mk 31/07/23

In [212]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [213]:
```
x=pd.read_csv(r"C:\Users\user\Downloads\3_Fitness-1.csv")
x
```

Out[213]:

	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 [214]:
```
x.head(10)
```

Out[214]:

	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 [215]:
```
x.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 [216]:
```
x.columns
```

Out[216]:
```
Index(['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales'],
      dtype='object')
```

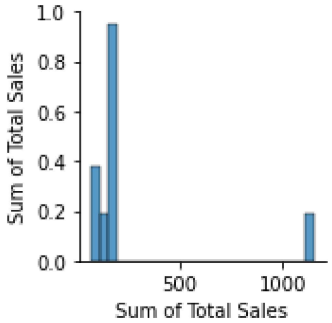
In [217]:
```
x.describe()
```

Out[217]:

	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 [218]:
```
sns.pairplot(x)
```

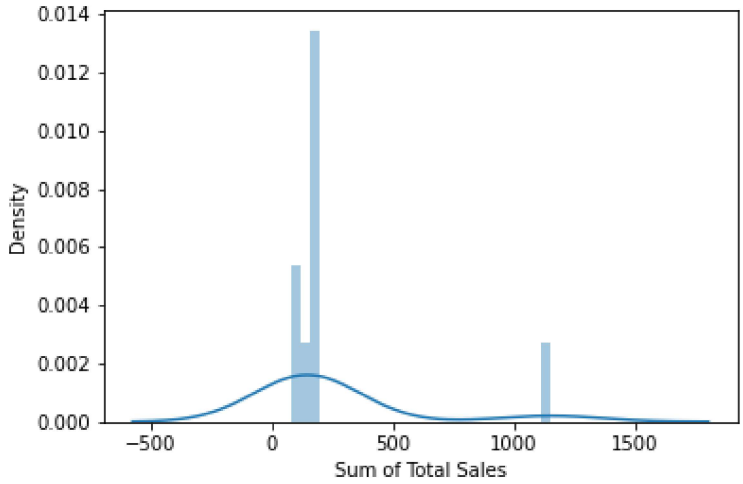
Out[218]:
```
<seaborn.axisgrid.PairGrid at 0x110f5006310>
```



In [219]:
```
sns.distplot(x['Sum of Total Sales'])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarni
ng: `distplot` is a deprecated function and will be removed in a future version. Plea
se adapt your code to use either `displot` (a figure-level function with similar flex
ibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[219]: <AxesSubplot:xlabel='Sum of Total Sales', ylabel='Density'>



In [220]:
```
x1=x[['Sum of Total Sales','Sum of Jan']]
```

In [221]:
```
sns.heatmap(x1.corr())
```

Out[221]: <AxesSubplot:>

In [226]:
```
a=x1[['Sum of Total Sales']]
b=x1['Sum of Total Sales']
```

In [227]:
```
from sklearn.model_selection import train_test_split
a_train,a_test, b_train, b_test=train_test_split(a,b,test_size=0.3)
```

In [228]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(a_train,b_train)
```

Out[228]:
```
LinearRegression()
```

In [229]:
```
print(lr.intercept_)
```

```
2.842170943040401e-14
```

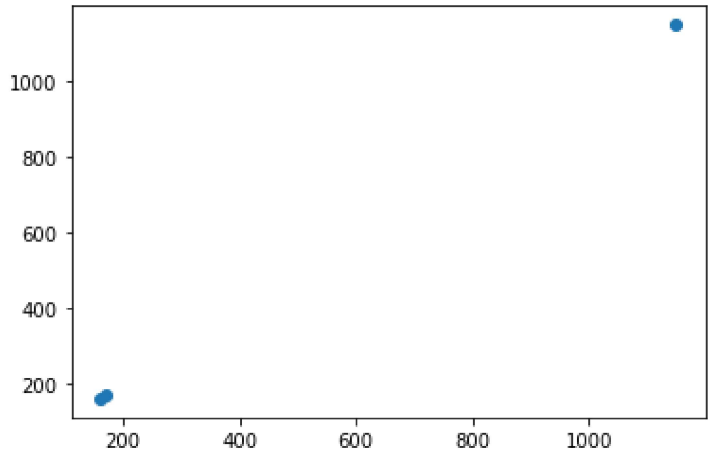
In [230]:
```
coeff=pd.DataFrame(lr.coef_,a.columns,columns=['Co-efficient'])
coeff
```

Out[230]:

	Co-efficient
Sum of Total Sales	1.0

In [231]:
```
prediction=lr.predict(a_test)
plt.scatter(b_test,prediction)
```

Out[231]:
```
<matplotlib.collections.PathCollection at 0x110f6ebe460>
```



In [232]:
```
lr.score(a_test,b_test)
```

Out[232]:
```
1.0
```

In [233]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [234]:
```
rr=Ridge(alpha=10)
rr.fit(a_train,b_train)
```

Out[234]:
```
Ridge(alpha=10)
```

In [235]:
```
rr.score(a_test,b_test)
```

Out[235]:
```
0.999998034952742
```

In [236]:
```
la=Lasso(alpha=10)
la.fit(a_train,b_train)
```

Out[236]:
```
Lasso(alpha=10)
```

In [237]:
```
la.score(a_test,b_test)
```

Out[237]:
```
0.9999291008588174
```

In [238]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(a_train,b_train)
```

Out[238]:
```
ElasticNet()
```

In [239]:
```
en.coef_
```

Out[239]:
```
array([0.99933293])
```

In [240]:
```
en.predict(a_test)
```

Out[240]:
```
array([1149.32392017,  170.97698594,  159.98432375])
```

In [241]:
```
en.intercept_
```

Out[241]:
```
0.09105564511645525
```

In [242]:
```
en.score(a_test,b_test)
```

Out[242]:
```
0.9999992914814593
```

In [243]:
```
from sklearn import metrics
```

In [244]:
```
print("Mean Absolute Error",metrics.mean_absolute_error(b_test,prediction))
```

```
Mean Absolute Error 7.579122514774402e-14
```

In [245]:
```
print("Mean Squared Error",metrics.mean_squared_error(b_test,prediction))
```

```
Mean Squared Error 1.7232929428188076e-26
```

In [246]:
```
print("Root Mean Squared Error",np.sqrt(metrics.mean_squared_error(b_test,prediction)))
```

```
Root Mean Squared Error 1.3127425272378462e-13
```

In [ ]: