D10

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

In [2]:
```
df=pd.read_csv(r"C:\Users\user\Downloads\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(10)
```

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
8	Grand Total	100.00%	100.00%	100.00%	1150

In [4]:
```
df["Sum of Jan"]=df["Sum of Jan"].replace("%","",regex=True).astype(float)
df["Sum of Feb"]=df["Sum of Feb"].replace("%","",regex=True).astype(float)
df["Sum of Mar"]=df["Sum of Mar"].replace("%","",regex=True).astype(float)
df
```

Out[4]:

	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 [5]:
```
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      float64
 2   Sum of Feb          9 non-null      float64
 3   Sum of Mar          9 non-null      float64
 4   Sum of Total Sales  9 non-null      int64  
dtypes: float64(3), int64(1), object(1)
memory usage: 488.0+ bytes
```

In [6]:
```
df.describe()
```

Out[6]:

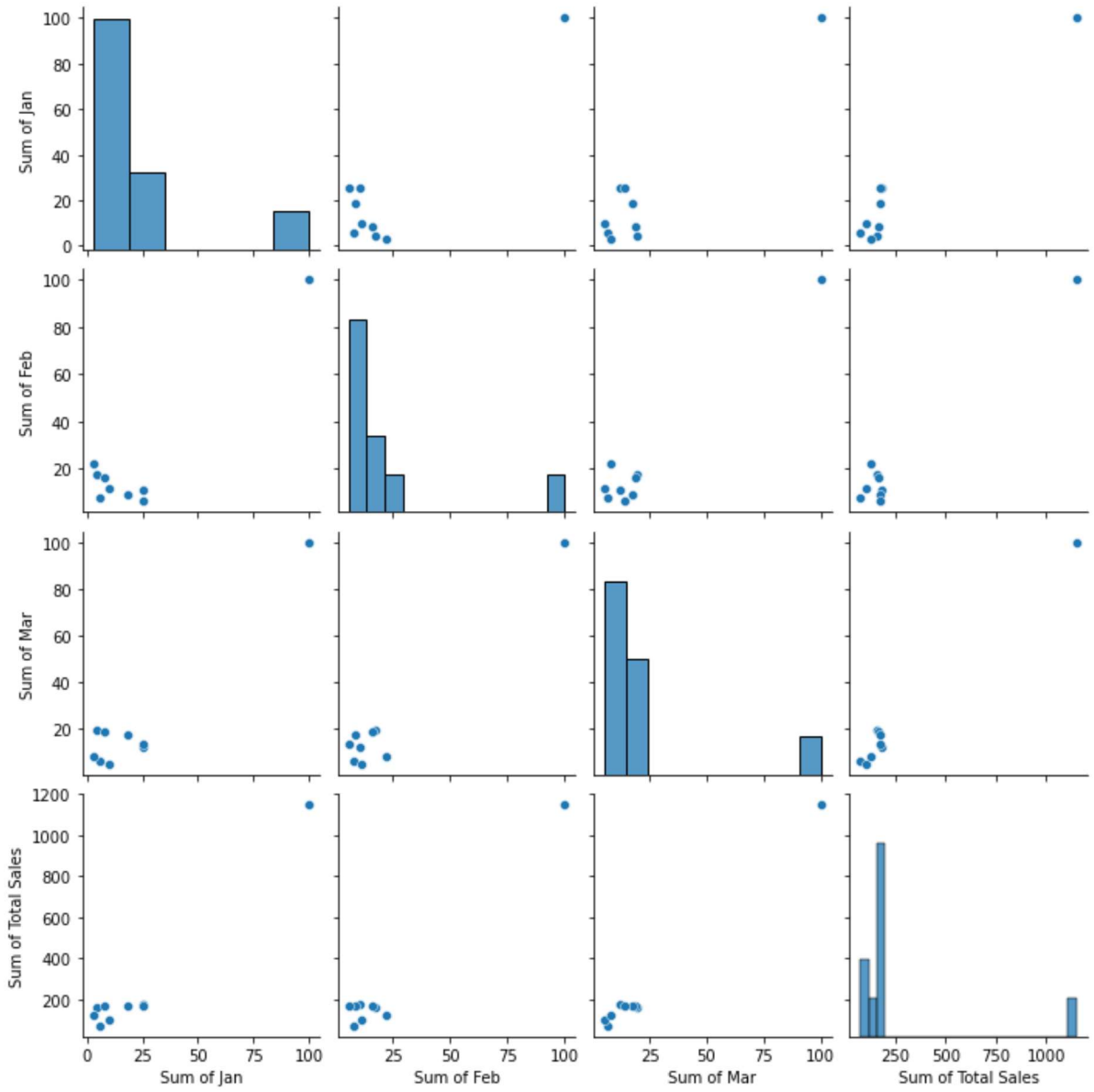
	Sum of Jan	Sum of Feb	Sum of Mar	Sum of Total Sales
count	9.000000	9.000000	9.000000	9.000000
mean	22.222222	22.223333	22.221111	255.555556
std	30.438329	29.612265	29.640999	337.332963
min	2.810000	5.930000	5.170000	75.000000
25%	5.620000	8.760000	7.880000	127.000000
50%	9.830000	11.600000	13.790000	167.000000
75%	25.280000	17.270000	18.470000	171.000000
max	100.000000	100.000000	100.000000	1150.000000

