drop ‘remarks’ column from the dataframe. Als
# values. Print the modified data
import numpy as np
import pandas as pd
# created a table with duplicate values and missing values
data = np.array([['Aishwarya',20,99],['Rita',20,None],['Sita',23,85],['Priya'
df = pd.DataFrame(data,columns=['name','age','percentage'])
df
# adding empty column
df["remarks"] = None
# drop a column 'remarks'
df.drop(columns='remarks', inplace=True)
df
```

Out[20]:

	name	age	percentage
0	Aishwarya	20	99
1	Rita	20	None
2	Sita	23	85
3	Priya	None	87
4	Sita	23	85

In [21]:

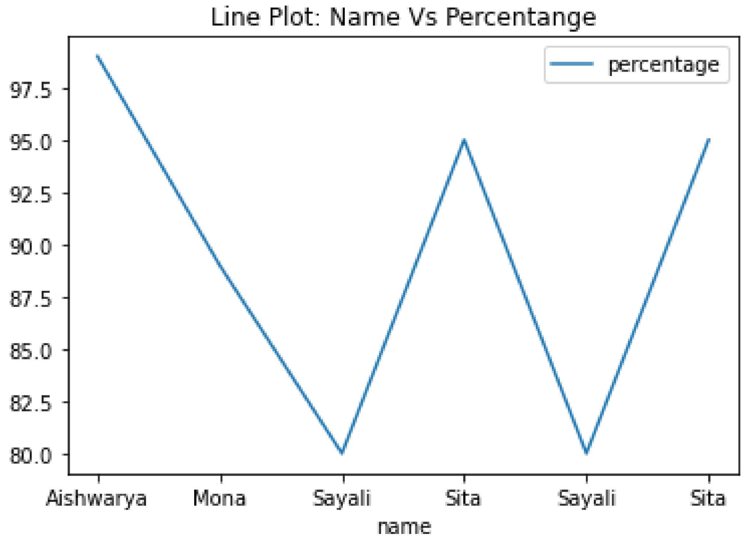
```
# removing all duplicate values
df.drop_duplicates(keep='first',inplace=True)
df
# removing all rows containing single NaN or none values
df.dropna()
```

Out[21]:

	name	age	percentage
0	Aishwarya	20	99
2	Sita	23	85

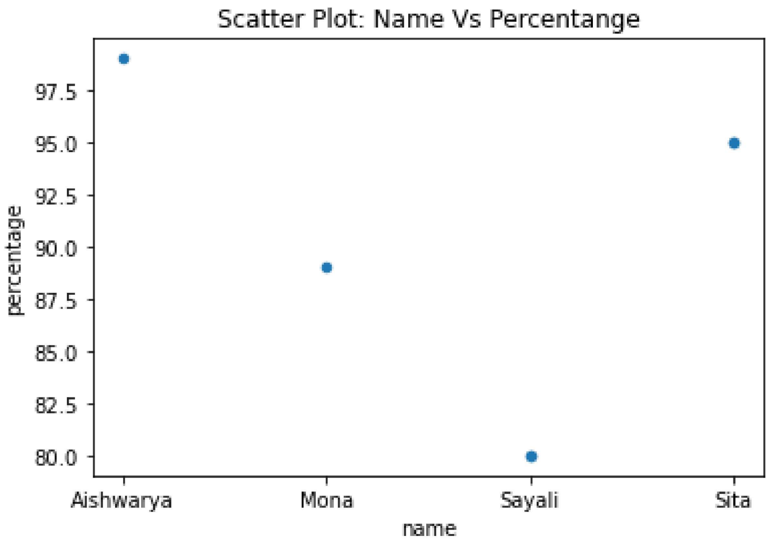
In [22]:

```python
# Q7. Write a Python program to generate a line plot of name vs percentage
import matplotlib.pyplot as plt
import numpy as np
df = pd.DataFrame(data,columns=['name','age','percentage'])
df.loc[0] = ['Aishwarya',20,99]
df.loc[1] = ['Mona',20,89]
df.loc[2] = ['Sayali',25,80]
df.loc[3] = ['Sita',18,95]
df.loc[4] = ['Sayali',18,80]
df.loc[5] = ['Sita',18,95]
df.plot('name','percentage')
plt.title('Line Plot: Name Vs Percentange')
plt.show()
```



In [23]:

```
# Q8. Write a Python program to generate a scatter plot of name vs percentage
import matplotlib.pyplot as plt
import numpy as np
df = pd.DataFrame(data,columns=['name','age','percentage'])
df.loc[0] = ['Aishwarya',20,99]
df.loc[1] = ['Mona',20,89]
df.loc[2] = ['Sayali',25,80]
df.loc[3] = ['Sita',18,95]
df.loc[4] = ['Sayali',18,80]
df.loc[5] = ['Sita',18,95]
df.plot(kind='scatter',x='name',y='percentage')
plt.title('Scatter Plot: Name Vs Percentange')
plt.show()
```



In [17]:

```
# Q1. Download the heights and weights dataset and load the dataset from a gi
# Print the first, last 10 rows and random 20 rows. (https://www.kaggle.com/b
import pandas as pd
df=pd.read_csv('HeightWeight.csv')
print('Reading first 10 rows: ')
print(df.head(10))
print('Reading last 10 rows: ')
print(df.tail(10))
print('Reading random 20 rows: ')
print(df.sample(20))
```

