Problem Statement

Linear Regression

Import Libraries

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

In [2]:
```
a=pd.read_csv("fitness.csv")
a
```

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

To display top 10 rows

In [3]:
```
c=a.head(15)
c
```

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
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

	Row Labels	Sum of Jan	Sum of Feb	Sum of Mar	Sum of Total Sales
8	Grand Total	100.00%	100.00%	100.00%	1150

To find Missing values

In [4]:
```
c.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
```

To display summary of statistics

In [5]:
```
a.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

To display column heading

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

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

Pairplot

In [7]:
```
s=a.dropna(axis=1)
s
```

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]:

```
s.columns
```

Out[8]:

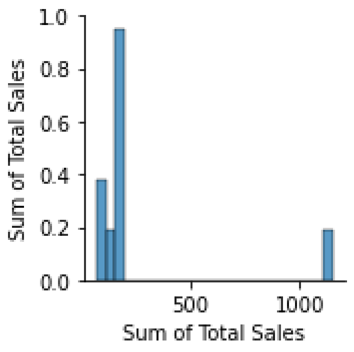
```
Index(['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales'],
      dtype='object')
```

In [9]:

```
sns.pairplot(a)
```

Out[9]:

```
<seaborn.axisgrid.PairGrid at 0x1c5a1213be0>
```



Distribution Plot

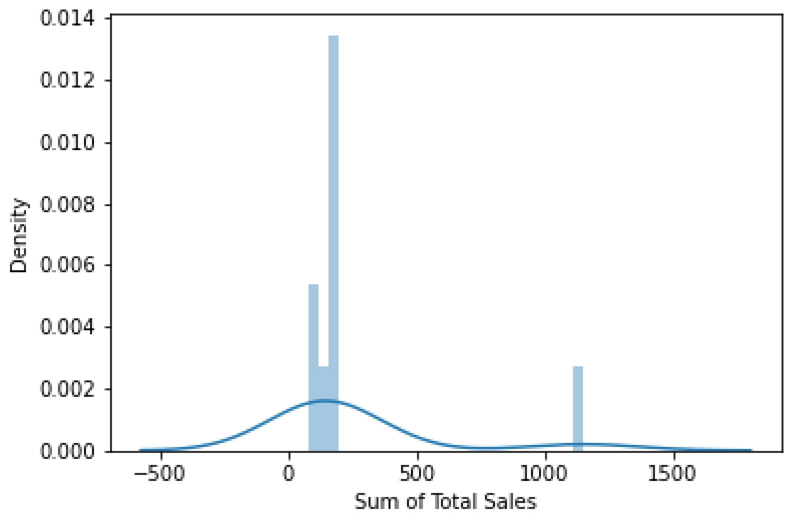
In [10]:

```
sns.distplot(c['Sum of Total Sales'])
```

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

Out[10]:

```
<AxesSubplot:xlabel='Sum of Total Sales', ylabel='Density'>
```

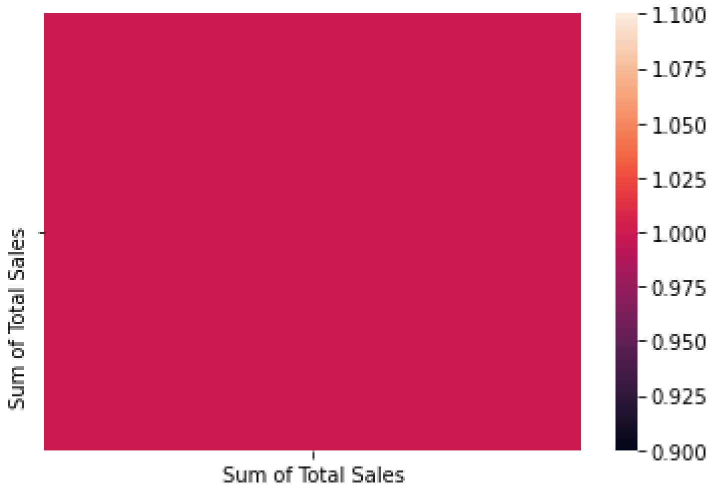


Correlation

Train the model - Model Building

In [11]:
```
b=a[['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales']]
sns.heatmap(b.corr())
```

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



In [12]:
```
g=c[['Sum of Total Sales']]
h=c['Sum of Total Sales']
```

To split dataset into training end test

In [13]:
```
from sklearn.model_selection import train_test_split
g_train,g_test,h_train,h_test=train_test_split(g,h,test_size=0.6)
```

To run the model

In [14]:
```
from sklearn.linear_model import LinearRegression
```

In [15]:
```
lr=LinearRegression()
lr.fit(g_train,h_train)
```

Out[15]:
```
LinearRegression()
```

In [16]:
```
print(lr.intercept_)
```

```
-8.526512829121202e-14
```

Coeffecient

In [17]:
```
coeff=pd.DataFrame(lr.coef_,g.columns,columns=['Co-effecient'])
coeff
```

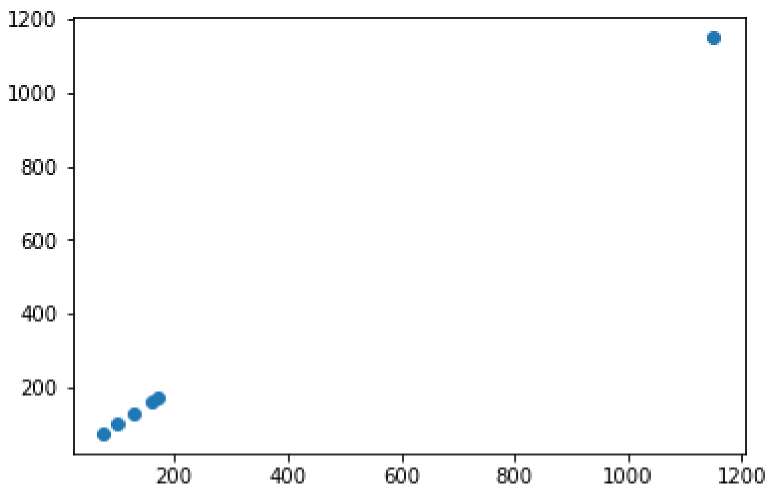
Out[17]:

	Co-effecient
Sum of Total Sales	1.0

Best Fit line

In [18]:
```
prediction=lr.predict(g_test)
plt.scatter(h_test,prediction)
```

Out[18]:
```
<matplotlib.collections.PathCollection at 0x1c5a34183d0>
```



To find score

In [19]:
```
print(lr.score(g_test,h_test))
```

```
1.0
```

Import Lasso and ridge

In [20]:
```
from sklearn.linear_model import Ridge,Lasso
```

Ridge

In [21]:
```
ri=Ridge(alpha=5)
ri.fit(g_train,h_train)
```

Out[21]:
```
Ridge(alpha=5)
```

In [22]:
```
ri.score(g_test,h_test)
```

Out[22]:
```
0.9956420479115011
```

In [23]:
```
ri.score(g_train,h_train)
```

Out[23]:
```
0.9960609933299487
```

Lasso

In [24]:
```
l=Lasso(alpha=6)
l.fit(g_train,h_train)
```

Out[24]:
```
Lasso(alpha=6)
```

In [25]:
```
l.score(g_test,h_test)
```

Out[25]:
```
0.9357035169610383
```

In [26]:
```
ri.score(g_train,h_train)
```

Out[26]:
```
0.9960609933299487
```