In [7]:
```
df.columns
```

Out[7]:
```
Index(['Row Labels', 'Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales'],
      dtype='object')
```

In [8]:
```
sns.pairplot(df)
```

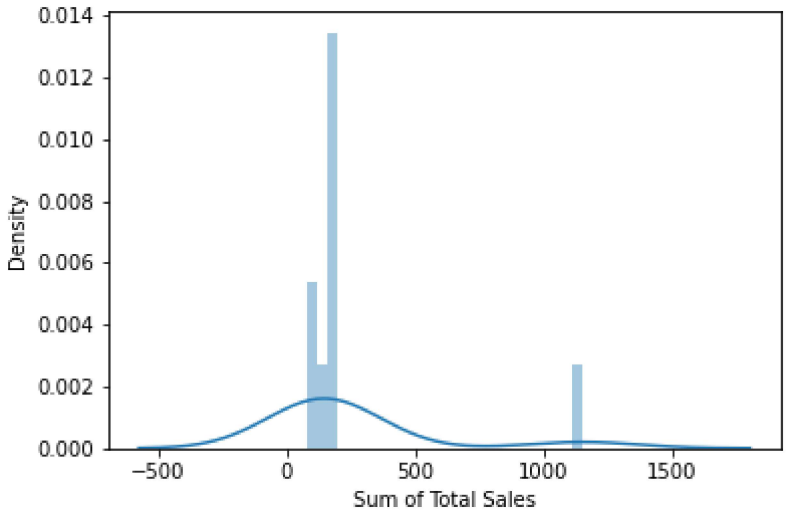
Out[8]:
```
<seaborn.axisgrid.PairGrid at 0x27b0776a220>
```



In [9]:
```
sns.distplot(df["Sum of Total Sales"])
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function for hi
stograms).
  warnings.warn(msg, FutureWarning)
```

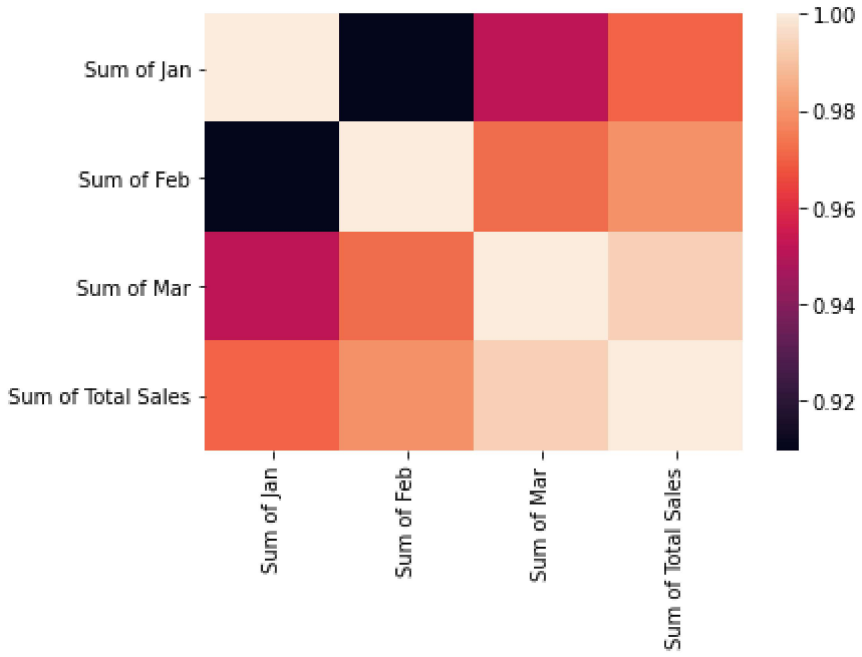
Out[9]: <AxesSubplot:xlabel='Sum of Total Sales', ylabel='Density'>



In [10]:
```
df1=df[['Sum of Jan', 'Sum of Feb', 'Sum of Mar',
       'Sum of Total Sales']]
```

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

Out[11]: <AxesSubplot:>



In [12]:
```
x=df1[['Sum of Jan', 'Sum of Feb', 'Sum of Mar']]
y=df1['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]:
```
print(lr.intercept_)
```

```
-0.01973032106442929
```

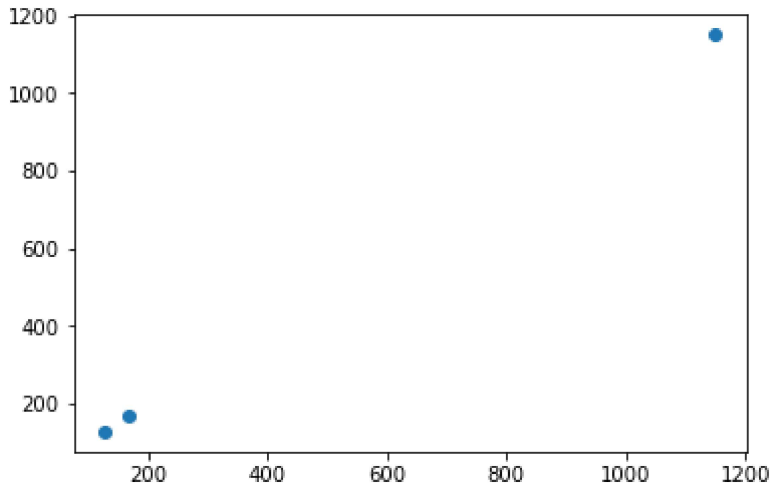
In [16]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])
coeff
```

Out[16]:

	Co-efficient
Sum of Jan	3.560449
Sum of Feb	3.881471
Sum of Mar	4.060133

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

Out[17]:
```
<matplotlib.collections.PathCollection at 0x27b0a3151c0>
```



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

```
0.9999999471505059
```

In [19]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [20]:
```
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]:
```
0.988982187114521
```

In [22]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[22]:
```
Lasso(alpha=10)
```

In [23]:
```
la.score(x_test,y_test)
```

Out[23]:
```
0.9590578412307781
```

In [24]:
```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[24]:
```
ElasticNet()
```

In [25]:
```
print(en.coef_)
```

```
[3.45305646 3.551375   4.0797226 ]
```

In [26]:
```
print(en.intercept_)
```

```
4.736837156624404
```

In [27]:
```
print(en.predict(x_train))
```

```
[158.97792232 177.79045934  76.72623444 170.31598867 171.22089722
 100.96849801]
```

In [28]:
```
print(en.score(x_train,y_train))
```

```
0.9994056978973949
```

In [29]:
```
from sklearn import metrics
```

In [30]:
```
print("Mean Absolytre Error:",metrics.mean_absolute_error(y_test,prediction))
```

```
Mean Absolytre Error: 0.07709068459324915
```

In [31]:
```
print("Mean Square Error:",metrics.mean_squared_error(y_test,prediction))
```

```
Mean Square Error: 0.011828996903466442
```

In [32]:
```
print("Root Mean Square Error:",np.sqrt(metrics.mean_absolute_error(y_test,pre
```

```
Root Mean Square Error: 0.2776520927226178
```

In [ ]:
```

```