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(r"C:\Users\Admin\Downloads\15_Horse Racing Results.csv - 15_Horse Racing Results.CS
df
```

Out[2]:

| | Dato | Track | Race Number | Distance | Surface | Prize money | Starting position | Jockey | Jockey weight | Country | ... | TrainerName | Ra tin |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 03.09.2017 | Sha Tin | 10 | 1400 | Gress | 1310000 | 6 | K C Leung | 52 | Sverige | ... | CH Yip | 83, |
| **1** | 16.09.2017 | Sha Tin | 10 | 1400 | Gress | 1310000 | 14 | C Y Ho | 52 | Sverige | ... | CH Yip | 81, |
| **2** | 14.10.2017 | Sha Tin | 10 | 1400 | Gress | 1310000 | 8 | C Y Ho | 52 | Sverige | ... | CH Yip | 82, |
| **3** | 11.11.2017 | Sha Tin | 9 | 1600 | Gress | 1310000 | 13 | Brett Prebble | 54 | Sverige | ... | CH Yip | 96, |
| **4** | 26.11.2017 | Sha Tin | 9 | 1600 | Gress | 1310000 | 9 | C Y Ho | 52 | Sverige | ... | CH Yip | 94, |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **27003** | 14.06.2020 | Sha Tin | 11 | 1200 | Gress | 1450000 | 6 | A Hamelin | 59 | Australia | ... | WY So | 70, |
| **27004** | 21.06.2020 | Sha Tin | 2 | 1200 | Gress | 967000 | 7 | K C Leung | 57 | Australia | ... | KL Man | 69, |
| **27005** | 21.06.2020 | Sha Tin | 4 | 1200 | Gress | 967000 | 6 | Blake Shinn | 57 | Australia | ... | P O'Sullivan | 69, |
| **27006** | 21.06.2020 | Sha Tin | 5 | 1200 | Gress | 967000 | 14 | Joao Moreira | 57 | New Zealand | ... | AS Cruz | 70, |
| **27007** | 21.06.2020 | Sha Tin | 11 | 1200 | Gress | 1450000 | 7 | C Schofield | 55 | New Zealand | ... | WY So | 69, |

27008 rows × 21 columns

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27008 entries, 0 to 27007
Data columns (total 21 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Dato               27008 non-null  object
 1   Track              27008 non-null  object
 2   Race Number        27008 non-null  int64 
 3   Distance           27008 non-null  int64 
 4   Surface            27008 non-null  object
 5   Prize money        27008 non-null  int64 
 6   Starting position  27008 non-null  int64 
 7   Jockey             27008 non-null  object
 8   Jockey weight      27008 non-null  int64 
 9   Country            27008 non-null  object
 10  Horse age          27008 non-null  int64 
 11  TrainerName        27008 non-null  object
 12  Race time          27008 non-null  object
 13  Path               27008 non-null  int64 
 14  Final place        27008 non-null  int64 
 15  FGrating           27008 non-null  int64 
 16  Odds               27008 non-null  object
 17  RaceType           27008 non-null  object
 18  HorseId            27008 non-null  int64 
 19  JockeyId           27008 non-null  int64 
 20  TrainerID          27008 non-null  int64 
dtypes: int64(12), object(9)
memory usage: 4.3+ MB
```

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

Out[4]:

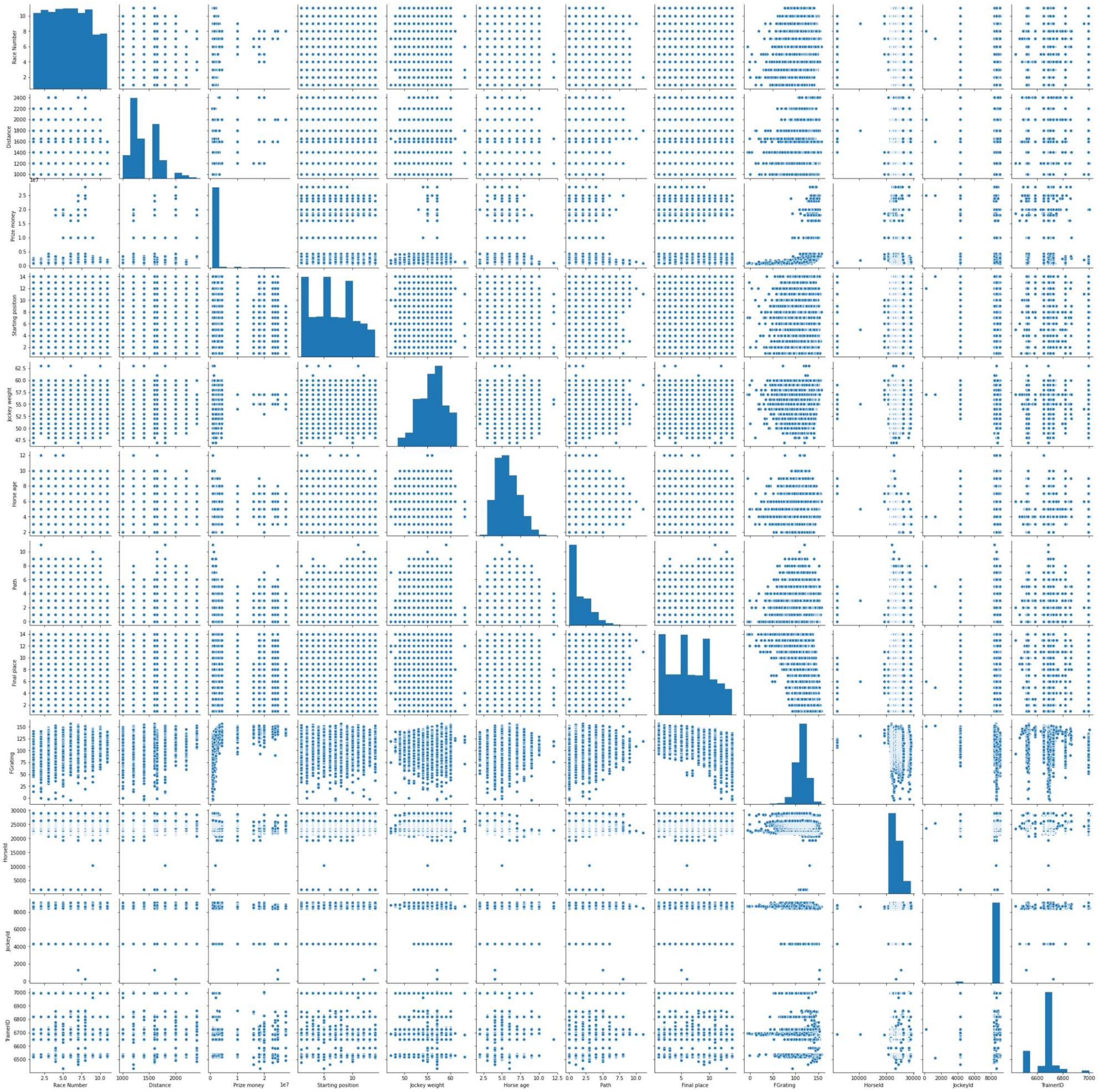
| | Race Number | Distance | Prize money | Starting position | Jockey weight | Horse age | Path | Final place |
|---|---|---|---|---|---|---|---|---|
| **count** | 27008.000000 | 27008.000000 | 2.700800e+04 | 27008.000000 | 27008.000000 | 27008.000000 | 27008.000000 | 27008.000000 |
| **mean** | 5.268624 | 1401.666173 | 1.479445e+06 | 6.741447 | 55.867373 | 5.246408 | 1.678021 | 6.685834 |
| **std** | 2.780088 | 276.065045 | 2.162109e+06 | 3.691071 | 2.737006 | 1.519880 | 1.631784 | 3.664551 |
| **min** | 1.000000 | 1000.000000 | 6.600000e+05 | 1.000000 | 47.000000 | 2.000000 | 0.000000 | 1.000000 |
| **25%** | 3.000000 | 1200.000000 | 9.200000e+05 | 4.000000 | 54.000000 | 4.000000 | 0.000000 | 4.000000 |
| **50%** | 5.000000 | 1400.000000 | 9.670000e+05 | 7.000000 | 56.000000 | 5.000000 | 1.000000 | 7.000000 |
| **75%** | 8.000000 | 1650.000000 | 1.450000e+06 | 10.000000 | 58.000000 | 6.000000 | 3.000000 | 10.000000 |
| **max** | 11.000000 | 2400.000000 | 2.800000e+07 | 14.000000 | 63.000000 | 12.000000 | 11.000000 | 14.000000 |

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Dato', 'Track', 'Race Number', 'Distance', 'Surface', 'Prize money',
       'Starting position', 'Jockey', 'Jockey weight', 'Country', 'Horse age',
       'TrainerName', 'Race time', 'Path', 'Final place', 'FGrating', 'Odds',
       'RaceType', 'HorseId', 'JockeyId', 'TrainerID'],
      dtype='object')
```

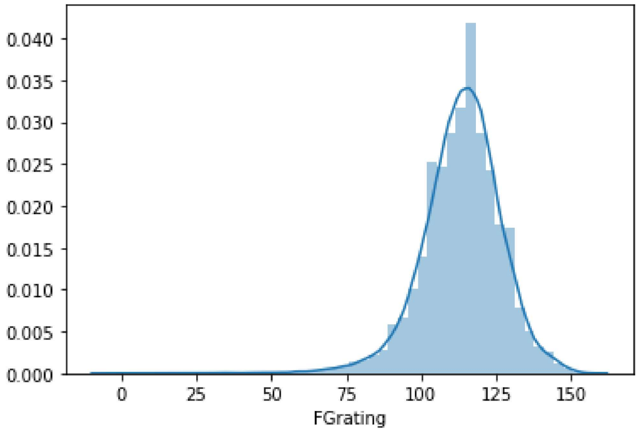
```
In [6]: sns.pairplot(df)
```

```
Out[6]: <seaborn.axisgrid.PairGrid at 0x1a7254e0340>
```



```
In [7]: sns.distplot(df['FGrating'])
```

```
Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x1a72fd433a0>
```



```
In [9]: df1=df[[ 'Race Number', 'Distance','Prize money',
               'Starting position','Jockey weight','Horse age','Path', 'Final place', 'FGrating']]
        df1
```

Out[9]:

| | Race Number | Distance | Prize money | Starting position | Jockey weight | Horse age | Path | Final place | FGrating |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 10 | 1400 | 1310000 | 6 | 52 | 7 | 2 | 9 | 110 |
| **1** | 10 | 1400 | 1310000 | 14 | 52 | 7 | 3 | 4 | 124 |
| **2** | 10 | 1400 | 1310000 | 8 | 52 | 7 | 1 | 6 | 118 |
| **3** | 9 | 1600 | 1310000 | 13 | 54 | 7 | 0 | 8 | 107 |
| **4** | 9 | 1600 | 1310000 | 9 | 52 | 7 | 0 | 3 | 123 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **27003** | 11 | 1200 | 1450000 | 6 | 59 | 3 | 1 | 9 | 104 |
| **27004** | 2 | 1200 | 967000 | 7 | 57 | 3 | 2 | 5 | 110 |
| **27005** | 4 | 1200 | 967000 | 6 | 57 | 3 | 0 | 3 | 114 |
| **27006** | 5 | 1200 | 967000 | 14 | 57 | 3 | 2 | 7 | 109 |
| **27007** | 11 | 1200 | 1450000 | 7 | 55 | 4 | 2 | 9 | 118 |

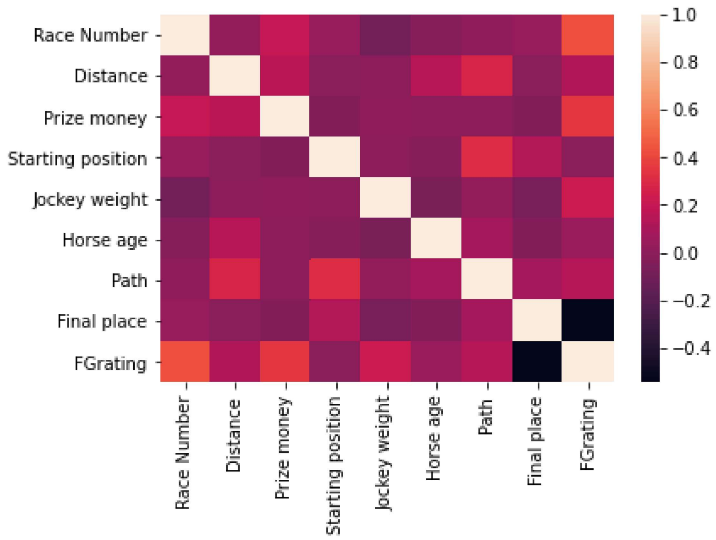
27008 rows × 9 columns

In [10]:

```
sns.heatmap(df1.corr())
```

Out[10]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a7310a87f0>
```



In [11]:

```
x=df1[['Race Number', 'Distance','Prize money',
       'Starting position','Jockey weight','Horse age','Path', 'Final place']]
y=df1['FGrating']
```

In [12]:

```
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 [13]:

```
from sklearn.linear_model import LinearRegression
lr= LinearRegression()
lr.fit(x_train,y_train)
```

Out[13]:

```
LinearRegression()
```

In [14]:

```
print(lr.intercept_)
```

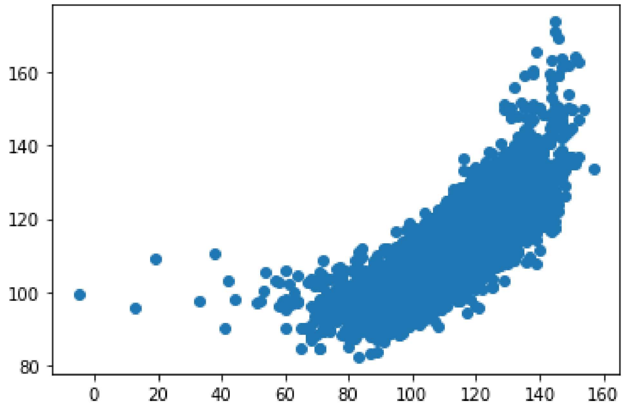
```
50.75252927688783
```

In [15]:

```
prediction= lr.predict(x_test)
plt.scatter(y_test,prediction)
```

Out[15]:

```
<matplotlib.collections.PathCollection at 0x1a7319c69a0>
```



In [16]:

```
print(lr.score(x_test,y_test))
```

```
0.6487572850780645
```

```
In [17]: print(lr.score(x_train,y_train))

         0.6344775719933136
```

```
In [18]: from sklearn.linear_model import Ridge,Lasso
         rr=Ridge(alpha=10)
         rr.fit(x_train,y_train)

Out[18]: Ridge(alpha=10)
```

```
In [19]: rr.score(x_test,y_test)

Out[19]: 0.6487577851117442
```

```
In [20]: la=Lasso(alpha=10)
         la.fit(x_train,y_train)

Out[20]: Lasso(alpha=10)
```

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

Out[21]: 0.4543705603483994
```

```
In [ ]:
```