```
Reading first 10 rows: 
   Index  Height(Inches)  Weight(Pounds)
0      1        65.78331        112.9925
1      2        71.51521        136.4873
2      3        69.39874        153.0269
3      4        68.21660        142.3354
4      5        67.78781        144.2971
5      6        68.69784        123.3024
6      7        69.80204        141.4947
7      8        70.01472        136.4623
8      9        67.90265        112.3723
9     10        66.78236        120.6672
Reading last 10 rows: 
       Index  Height(Inches)  Weight(Pounds)
24990  24991        69.97767        125.3672
24991  24992        71.91656        128.2840
24992  24993        70.96218        146.1936
24993  24994        66.19462        118.7974
24994  24995        67.21126        127.6603
24995  24996        69.50215        118.0312
24996  24997        64.54826        120.1932
24997  24998        64.69855        118.2655
24998  24999        67.52918        132.2682
24999  25000        68.87761        124.8742
Reading random 20 rows: 
       Index  Height(Inches)  Weight(Pounds)
12355  12356        68.81960        128.3999
12688  12689        68.28734        116.2769
3264    3265        66.77115        130.2165
9684    9685        67.57915        113.2566
23540  23541        70.12057        128.2378
2411    2412        68.02851        135.0685
16899  16900        67.56819        135.5679
14020  14021        70.00883        129.8113
18937  18938        68.03350        132.2110
24497  24498        68.47918        115.6665
13805  13806        68.55061        134.7133
24984  24985        67.58699        127.7214
8099    8100        66.12177        120.4629
10763  10764        69.11316        151.1101
2653    2654        68.82694        127.2504
3850    3851        68.33558        127.2486
11831  11832        67.71809        123.3103
5775    5776        66.94962        137.4870
```

```
777       778         66.90106        132.9048
5478     5479         69.58930        127.2246
```

In [18]:

```
# Q2. Write a Python program to find the shape, size, datatypes of the datafr
print(df.shape)
print(df.size)
print(df.dtypes)
```

```
(25000, 3)
75000
Index                 int64
Height(Inches)      float64
Weight(Pounds)      float64
dtype: object
```

In [58]:

```
# Q3. Write a Python program to view basic statistical details of the data.
print('Statistical details:-\n')
df.describe()
```

```
Statistical details:-
```

Out[58]:

	Index	Height(Inches)	Weight(Pounds)	BMI
count	25000.000000	25000.000000	25000.000000	25000.000000
mean	12500.500000	67.993114	127.079421	0.027482
std	7217.022701	1.901679	11.660898	0.002207
min	1.000000	60.278360	78.014760	0.018591
25%	6250.750000	66.704397	119.308675	0.025998
50%	12500.500000	67.995700	127.157750	0.027454
75%	18750.250000	69.272958	134.892850	0.028955
max	25000.000000	75.152800	170.924000	0.037014

In [30]:

```
# Q4. Write a Python program to get the number of observations, missing value
print('number of observation: ',len(df.index))
print('Total of missing values: ',df.isnull().sum().sum())
```

```
number of observation:  25000
Total of missing values:  0
```

In [46]:

```
# Q5. Write a Python program to add a column to the dataframe “BMI” which is
df['BMI'] = (df['Weight(Pounds)']) / (df['Height(Inches)']**2)
df
```

Out[46]:

	Index	Height(Inches)	Weight(Pounds)	BMI
0	1	65.78331	112.9925	0.026111
1	2	71.51521	136.4873	0.026687
2	3	69.39874	153.0269	0.031773
3	4	68.21660	142.3354	0.030587
4	5	67.78781	144.2971	0.031402
...	...	...	...	...
24995	24996	69.50215	118.0312	0.024434
24996	24997	64.54826	120.1932	0.028848
24997	24998	64.69855	118.2655	0.028253
24998	24999	67.52918	132.2682	0.029005
24999	25000	68.87761	124.8742	0.026322

25000 rows × 4 columns

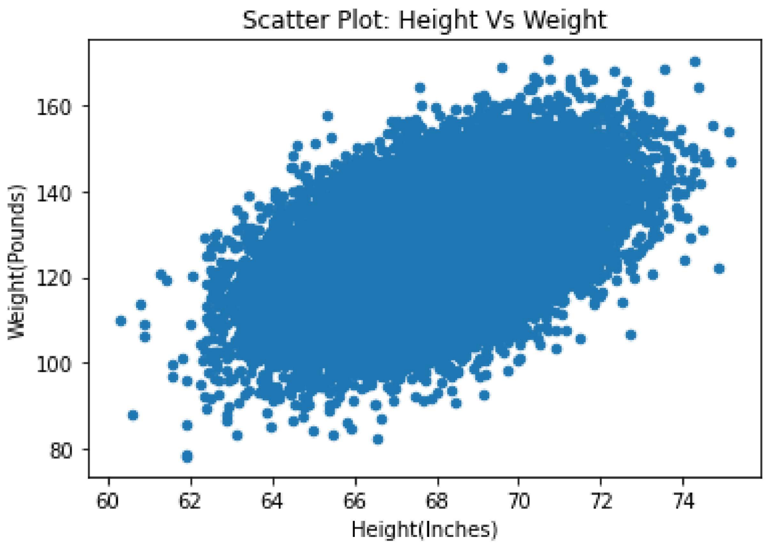
In [54]:

```
# Q6. Write a Python program to find the maximum and minimum BMI.
print(df["BMI"].max())
print(df["BMI"].min())
```

```
0.03701443692089851
0.018591137267932455
```

In [56]:

```
# Q7. Write a Python program to generate a scatter plot of height vs weight.
import matplotlib.pyplot as plt
df.plot(kind='scatter',x='Height(Inches)',y='Weight(Pounds)')
plt.title('Scatter Plot: Height Vs Weight')
plt.show()
```



In [ ]: