In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df=pd.read_csv("3_Fitness-1.csv")
df
```

Out[2]:

	Row Labels	Sum of Jan	Sum of Feb	Sum of Mar	Sum of Total Sales
0	A	5.62%	7.73%	6.16%	75
1	B	4.21%	17.27%	19.21%	160
2	C	9.83%	11.60%	5.17%	101
3	D	2.81%	21.91%	7.88%	127
4	E	25.28%	10.57%	11.82%	179
5	F	8.15%	16.24%	18.47%	167
6	G	18.54%	8.76%	17.49%	171
7	H	25.56%	5.93%	13.79%	170
8	Grand Total	100.00%	100.00%	100.00%	1150

In [3]:
```
df.head()
```

Out[3]:

	Row Labels	Sum of Jan	Sum of Feb	Sum of Mar	Sum of Total Sales
0	A	5.62%	7.73%	6.16%	75
1	B	4.21%	17.27%	19.21%	160
2	C	9.83%	11.60%	5.17%	101
3	D	2.81%	21.91%	7.88%	127
4	E	25.28%	10.57%	11.82%	179

DATA CLEANING AND DATA PREPROCESSING

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 5 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Row Labels         9 non-null      object
 1   Sum of Jan         9 non-null      object
 2   Sum of Feb         9 non-null      object
 3   Sum of Mar         9 non-null      object
```

```
 4   Sum of Total Sales  9 non-null      int64
dtypes: int64(1), object(4)
memory usage: 488.0+ bytes
```

In [5]:
```
df.describe()
```

Out[5]:

	Sum of Total Sales
count	9.000000
mean	255.555556
std	337.332963
min	75.000000
25%	127.000000
50%	167.000000
75%	171.000000
max	1150.000000

In [6]:
```
df.columns
```

Out[6]:
```
Index(['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales'],
      dtype='object')
```

In [7]:
```
df1=df.dropna(axis=1)
df1
```

Out[7]:

	Row Labels	Sum of Jan	Sum of Feb	Sum of Mar	Sum of Total Sales
0	A	5.62%	7.73%	6.16%	75
1	B	4.21%	17.27%	19.21%	160
2	C	9.83%	11.60%	5.17%	101
3	D	2.81%	21.91%	7.88%	127
4	E	25.28%	10.57%	11.82%	179
5	F	8.15%	16.24%	18.47%	167
6	G	18.54%	8.76%	17.49%	171
7	H	25.56%	5.93%	13.79%	170
8	Grand Total	100.00%	100.00%	100.00%	1150

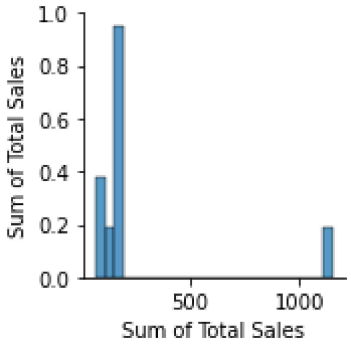
In [8]:
```
df1.columns
```

Out[8]:
```
Index(['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales'],
      dtype='object')
```

EDA AND VISUALIZATION

In [9]:
```
sns.pairplot(df1)
```

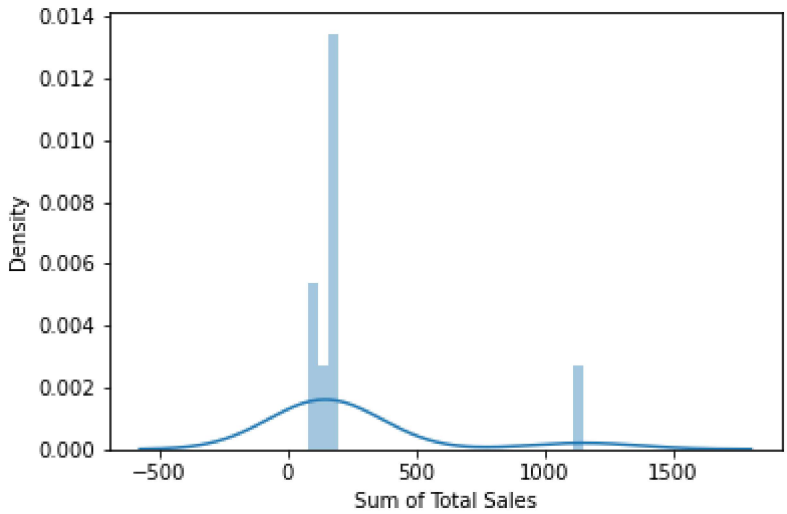
Out[9]:
```
<seaborn.axisgrid.PairGrid at 0x235c03294c0>
```



In [10]:
```
sns.distplot(df1['Sum of Total Sales'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[10]:
```
<AxesSubplot:xlabel='Sum of Total Sales', ylabel='Density'>
```



In [11]:
```
sns.heatmap(df1.corr())
```

Out[11]:
```
<AxesSubplot:>
```

TO TRAIN THE MODEL AND MODEL BULDING

In [12]:
```
x=df[['Sum of Total Sales','Sum of Total Sales' ]]
y=df['Sum of Total Sales']
```

In [13]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [14]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[14]:
```
LinearRegression()
```

In [15]:
```
lr.intercept_
```

Out[15]:
```
1.7053025658242404e-13
```

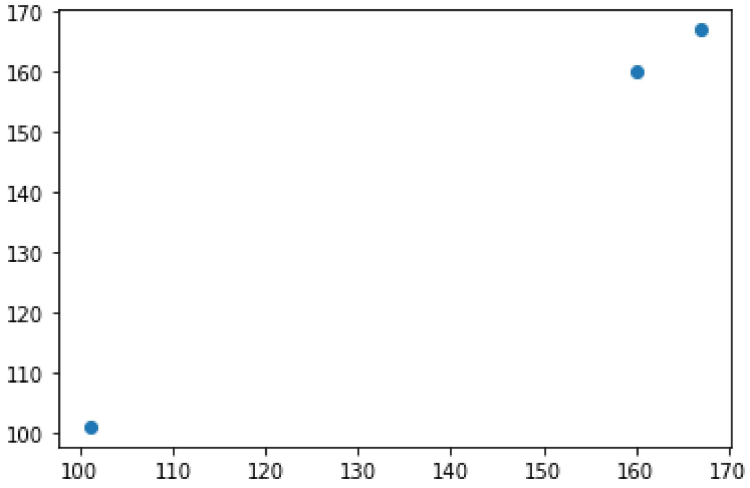
In [16]:
```
coeff=pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[16]:

	Co-efficient
Sum of Total Sales	0.5
Sum of Total Sales	0.5

In [17]:
```
prediction =lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[17]:
```
<matplotlib.collections.PathCollection at 0x235c253bd30>
```



ACCURACY

In [18]:
```
lr.score(x_test,y_test)
```

Out[18]: 1.0

In [19]:
```
lr.score(x_train,y_train)
```

Out[19]: 1.0

In [20]:
```
from sklearn.linear_model import Ridge,Lasso
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[20]: Ridge(alpha=10)

In [21]:
```
rr.score(x_test,y_test)
```

Out[21]: 0.999999998834103

In [22]:
```
rr.score(x_train,y_train)
```

Out[22]: 0.9999999999654285

In [23]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[23]: Lasso(alpha=10)

In [24]:
```
la.score(x_train,y_train)
```

Out[24]: 0.9999999950216595

In [25]:
```
la.score(x_test,y_test)
```

Out[25]: 0.9999998321088441