A real estate agent want help to predict the house price for regions in Usa.he gave us the dataset to work on to use linear Regression model.Create a model that helps him to estimate

Data Collection

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

In [118]:
```
#import the dataset
data=pd.read_csv(r"C:\Users\user\Desktop\Vicky\17_student_marks.csv")
```

In [119]:
```
#to display top 10 rows
data.head()
```

Out[119]:

	Student_ID	Test_1	Test_2	Test_3	Test_4	Test_5	Test_6	Test_7	Test_8	Test_9	Test_10	T
0	22000	78	87	91	91	88	98	94	100	100	100	
1	22001	79	71	81	72	73	68	59	69	59	60	
2	22002	66	65	70	74	78	86	87	96	88	82	
3	22003	60	58	54	61	54	57	64	62	72	63	
4	22004	99	95	96	93	97	89	92	98	91	98	

In [120]:
```
#to display null values
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 56 entries, 0 to 55
Data columns (total 13 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Student_ID  56 non-null     int64
 1   Test_1      56 non-null     int64
 2   Test_2      56 non-null     int64
 3   Test_3      56 non-null     int64
 4   Test_4      56 non-null     int64
 5   Test_5      56 non-null     int64
 6   Test_6      56 non-null     int64
 7   Test_7      56 non-null     int64
 8   Test_8      56 non-null     int64
 9   Test_9      56 non-null     int64
 10  Test_10     56 non-null     int64
 11  Test_11     56 non-null     int64
 12  Test_12     56 non-null     int64
dtypes: int64(13)
memory usage: 5.8 KB
```

In [121]:
```
data.shape
```

Out[121]:
```
(56, 13)
```

In [122]:
```
#to display summary of statistics
data.describe()
```

Out[122]:

	Student_ID	Test_1	Test_2	Test_3	Test_4	Test_5	Test_6	
count	56.000000	56.000000	56.000000	56.000000	56.000000	56.000000	56.000000	5
mean	22027.500000	70.750000	69.196429	68.089286	67.446429	67.303571	66.000000	6
std	16.309506	17.009356	17.712266	18.838333	19.807179	20.746890	21.054043	2
min	22000.000000	40.000000	34.000000	35.000000	28.000000	26.000000	29.000000	2
25%	22013.750000	57.750000	55.750000	53.000000	54.500000	53.750000	50.250000	4
50%	22027.500000	70.500000	68.500000	70.000000	71.500000	69.000000	65.500000	6
75%	22041.250000	84.000000	83.250000	85.000000	84.000000	85.250000	83.750000	8
max	22055.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	1(

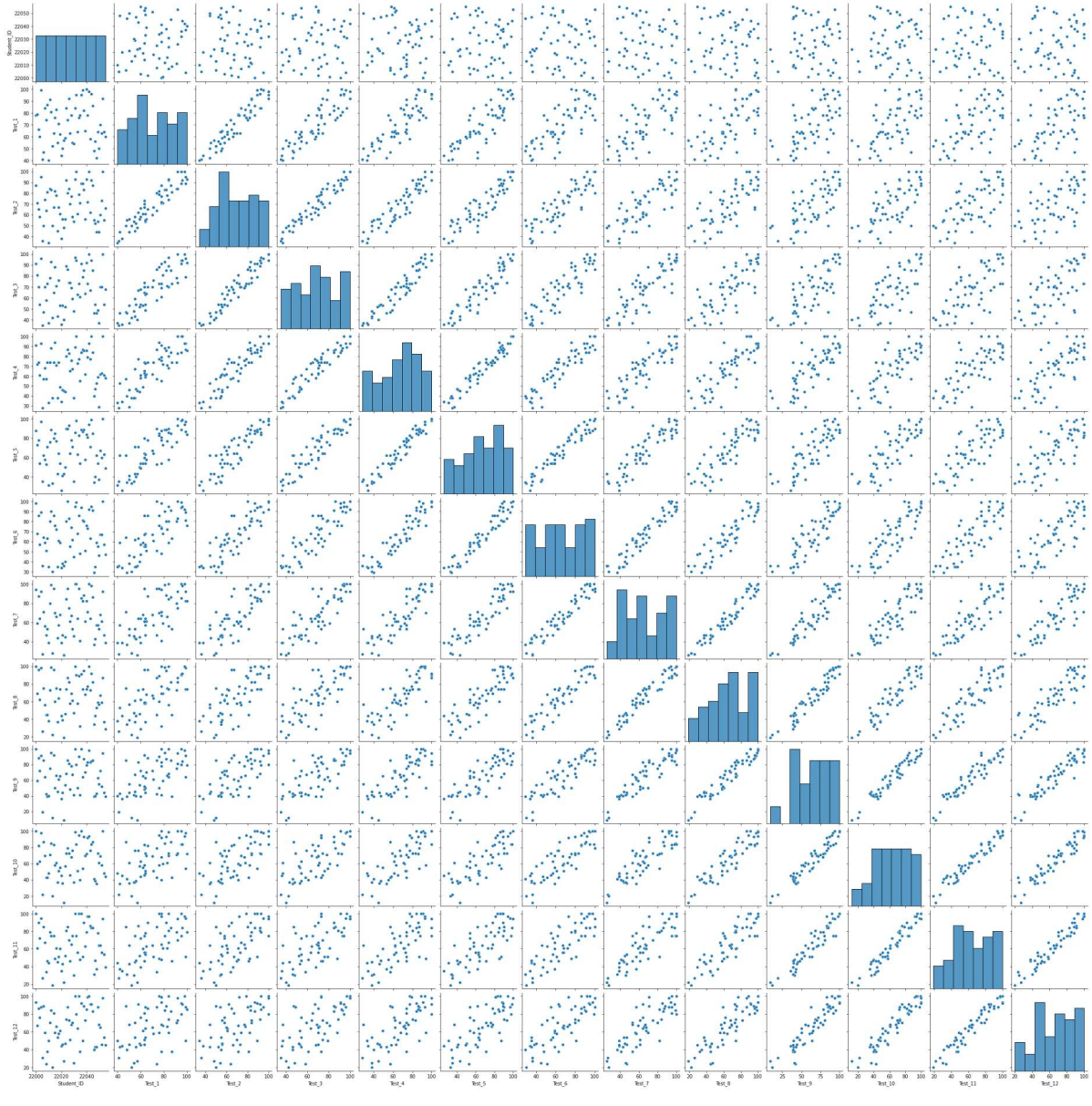
In [123]:
```
#to display columns name
data.columns
```

Out[123]:
```
Index(['Student_ID', 'Test_1', 'Test_2', 'Test_3', 'Test_4', 'Test_5',
       'Test_6', 'Test_7', 'Test_8', 'Test_9', 'Test_10', 'Test_11',
       'Test_12'],
      dtype='object')
```

In [124]:
```
data1=data[['Student_ID', 'Test_1', 'Test_2', 'Test_3', 'Test_4', 'Test_5',
       'Test_6', 'Test_7', 'Test_8', 'Test_9', 'Test_10', 'Test_11',
       'Test_12']]
```

In [125]:
```
sns.pairplot(data1)
```

Out[125]:
```
<seaborn.axisgrid.PairGrid at 0x2388fe0efd0>
```

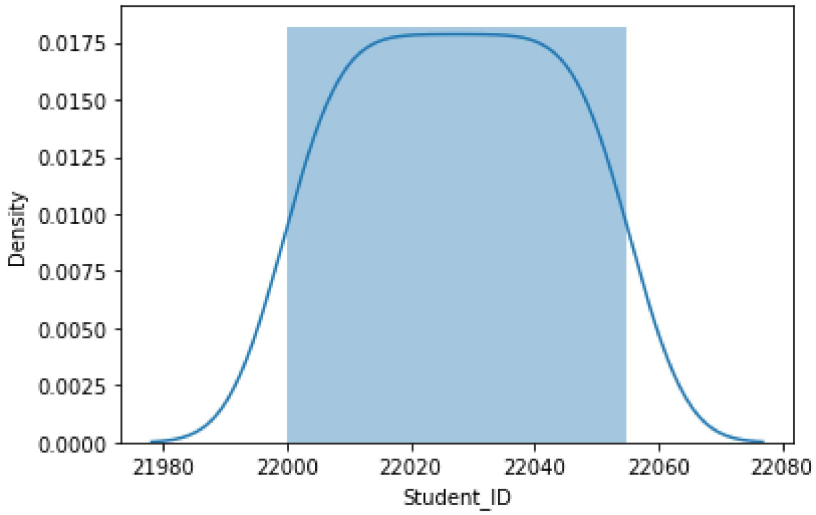


EDA and Visualization

In [126]:
```
sns.distplot(data['Student_ID'])
```

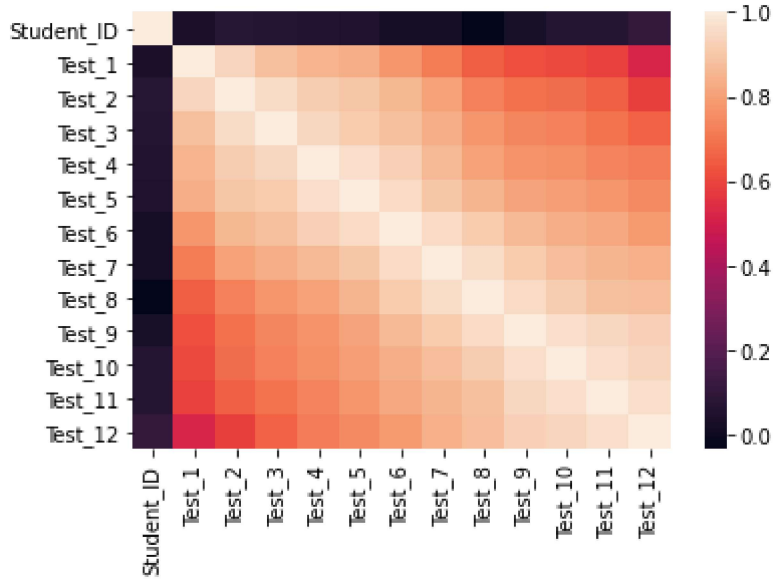
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning: `distplot` is a deprecated function and will be removed in a futu
re version. Please adapt your code to use either `displot` (a figure-level fu
nction with similar flexibility) or `histplot` (an axes-level function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

Out[126]: <AxesSubplot:xlabel='Student_ID', ylabel='Density'>



In [127]:
```
sns.heatmap(data1.corr())
```

Out[127]: <AxesSubplot:>



To train the model

we are going to train the linear regression model ;We need to split the two variable x and y where x in independent variable (input) and y is dependent of x(output) so we could ignore address columns as it is not requires for our model

In [129]:
```
x=data[['Student_ID', 'Test_1', 'Test_2', 'Test_3', 'Test_4', 'Test_5',
       'Test_6', 'Test_7', 'Test_8', 'Test_9', 'Test_10', 'Test_11']]
y=data1['Test_12']
```

In [130]:
```
#To split test and train data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.6)
```

In [131]:
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[131]:
```
LinearRegression()
```

In [132]:
```
lr.intercept_
```

Out[132]:
```
-1005.395718092095
```

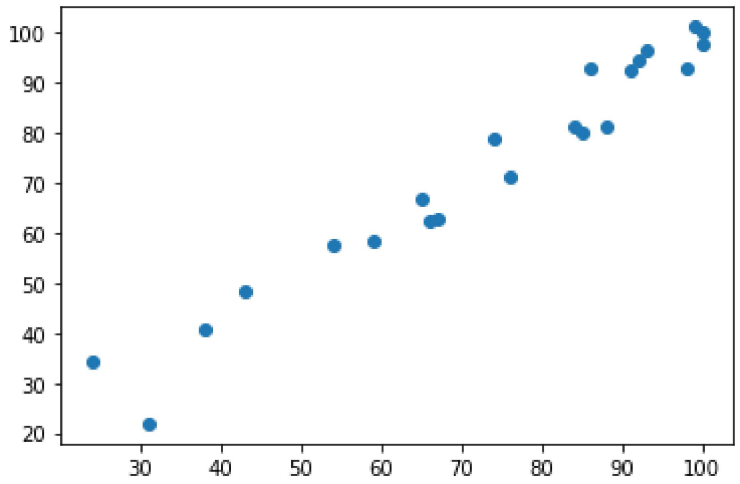
In [133]:
```
coeff = pd.DataFrame(lr.coef_,x.columns,columns=["Co-efficient"])
coeff
```

Out[133]:

	Co-efficient
Student_ID	0.045606
Test_1	0.149227
Test_2	-0.506677
Test_3	-0.141503
Test_4	0.489628
Test_5	0.292570
Test_6	-0.419304
Test_7	0.195901
Test_8	0.209811
Test_9	-0.132142
Test_10	0.031968
Test_11	0.852834

In [134]:
```
prediction = lr.predict(x_train)
plt.scatter(y_train,prediction)
```

Out[134]:
```
<matplotlib.collections.PathCollection at 0x23897e21550>
```



In [135]:
```
lr.score(x_test,y_test)
```

Out[135]:
```
0.9064609926038321
```

In [136]:
```
lr.score(x_train,y_train)
```

Out[136]:
```
0.9564154194757654
```

In [137]:
```
from sklearn.linear_model import Ridge,Lasso
```

In [138]:
```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
rr.score(x_test,y_test)
```

Out[138]:
```
0.911903098428319
```

In [139]:
```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
la.score(x_test,y_test)
```

Out[139]:
```
0.9268103105461197
```

In [ ]: