In [141]:

```
import pandas as pd
import numpy as np

df = pd.read_csv('IPL_Data.csv')

# Displaying First 5 Rows
df.head()
```

Out[141]:

	Name	Team	Type	ValueinCR	Age	National Side	Batting Style	Bowling	MatchPlayed	N
0	Mayank Agarwal	PBKS	Batsman	12.00	31 Years, 0 Months, 28 Days	India	Right Handed	Off break	100.0	
1	Liam Livingstone	PBKS	All-Rounder	11.50	28 Years, 7 Months, 11 Days	England	Right Handed	Leg break	9.0	
2	Kagiso Rabada	PBKS	Bowler	9.25	26 Years, 9 Months, 22 Days	South Africa	Left Handed	Right-arm fast	50.0	
3	Shahrukh Khan	PBKS	All-Rounder	9.00	26 Years, 9 Months, 20 Days	India	Right Handed	Off break	11.0	
4	Shikhar Dhawan	PBKS	Batsman	8.25	36 Years, 3 Months, 10 Days	India	Left Handed	Off break	192.0	

5 rows × 23 columns

In [142]:

```
# Displaying Total Numbers of Rows and Columns
df.shape
```

Out[142]:

```
(237, 23)
```

In [143]:

```
# Displaying Summary of data
df.describe()
```

Out[143]:

	ValueinCR	MatchPlayed	NotOuts	RunsScored	100s	50s	4s
count	237.000000	162.000000	162.000000	153.000000	162.000000	162.000000	162.000000
mean	3.695781	50.043210	8.179012	792.287582	0.185185	4.148148	65.987654
std	4.238092	53.592359	11.182786	1334.058241	0.652013	8.865610	121.038723
min	0.200000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.400000	10.000000	1.000000	25.000000	0.000000	0.000000	1.000000
50%	1.900000	29.000000	4.000000	148.000000	0.000000	0.000000	8.500000
75%	6.500000	75.250000	11.000000	954.000000	0.000000	3.750000	79.000000
max	17.000000	220.000000	73.000000	6283.000000	5.000000	50.000000	654.000000

In [144]:

```
# Displaying Column and their Data Type
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 237 entries, 0 to 236
Data columns (total 23 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Name            237 non-null    object 
 1   Team            237 non-null    object 
 2   Type            237 non-null    object 
 3   ValueinCR       237 non-null    float64
 4   Age             231 non-null    object 
 5   National Side   231 non-null    object 
 6   Batting Style   227 non-null    object 
 7   Bowling         206 non-null    object 
 8   MatchPlayed     162 non-null    float64
 9   NotOuts         162 non-null    float64
 10  RunsScored      153 non-null    float64
 11  100s            162 non-null    float64
 12  50s             162 non-null    float64
 13  4s              162 non-null    float64
 14  6s              162 non-null    float64
 15  BattingS/R      162 non-null    float64
 16  CatchesTaken    145 non-null    float64
 17  StumpingsMade   145 non-null    float64
 18  Overs           122 non-null    float64
 19  Maidens         122 non-null    float64
 20  Wickets         122 non-null    float64
 21  EconomyRate     122 non-null    float64
 22  S/R             105 non-null    float64
dtypes: float64(16), object(7)
memory usage: 42.7+ KB
```

In [145]:

```
# Checking for missing values
df.isnull().sum()
```

Out[145]:

```
Name                0
Team                0
Type                0
ValueinCR           0
Age                 6
National Side       6
Batting Style      10
Bowling            31
MatchPlayed        75
NotOuts            75
RunsScored         84
100s               75
50s                75
4s                 75
6s                 75
BattingS/R         75
CatchesTaken       92
StumpingsMade      92
Overs             115
Maidens           115
Wickets           115
EconomyRate       115
S/R               132
dtype: int64
```

In [146]:

```
# Filling the Undefined Data
df["Age"].fillna("Unknown", inplace = True)
df["Batting Style"].fillna("None", inplace = True)
df["Bowling"].fillna("None", inplace = True)
df['MatchPlayed']= df['MatchPlayed'].replace(np.nan, 0)
df['NotOuts']= df['NotOuts'].replace(np.nan, 0)
df['RunsScored']= df['RunsScored'].replace(np.nan, 0)
df['100s']= df['100s'].replace(np.nan, 0)
df['50s']= df['50s'].replace(np.nan, 0)
df['4s']= df['4s'].replace(np.nan, 0)
df['6s']= df['6s'].replace(np.nan, 0)
df['BattingS/R']= df['BattingS/R'].replace(np.nan, 0)
df['CatchesTaken']= df['CatchesTaken'].replace(np.nan, 0)
df['StumpingsMade']= df['StumpingsMade'].replace(np.nan, 0)
df['Overs']= df['Overs'].replace(np.nan, 0)
df['Maidens']= df['Maidens'].replace(np.nan, 0)
df['Wickets']= df['Wickets'].replace(np.nan, 0)
df['EconomyRate']= df['EconomyRate'].replace(np.nan, 0)
df['S/R']= df['S/R'].replace(np.nan, 0)

#Dropping the rows
df = df.dropna(axis = 0, how ='any')

# After removing missing values
df.isnull().sum()
```

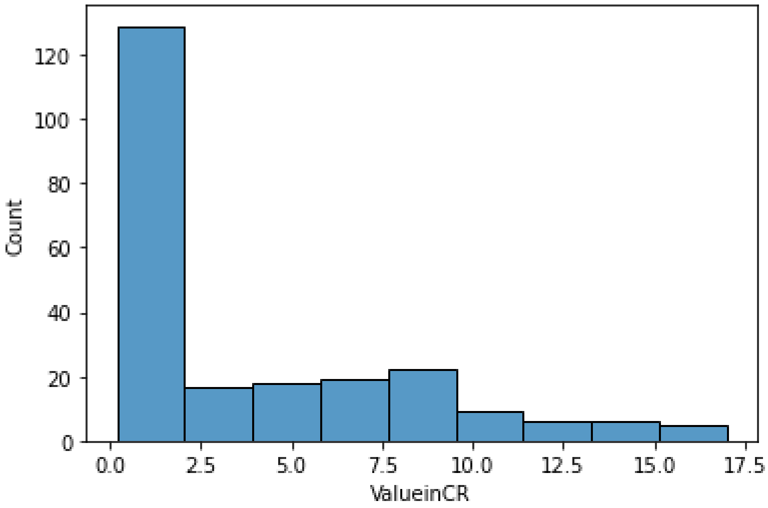
Out[146]:

```
Name             0
Team             0
Type             0
ValueinCR        0
Age              0
National Side    0
Batting Style    0
Bowling          0
MatchPlayed      0
NotOuts          0
RunsScored       0
100s             0
50s              0
4s               0
6s               0
BattingS/R       0
CatchesTaken     0
StumpingsMade    0
Overs            0
Maidens          0
Wickets          0
EconomyRate      0
S/R              0
dtype: int64
```

In [147]:

```
import seaborn as sns
import matplotlib.pyplot as plt

# Displaying Graph for Player Values in CR
sns.histplot(x='ValueinCR', data=df, )
plt.show()
```



In [148]:

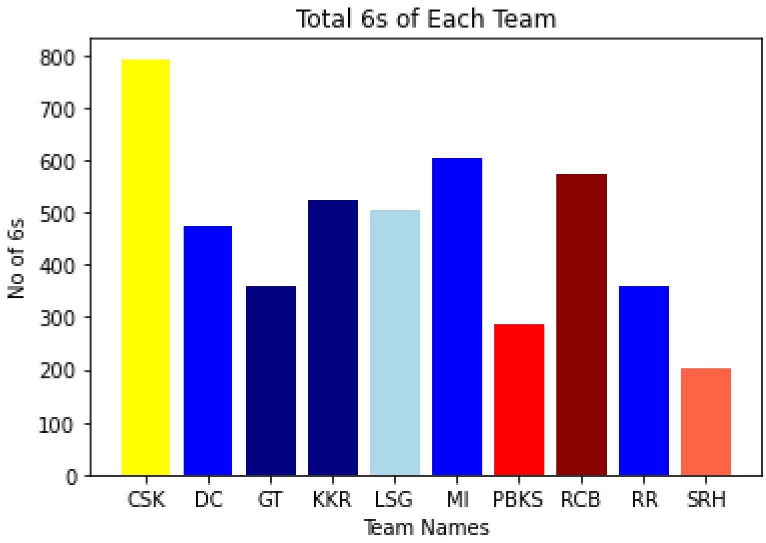
```
# Grouping the data on basis of team and calculating sum of each rows values for each team
gk = df.groupby('Team').sum()
gk
```

Out[148]:

Team	ValueinCR	MatchPlayed	NotOuts	RunsScored	100s	50s	4s	6s	BattingS/R
CSK	86.55	1134.0	246.0	19463.0	2.0	88.0	1573.0	793.0	1808.97
DC	86.40	814.0	114.0	13003.0	5.0	83.0	1224.0	475.0	1856.52
GT	80.85	762.0	158.0	9033.0	2.0	39.0	735.0	361.0	1725.39
KKR	80.95	858.0	112.0	12520.0	2.0	79.0	1144.0	524.0	2004.12
LSG	87.20	798.0	124.0	13443.0	4.0	75.0	1176.0	504.0	1555.41
MI	89.90	887.0	158.0	13147.0	1.0	78.0	1116.0	604.0	1509.10
PBKS	84.55	599.0	86.0	9746.0	4.0	62.0	1014.0	285.0	1415.51
RCB	88.20	907.0	137.0	16187.0	5.0	95.0	1425.0	575.0	1351.34
RR	88.05	830.0	96.0	9246.0	5.0	46.0	832.0	359.0	1598.91
SRH	89.90	518.0	94.0	5432.0	0.0	27.0	451.0	203.0	1770.62

In [149]:

```
# Displaying graph for 6s
left=[1,2,3,4,5,6,7,8,9,10]
lab=['CSK','DC','GT','KKR','LSG','MI','PBKS','RCB','RR','SRH']
plt.title('Total 6s of Each Team')
plt.xlabel('Team Names')
plt.ylabel('No of 6s')
plt.bar(left, gk['6s'], tick_label =lab, width = 0.8,
        color = ['yellow','blue','#000080','#000080','lightblue',
                 'blue','red','darkred','blue','tomato'])
plt.show()
```



In [150]:

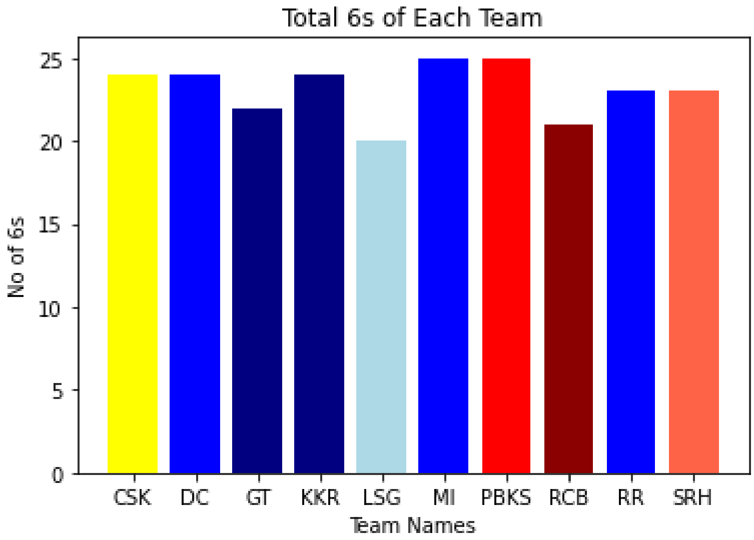
```
# Counting Total no of Players of each team
gk2=df.groupby('Team').count()
gk2['Name']
```

Out[150]:

```
Team
CSK     24
DC      24
GT      22
KKR     24
LSG     20
MI      25
PBKS    25
RCB     21
RR      23
SRH     23
Name: Name, dtype: int64
```

In [151]:

```
# Showing graph for Total no of Players of each team
left=[1,2,3,4,5,6,7,8,9,10]
lab=['CSK','DC','GT','KKR','LSG','MI','PBKS','RCB','RR','SRH']
plt.title('Total 6s of Each Team')
plt.xlabel('Team Names')
plt.ylabel('No of 6s')
plt.bar(left, gk2['Name'], tick_label =lab, width = 0.8,
        color = ['yellow','blue','#000080','#000080','lightblue',
                 'blue','red','darkred','blue','tomato'])
plt.show()
```



In [152]:

```
# Counting Total no of Players of each country
gk3=df.groupby('National Side').count()
gk3['Name']
```

Out[152]:

```
National Side
Afghanistan       4
Australia        13
Bangladesh        1
England          13
India           157
New Zealand      11
Singapore         1
South Africa     10
Sri Lanka         5
West Indies      16
Name: Name, dtype: int64
```

In [153]:

```
# Extracting dependent and independent variables from the given dataset
x= df.iloc[:, 9:].values
y= df.iloc[:, 8].values
```

In [154]:

```
# Splitting the dataset into training and test dataset
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.25, random_state=0)
```

In [155]:

```
# Fitting Linear Regression model to the training dataset
from sklearn.linear_model import LinearRegression
regressor= LinearRegression()
regressor.fit(x_train, y_train)
```

Out[155]:

```
LinearRegression()
```

In [156]:

```
#Prediction of Test and Training set result
y_pred= regressor.predict(x_test)
x_pred= regressor.predict(x_train)

df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
df
```

Out[156]:

	Actual	Predicted
0	3.0	11.765495
1	77.0	62.054032
2	150.0	141.481302
3	0.0	0.435475
4	0.0	0.435475
5	0.0	0.435475
6	26.0	32.650041
7	53.0	39.543418
8	0.0	0.435475
9	0.0	0.435475
10	0.0	0.435475
11	24.0	25.260178
12	0.0	0.435475
13	0.0	0.435475
14	21.0	29.561681
15	23.0	24.971083
16	207.0	221.801123
17	19.0	16.447275
18	0.0	0.435475
19	28.0	34.407511
20	105.0	86.428629
21	0.0	0.435475
22	24.0	28.315753
23	35.0	46.770899
24	0.0	0.435475
25	0.0	0.435475
26	6.0	13.005354
27	100.0	126.282952
28	11.0	10.410902
29	58.0	58.243182
30	3.0	8.412846

	Actual	Predicted
31	24.0	26.209415
32	11.0	16.764163
33	23.0	22.188844
34	76.0	85.522512
35	4.0	10.116032
36	5.0	10.521139
37	1.0	3.534541
38	0.0	0.435475
39	167.0	137.430840
40	42.0	35.063693
41	72.0	47.480387
42	61.0	57.958699
43	6.0	11.018318
44	56.0	57.924908
45	115.0	99.466799
46	0.0	0.435475
47	2.0	3.174618
48	3.0	4.414933
49	22.0	24.036781
50	0.0	0.435475
51	0.0	0.435475
52	121.0	94.257712
53	151.0	189.481753
54	77.0	58.256292
55	151.0	147.123079
56	99.0	95.348333
57	0.0	0.435475

In [157]:

```
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 6.631552426261754
Mean Squared Error: 117.38676058344303
Root Mean Squared Error: 10.834517090458764
```

In [ ]: