```
# load file in google colab
from google.colab import files
uploaded = files.upload()
```

Choose Files matches.csv

- **matches.csv**(application/vnd.ms-excel) - 140113 bytes, last modified: 10/1/2019 - 100% done

```
Saving matches.csv to matches.csv
```

```
# upload second file in google colab
uploaded2=files.upload()
```

Choose Files deliveries.csv

- **deliveries.csv**(application/vnd.ms-excel) - 18235327 bytes, last modified: 10/1/2019 - 100% done

```
Saving deliveries.csv to deliveries.csv
```

```
import pandas as pd
import io
# read matches.csv
match_df=pd.read_csv(io.BytesIO(uploaded['matches.csv']))
```

```
# read deliveries.csv
deliveries_df=pd.read_csv(io.BytesIO(uploaded2['deliveries.csv']))
```

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

```
match_df.head()
```

	id	season	city	date	team1	team2	toss_winner	toss_decision	resu
0	1	2017	Hyderabad	2017-04-05	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	norr

```
deliveries_df.head()
```

	match_id	inning	batting_team	bowling_team	over	ball	batsman	non_striker	bowle
0	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	1	DA Warner	S Dhawan	T Mil
1	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	2	DA Warner	S Dhawan	T Mil
2	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	3	DA Warner	S Dhawan	T Mil
3	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	4	DA Warner	S Dhawan	T Mil
4	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	5	DA Warner	S Dhawan	T Mil

```
# basic info
match_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 756 entries, 0 to 755
Data columns (total 18 columns):
 #   Column           Non-Null Count  Dtype
```

```
#basic info
deliveries_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 179078 entries, 0 to 179077
Data columns (total 21 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   match_id          179078 non-null  int64
 1   inning            179078 non-null  int64
 2   batting_team      179078 non-null  object
 3   bowling_team      179078 non-null  object
 4   over              179078 non-null  int64
 5   ball              179078 non-null  int64
 6   batsman           179078 non-null  object
 7   non_striker       179078 non-null  object
 8   bowler            179078 non-null  object
 9   is_super_over     179078 non-null  int64
 10  wide_runs         179078 non-null  int64
 11  bye_runs          179078 non-null  int64
 12  legbye_runs       179078 non-null  int64
 13  noball_runs       179078 non-null  int64
 14  penalty_runs      179078 non-null  int64
 15  batsman_runs      179078 non-null  int64
 16  extra_runs        179078 non-null  int64
 17  total_runs        179078 non-null  int64
 18  player_dismissed  8834 non-null    object
 19  dismissal_kind    8834 non-null    object
 20  fielder           6448 non-null    object
dtypes: int64(13), object(8)
memory usage: 28.7+ MB
```

```
# basic statistical details
match_df.describe()
```

```
# basic statistical details
deliveries_df.describe()
```

	match_id	inning	over	ball	is_super_over	wid
count	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.
mean	1802.252957	1.482952	10.162488	3.615587	0.000452	0.
std	3472.322805	0.502074	5.677684	1.806966	0.021263	0
min	1.000000	1.000000	1.000000	1.000000	0.000000	0.
25%	190.000000	1.000000	5.000000	2.000000	0.000000	0.
50%	379.000000	1.000000	10.000000	4.000000	0.000000	0.
75%	567.000000	2.000000	15.000000	5.000000	0.000000	0.
max	11415.000000	5.000000	20.000000	9.000000	1.000000	5.

```
# null value
match_df.isnull().sum()
```

```
id                   0
season               0
city                 7
date                 0
team1                0
team2                0
toss_winner          0
toss_decision        0
result               0
dl_applied           0
winner               4
win_by_runs          0
win_by_wickets       0
player_of_match      4
venue                0
umpire1              2
umpire2              2
umpire3            637
dtype: int64
```

```
a=(match_df['umpire3'].isnull().sum()/match_df.shape[0])*100
print('percentage of null value in umpire3 column',a)
```

```
percentage of null value in umpire3 column 84.25925925925925
```

Around 84% null values are there for the umpire3 column.So we should drop this column

```
# drop umpire3 column
match_df.drop(['umpire3'],axis=1,inplace=True)


#fill null value in city,player_of_match,umpire1,umpire2,winner column by mode value
match_df['city'].fillna(match_df['city'].mode()[0],inplace=True)
match_df['player_of_match'].fillna(match_df['player_of_match'].mode()[0],inplace=True)
match_df['umpire1'].fillna('unknown',inplace=True)
match_df['umpire2'].fillna('unknown',inplace=True)
match_df['winner'].fillna('winner',inplace=True)


# null value
deliveries_df.isnull().sum()
```

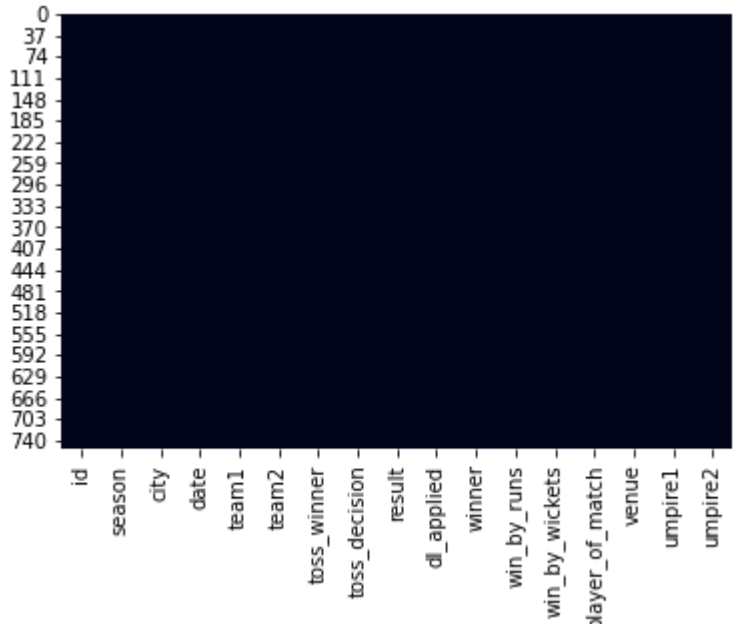
```
match_id                0
inning                  0
batting_team            0
bowling_team            0
over                    0
ball                    0
batsman                 0
non_striker             0
bowler                  0
is_super_over           0
wide_runs               0
bye_runs                0
legbye_runs             0
noball_runs             0
penalty_runs            0
batsman_runs            0
extra_runs              0
total_runs              0
player_dismissed    170244
dismissal_kind      170244
fielder             172630
dtype: int64
```

```
x=(deliveries_df['player_dismissed'].isnull().sum()/deliveries_df.shape[0])*100
y=(deliveries_df['dismissal_kind'].isnull().sum()/deliveries_df.shape[0])*100
z=(deliveries_df['fielder'].isnull().sum()/deliveries_df.shape[0])*100
print('percentage of null value in player_dismissed column',x)
print('percentage of null value in dismissal_kind column',y)
print('percentage of null value in fielder column',z)
```

```
percentage of null value in player_dismissed column 95.06695406470924
percentage of null value in dismissal_kind column 95.06695406470924
percentage of null value in fielder column 96.3993343682641
```

```
# checking null values after dropping column
sns.heatmap(match_df.isnull(),cbar=False)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6e4f809a58>
```



From the heatmap we can see that we do not have any null values

```
# unique team in team1 column
match_df['team1'].nunique()
```

```
15
```

```
# unique team in team2 column
match_df['team2'].nunique()
```

```
15
```

```
# count in team1
match_df['team1'].value_counts()
```

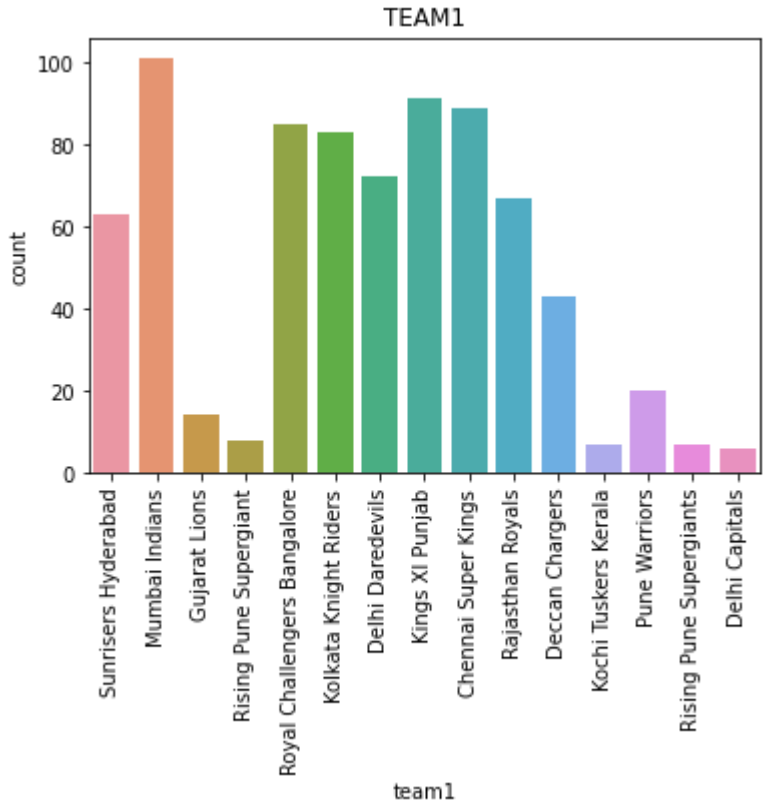
```
Mumbai Indians                101
Kings XI Punjab                91
Chennai Super Kings            89
Royal Challengers Bangalore    85
Kolkata Knight Riders          83
Delhi Daredevils               72
Rajasthan Royals               67
Sunrisers Hyderabad            63
Deccan Chargers                43
Pune Warriors                  20
Gujarat Lions                  14
Rising Pune Supergiant          8
Kochi Tuskers Kerala            7
Rising Pune Supergiants         7
Delhi Capitals                  6
Name: team1, dtype: int64
```

```
# count in team2
match_df['team2'].value_counts()
```

```
Kolkata Knight Riders          95
Royal Challengers Bangalore    95
Delhi Daredevils               89
Mumbai Indians                 86
Kings XI Punjab                85
Rajasthan Royals               80
Chennai Super Kings            75
Sunrisers Hyderabad            45
Deccan Chargers                32
Pune Warriors                  26
Gujarat Lions                  16
Delhi Capitals                 10
Rising Pune Supergiant          8
Kochi Tuskers Kerala            7
Rising Pune Supergiants         7
Name: team2, dtype: int64
```

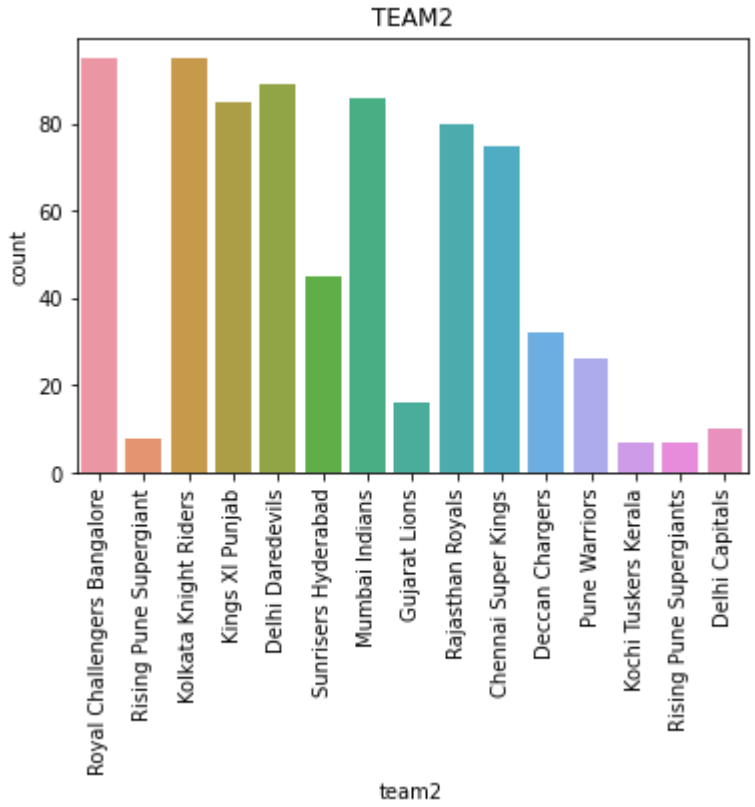
```
p=sns.countplot(x='team1',data=match_df)
p.set_xticklabels(p.get_xticklabels(),rotation=90)
plt.title('TEAM1')
```

```
Text(0.5, 1.0, 'TEAM1')
```




```
plot_c=sns.countplot(x='team2',data=match_df)
plot_c.set_xticklabels(plot_c.get_xticklabels(),rotation=90)
plt.title('TEAM2')
```

```
Text(0.5, 1.0, 'TEAM2')
```



Ques 1.Finding winner

```
win_team=sns.countplot(x='winner',data=match_df)
win_team.set_xticklabels(win_team.get_xticklabels(),rotation=90)
```

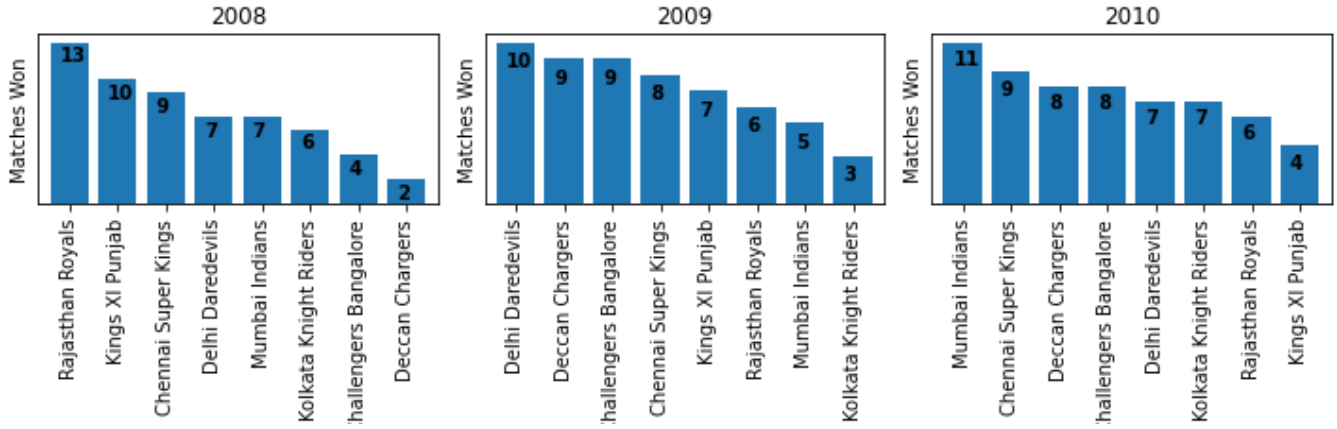
```
 [Text(0, 0, 'Sunrisers Hyderabad'),
  Text(0, 0, 'Rising Pune Supergiant'),
  Text(0, 0, 'Kolkata Knight Riders'),
  Text(0, 0, 'Kings XI Punjab'),
  Text(0, 0, 'Royal Challengers Bangalore'),
  Text(0, 0, 'Mumbai Indians'),
  Text(0, 0, 'Delhi Daredevils'),
  Text(0, 0, 'Gujarat Lions'),
  Text(0, 0, 'Chennai Super Kings'),
  Text(0, 0, 'Rajasthan Royals'),
  Text(0, 0, 'Deccan Chargers'),
  Text(0, 0, 'Pune Warriors'),
  Text(0, 0, 'Kochi Tuskers Kerala'),
  Text(0, 0, 'winner'),
  Text(0, 0, 'Rising Pune Supergiants'),
  Text(0, 0, 'Delhi Capitals')]
```

```
max_winner = match_df.groupby('season')['winner'].value_counts()
max_winner
```

```
season  winner
2008    Rajasthan Royals              13
        Kings XI Punjab               10
        Chennai Super Kings            9
        Delhi Daredevils               7
        Mumbai Indians                 7
                                      ..
2019    Kolkata Knight Riders          6
        Sunrisers Hyderabad            6
        Rajasthan Royals               5
        Royal Challengers Bangalore    5
        winner                         1
Name: winner, Length: 103, dtype: int64
```

```
groups = max_winner.groupby('season')
fig = plt.figure()
count = 1

for year, group in groups:
    ax = fig.add_subplot(4,3,count)
    ax.set_title(year)
    ax = group[year].plot.bar(figsize = (10,15), width = 0.8)

    count+=1;

    plt.xlabel('')
    plt.yticks([])
    plt.ylabel('Matches Won')

    total_of_matches = []
    for i in ax.patches:
        total_of_matches.append(i.get_height())
    total = sum(total_of_matches)
```

```
        for i in ax.patches:
            ax.text(i.get_x()+0.2, i.get_height()-1.5,s= i.get_height(),color="black",fontweight=
    plt.tight_layout()
    plt.show()
```



From above graph checking for last 2 year match MI won 8.5 matches on an average and RR won 6 matches on an average

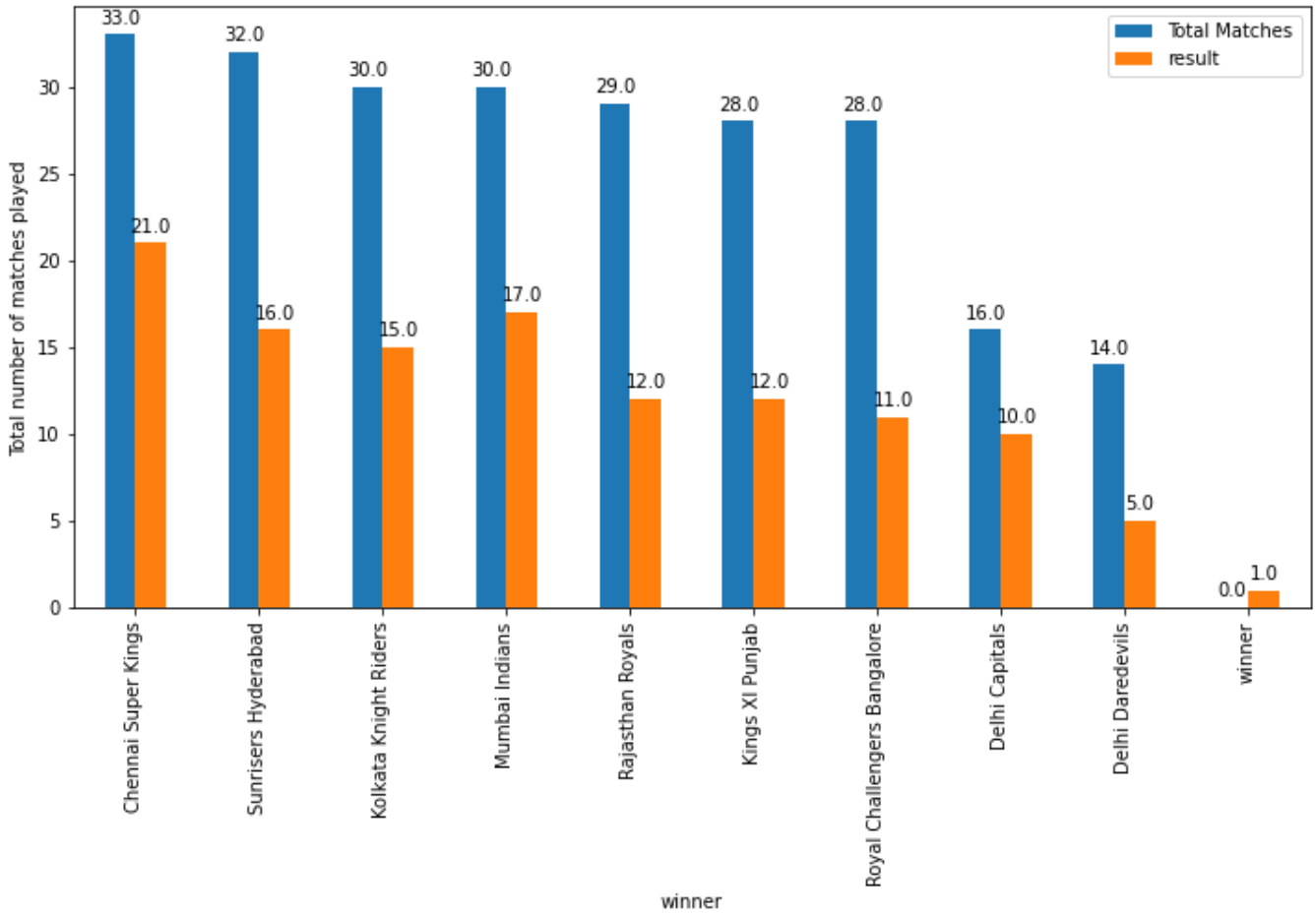
```
# taking match dataframe for last 2 years
match_18_19=match_df[match_df['season']>2017]

# first 4 rows
match_18_19.head(4)
```

	id	season	city	date	team1	team2	toss_winner	toss_decisior
636	7894	2018	Mumbai	07/04/18	Mumbai Indians	Chennai Super Kings	Chennai Super Kings	fielc
637	7895	2018	Mohali	08/04/18	Delhi Daredevils	Kings XI Punjab	Kings XI Punjab	fielc
638	7896	2018	Kolkata	08/04/18	Royal Challengers Bangalore	Kolkata Knight Riders	Kolkata Knight Riders	fielc
639	7897	2018	Hyderabad	09/04/18	Rajasthan Royals	Sunrisers Hyderabad	Sunrisers Hyderabad	fielc

```
matches_won = match_18_19.groupby('winner').count()
total_matches = match_18_19['team1'].value_counts()+ match_18_19['team2'].value_counts()
matches_won['Total Matches'] = total_matches
win_df = matches_won[["Total Matches","result"]]
```

```
graph=win_df[['Total Matches','result']].sort_values('Total Matches',ascending=False).plot.ba
```

```
    plt.ylabel('Total number of matches played')
    for p in graph.patches:
        graph.annotate(format(p.get_height(), '.1f'),
                       (p.get_x() + p.get_width() / 2., p.get_height()),
                       ha = 'center', va = 'center',
                       xytext = (0, 9),
                       textcoords = 'offset points')
```




```
    # 2018-2019 data
    # selecting MI and RR as team1 and team2 respectively
    MI_RR = match_18_19[(match_18_19['team1']=='Mumbai Indians') & (match_18_19['team2']=='Rajast

    # selecting RR and MI as team1 and team2 respectively
    RR_MI = match_18_19[(match_18_19['team1']=='Rajasthan Royals') & (match_18_19['team2']=='Mumb

    # selecting when MI won
    MI_won1=MI_RR[MI_RR['winner']=='Mumbai Indians']
    MI_won2=RR_MI[RR_MI['winner']=='Mumbai Indians']

    # selecting when RR won
    RR_won1=MI_RR[MI_RR['winner']=='Rajasthan Royals']
    RR_won2=RR_MI[RR_MI['winner']=='Rajasthan Royals']
```

```
print('Total no of matches between MI and RR are',len(MI_RR)+len(RR_MI))
print('Total no of matches MI won against RR are',len(MI_won1)+len(MI_won2))
print('Total no of matches RR won against MI are',len(RR_won1)+len(RR_won2))
```

```
Total no of matches between MI and RR are 4
Total no of matches MI won against RR are 0
Total no of matches RR won against MI are 4
```

```
MI_RR[['team1','team2','winner']]
```

	team1	team2	winner
656	Mumbai Indians	Rajasthan Royals	Rajasthan Royals
682	Mumbai Indians	Rajasthan Royals	Rajasthan Royals
722	Mumbai Indians	Rajasthan Royals	Rajasthan Royals
731	Mumbai Indians	Rajasthan Royals	Rajasthan Royals

Ans and Explanation ans

1.From above graph we came to know that MI won 56.66% matches And RR won 41.3% matches for last 2 years

3.In last 2 year matches, MI won 8.5 matches on an average and RR won 6 matches

4.MI won 0 match against RR and RR won 4 match against MI

So from all above analysis we came to know that MI won 0 matches against RR but MI won more than 55% matches and currently MI is in good form and probably win the match against RR

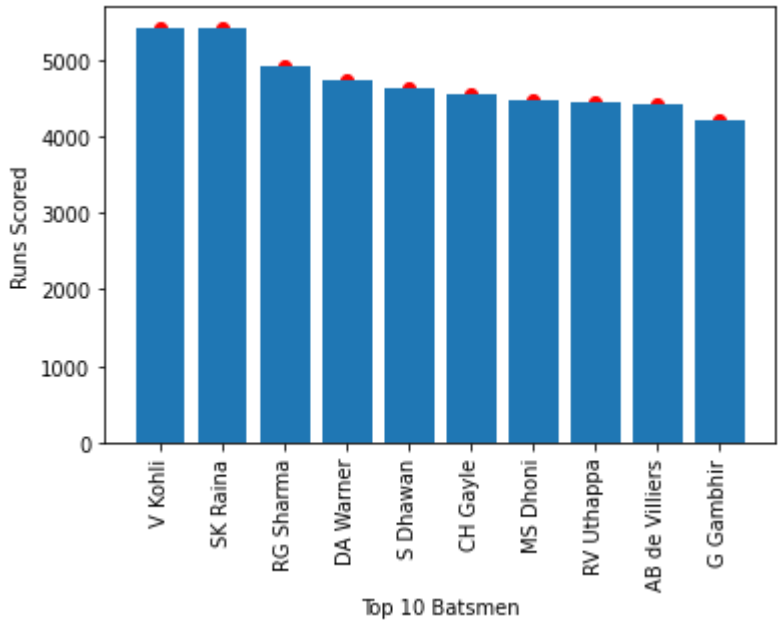
Ans 1:MI will win the match

Ques2.BIS?

```
import numpy as np

# checking Warner in top 10 batsman
batting_tot=deliveries_df.groupby('batsman').apply(lambda x:np.sum(x['batsman_runs'])).reset_
batting_sorted=batting_tot.sort_values(by='Runs',ascending=False)
top_batsmen=batting_sorted[:10]
plt.bar(top_batsmen['batsman'],top_batsmen['Runs'])
plt.scatter(top_batsmen['batsman'],top_batsmen['Runs'],color='r')
plt.xticks(rotation=90)
plt.xlabel('Top 10 Batsmen',size=10)
```

```
plt.ylabel('Runs Scored',size=10)
```

```
Text(0, 0.5, 'Runs Scored')
```




```
# taking MI and RR ipl 2020 batsman
batsman_20 = deliveries_df[(deliveries_df['batsman']=='RG Sharma') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='SA Yadav')  | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='SS Tiwary') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='Q de Kock') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='HH Pandya') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='KH Pandya') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='M Vohra') | (deliveries_df['batsman']=
                           (deliveries_df['batsman']=='S Singh') | (deliveries_df['batsman']=
                           (deliveries_df['batsman']=='DA Miller') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='T Curran') | (deliveries_df['batsman']
                           (deliveries_df['batsman']=='S Gopal') | (deliveries_df['batsman']=
                           (deliveries_df['batsman']=='SV Samson')]


# boundary runs
boundaryrun_given = batsman_20[(batsman_20['total_runs']==4) | (batsman_20['total_runs']==6)]

# runs = 1 or 2 or 3
lessrun_given = batsman_20[(batsman_20['total_runs']==1) | (batsman_20['total_runs']==2) | (b


list_batsman1 = ['RG Sharma' , 'S Rutherford' , 'SA Yadav' , 'CA Lynn' , 'SS Tiwary' ,'Ishan
                 'Q de Kock' , 'AP Tare' , 'HH Pandya' , 'KA Pollard', 'KH Pandya' , 'SPD Smit
                  'S Singh' , 'R Parag' , 'JC Buttler' , 'DA Miller' , 'T Curran' , 'BA Stoke
                 'S Gopal' , 'SV Samson' , 'M Lomror' ]


for i in list_batsman1:
    x = boundaryrun_given[boundaryrun_given['batsman']==i]
    y = lessrun_given[lessrun_given['batsman']==i]
```

```
    w = batsman_20[batsman_20['batsman']==i]
    numerator = sum(w['total_runs']) + sum(x['total_runs'])
    denominator = len(batsman_20)
    bis = numerator/denominator
    print('BIS of' , i , 'is' ,bis)
```

```
 BIS of RG Sharma is 0.3530955346023839
 BIS of S Rutherford is 0.006271126134139833
 BIS of SA Yadav is 0.04363102650773883
 BIS of CA Lynn is 0.09998220957125066
 BIS of SS Tiwary is 0.08886319160291763
 BIS of Ishan Kishan is 0.05341576231987191
 BIS of Q de Kock is 0.112479985767657
 BIS of AP Tare is 0.02646326276463263
 BIS of HH Pandya is 0.0849492972780644
 BIS of KA Pollard is 0.20992705924212773
 BIS of KH Pandya is 0.06760362924746487
 BIS of SPD Smith is 0.13996619818537626
 BIS of M Vohra is 0.07547589396904465
 BIS of RV Uthappa is 0.32961216865326454
 BIS of S Singh is 0.0012453300124533001
 BIS of R Parag is 0.012453300124533
 BIS of JC Buttler is 0.10838818715531044
 BIS of DA Miller is 0.13427326098558975
 BIS of T Curran is 0.0016456146593132894
 BIS of BA Stokes is 0.043986835082725495
 BIS of S Gopal is 0.008761786159046433
 BIS of SV Samson is 0.15668920120974916
 BIS of M Lomror is 0.001779042874933286
```

Ques2.Ans and Explanation

We can see that the Kock have very good BIL that is 0.11 which is in top list and also Smith also have 0.13.So we choose first option

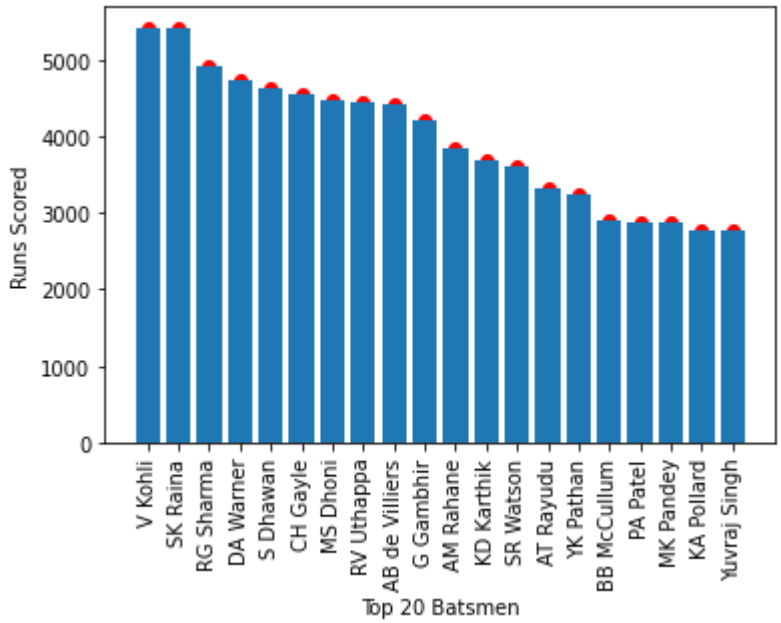
Ans2.Quinton De Kock or Steve Smith

Ques3.Ratio for batsman

```
import numpy as np

# top 20 batsman
batting_tot=deliveries_df.groupby('batsman').apply(lambda x:np.sum(x['batsman_runs'])).reset_
batting_sorted=batting_tot.sort_values(by='Runs',ascending=False)
top_batsmen=batting_sorted[:20]
print('The Top 20 Batsmen in the Tournament are:\n',top_batsmen)
plt.bar(top_batsmen['batsman'],top_batsmen['Runs'])
```

```
plt.scatter(top_batsmen['batsman'],top_batsmen['Runs'],color='r')
plt.xticks(rotation=90)
plt.xlabel('Top 20 Batsmen',size=10)
plt.ylabel('Runs Scored',size=10)
```

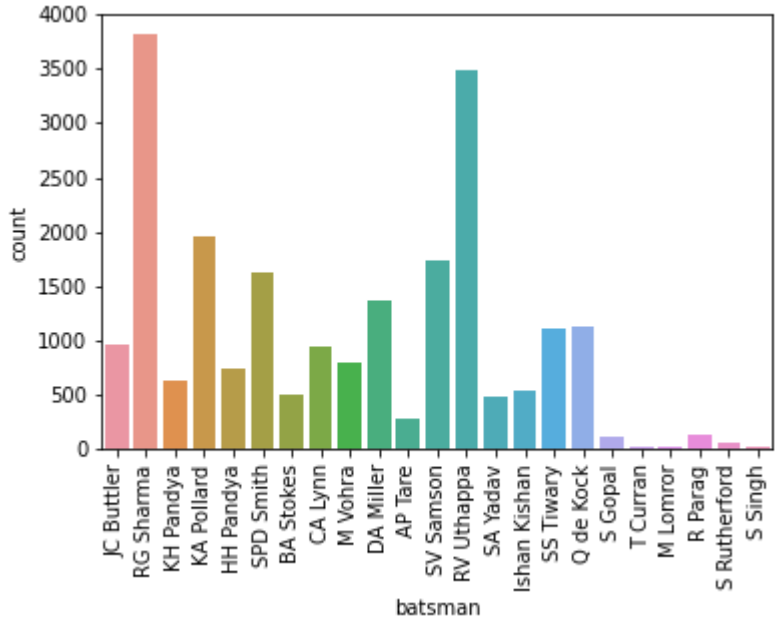
```
The Top 20 Batsmen in the Tournament are:
             batsman  Runs
486         V Kohli  5434
428        SK Raina  5415
367       RG Sharma  4914
112       DA Warner  4741
392        S Dhawan  4632
92         CH Gayle  4560
290        MS Dhoni  4477
384      RV Uthappa  4446
26   AB de Villiers  4428
147       G Gambhir  4223
42         AM Rahane  3850
218      KD Karthik  3688
444       SR Watson  3614
53         AT Rayudu  3326
509       YK Pathan  3241
72      BB McCullum  2893
329        PA Patel  2874
280       MK Pandey  2872
213      KA Pollard  2784
514    Yuvraj Singh  2765
Text(0, 0.5, 'Runs Scored')
```




```
# taking MI and RR ipl 2020 batsman
batsman_20 = deliveries_df[(deliveries_df['batsman']=='RG Sharma') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='SA Yadav')  | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='SS Tiwary') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='Q de Kock') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='HH Pandya') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='KH Pandya') | (deliveries_df['batsman'
                           (deliveries_df['batsman']=='M Vohra') | (deliveries_df['batsman']=
```

```
                        (deliveries_df['batsman']=='S Singh') | (deliveries_df['batsman']=
                        (deliveries_df['batsman']=='DA Miller') | (deliveries_df['batsman'
                        (deliveries_df['batsman']=='T Curran') | (deliveries_df['batsman']
                        (deliveries_df['batsman']=='S Gopal') | (deliveries_df['batsman']=
                        (deliveries_df['batsman']=='SV Samson')]

# selecting boundary runs
boundary_score = batsman_20[(batsman_20['total_runs']==4) | (batsman_20['total_runs']==6)]


batsman_team=sns.countplot(x='batsman',data=batsman_20)
batsman_team.set_xticklabels(batsman_team.get_xticklabels(),rotation=90)
```

```
[Text(0, 0, 'JC Buttler'),
 Text(0, 0, 'RG Sharma'),
 Text(0, 0, 'KH Pandya'),
 Text(0, 0, 'KA Pollard'),
 Text(0, 0, 'HH Pandya'),
 Text(0, 0, 'SPD Smith'),
 Text(0, 0, 'BA Stokes'),
 Text(0, 0, 'CA Lynn'),
 Text(0, 0, 'M Vohra'),
 Text(0, 0, 'DA Miller'),
 Text(0, 0, 'AP Tare'),
 Text(0, 0, 'SV Samson'),
 Text(0, 0, 'RV Uthappa'),
 Text(0, 0, 'SA Yadav'),
 Text(0, 0, 'Ishan Kishan'),
 Text(0, 0, 'SS Tiwary'),
 Text(0, 0, 'Q de Kock'),
 Text(0, 0, 'S Gopal'),
 Text(0, 0, 'T Curran'),
 Text(0, 0, 'M Lomror'),
 Text(0, 0, 'R Parag'),
 Text(0, 0, 'S Rutherford'),
 Text(0, 0, 'S Singh')]
```



```
list_batsman = ['RG Sharma' , 'S Rutherford' , 'SA Yadav' , 'CA Lynn' , 'SS Tiwary' ,'Ishan K
                'Q de Kock' , 'AP Tare' , 'HH Pandya' , 'KA Pollard', 'KH Pandya' , 'SPD Smit
                 'S Singh' , 'R Parag' , 'JC Buttler' , 'DA Miller' , 'T Curran' , 'BA Stoke
                'S Gopal' , 'SV Samson' , 'M Lomror' ]
for i in list_batsman:
    a = boundary_score[boundary_score['batsman']==i]
    b = batsman_20[batsman_20['batsman']==i]
    ratio_runs = sum(a['total_runs'])/sum(b['total_runs'])
    print('Ratio of ' , i , 'is' ,ratio_runs)
```

```
 Ratio of  RG Sharma is 0.5699031046074748
 Ratio of  S Rutherford is 0.5494505494505495
 Ratio of  SA Yadav is 0.5256609642301711
 Ratio of  CA Lynn is 0.6505139500734214
 Ratio of  SS Tiwary is 0.47345132743362833
 Ratio of  Ishan Kishan is 0.6207827260458839
 Ratio of  Q de Kock is 0.622193713919179
 Ratio of  AP Tare is 0.6301369863013698
 Ratio of  HH Pandya is 0.5916666666666667
 Ratio of  KA Pollard is 0.6054421768707483
 Ratio of  KH Pandya is 0.5833333333333334
 Ratio of  SPD Smith is 0.47538677918424754
 Ratio of  M Vohra is 0.5845004668534081
 Ratio of  RV Uthappa is 0.5758026791409738
 Ratio of  S Singh is 0.5555555555555556
 Ratio of  R Parag is 0.5730337078651685
 Ratio of  JC Buttler is 0.6235842771485676
 Ratio of  DA Miller is 0.5255179383527034
 Ratio of  T Curran is 0.48
 Ratio of  BA Stokes is 0.4827586206896552
 Ratio of  S Gopal is 0.45925925925925926
 Ratio of  SV Samson is 0.5310734463276836
 Ratio of  M Lomror is 0.42857142857142855
```

Ques3.And and Explanation

From the above calculation we can see that Q De Kock has ratio 0.62 whih is 3rd highest but now he is playing very best.So our ans is Q de Kock

Ans:Quinton De Kock or Sanju Samson

Ques4.Balls difference to score 100 runs*

```
# taking batting team as MI and bowling team as RR
mi_bat = deliveries_df[(deliveries_df['bowling_team']=='Rajasthan Royals') & (deliveries_df['

# taking batting team as RR and bowling team as MI
```

```
rr_bat = deliveries_df[(deliveries_df['batting_team']=='Rajasthan Royals') & (deliveries_df['
```

```
print('No of runs made by MI are',sum(mi_bat['total_runs']), 'in',len(mi_bat),'balls')
print('No of runs made by RR are',sum(rr_bat['total_runs']),'in', len(rr_bat),'balls')
```

```
No of runs made by MI are 3227 in 2414 balls
No of runs made by RR are 3197 in 2389 balls
```

```
# no of balls taken by team to score 100 runs

mi_balls = len(mi_bat)/sum(mi_bat['total_runs'])*100
rr_balls = len(rr_bat)/sum(rr_bat['total_runs'])*100
```

```
print('No of balls MI have taken to score 100 runs are',mi_balls)
print('No of balls RR have taken to score 100 runs are',rr_balls)
```

```
No of balls MI have taken to score 100 runs are 74.80632166098543
No of balls RR have taken to score 100 runs are 74.7263059117923
```

```
# ball difference

print('Balls will RR take to reach a team total of 100 as compared to MI are',mi_balls-rr_bal
```

```
Balls will RR take to reach a team total of 100 as compared to MI are 0.0800157491931372
```

Ques4.Ans and Explanation for ans

From above we can see that possible no of balls difference is 0.08

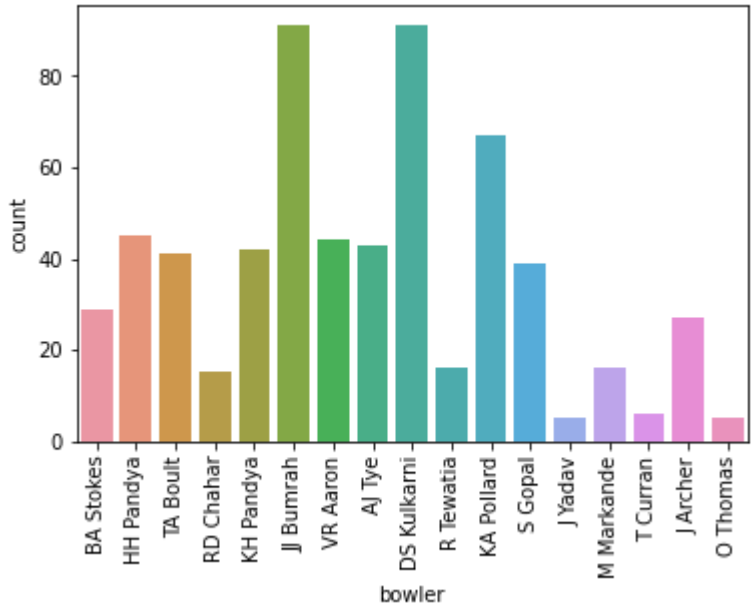
Ans4:0-10

Ques 5.BLS?

```
# taking current bowlers
team_bowlers = deliveries_df[(deliveries_df['bowler']=='JJ Bumrah') | (deliveries_df['bowler'
                             (deliveries_df['bowler']=='KA Pollard') | (deliveries_df['bowler
                             (deliveries_df['bowler']=='KH Pandya') | (deliveries_df['bowler'
                             (deliveries_df['bowler']=='TA Boult') | (deliveries_df['bowler']
                             (deliveries_df['bowler']=='BA Stokes') | (deliveries_df['bowler'
                             (deliveries_df['bowler']=='S Gopal') | (deliveries_df['bowler']=
                             (deliveries_df['bowler']=='O Thomas') | (deliveries_df['bowler']
                             (deliveries_df['bowler']=='R Tewatia') | (deliveries_df['bowler'
                             (deliveries_df['bowler']=='J Archer') | (deliveries_df['bowler']
```

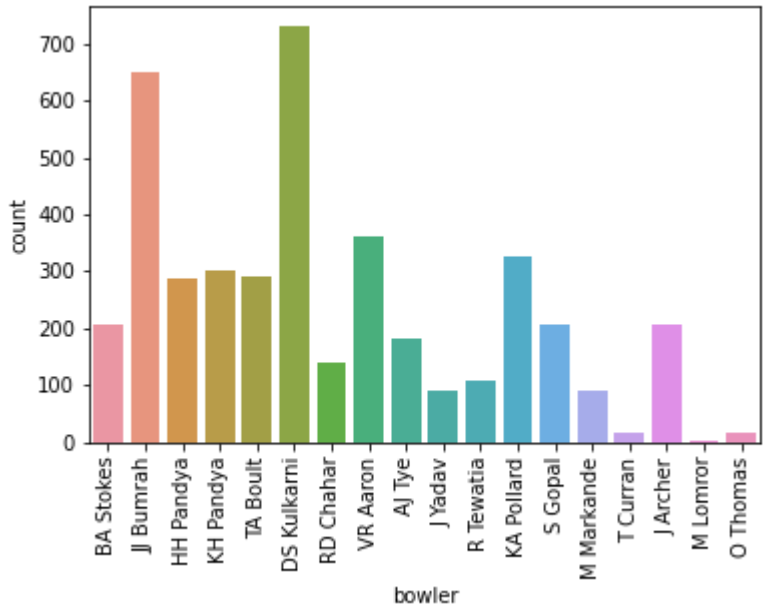
```
# wicket taken
wicket_taken = team_bowlers[team_bowlers['dismissal_kind'].isnull()==False]

# dot balls
dot_balls = team_bowlers[team_bowlers['total_runs']==0]

# boundary runs given
boundaryrun_given = team_bowlers[(team_bowlers['total_runs']==4) | (team_bowlers['total_runs'


z = sns.countplot(x='bowler',data = wicket_taken)
z.set_xticklabels(z.get_xticklabels(),rotation=90)
```

```
[Text(0, 0, 'BA Stokes'),
 Text(0, 0, 'HH Pandya'),
 Text(0, 0, 'TA Boult'),
 Text(0, 0, 'RD Chahar'),
 Text(0, 0, 'KH Pandya'),
 Text(0, 0, 'JJ Bumrah'),
 Text(0, 0, 'VR Aaron'),
 Text(0, 0, 'AJ Tye'),
 Text(0, 0, 'DS Kulkarni'),
 Text(0, 0, 'R Tewatia'),
 Text(0, 0, 'KA Pollard'),
 Text(0, 0, 'S Gopal'),
 Text(0, 0, 'J Yadav'),
 Text(0, 0, 'M Markande'),
 Text(0, 0, 'T Curran'),
 Text(0, 0, 'J Archer'),
 Text(0, 0, 'O Thomas')]
```




```
z3 = sns.countplot(x='bowler',data = dot_balls)
z3.set_xticklabels(z3.get_xticklabels(),rotation=90)
```

```
[Text(0, 0, 'BA Stokes'),
 Text(0, 0, 'JJ Bumrah'),
 Text(0, 0, 'HH Pandya'),
 Text(0, 0, 'KH Pandya'),
 Text(0, 0, 'TA Boult'),
 Text(0, 0, 'DS Kulkarni'),
 Text(0, 0, 'RD Chahar'),
 Text(0, 0, 'VR Aaron'),
 Text(0, 0, 'AJ Tye'),
 Text(0, 0, 'J Yadav'),
 Text(0, 0, 'R Tewatia'),
 Text(0, 0, 'KA Pollard'),
 Text(0, 0, 'S Gopal'),
 Text(0, 0, 'M Markande'),
 Text(0, 0, 'T Curran'),
 Text(0, 0, 'J Archer'),
 Text(0, 0, 'M Lomror'),
 Text(0, 0, 'O Thomas')]
```




```
z5 = sns.countplot(x='bowler',data = boundaryrun_given)
z5.set_xticklabels(z5.get_xticklabels(),rotation=90)
```

```
[Text(0, 0, 'BA Stokes'),
 Text(0, 0, 'HH Pandya'),
 Text(0, 0, 'KH Pandya'),
 Text(0, 0, 'JJ Bumrah'),
 Text(0, 0, 'KA Pollard'),
 Text(0, 0, 'TA Boult'),
 Text(0, 0, 'DS Kulkarni'),
 Text(0, 0, 'RD Chahar'),
 Text(0, 0, 'VR Aaron'),
 Text(0, 0, 'AJ Tye'),
 Text(0, 0, 'J Yadav'),
 Text(0, 0, 'R Tewatia'),
 Text(0, 0, 'S Gopal'),
 Text(0, 0, 'M Markande'),
 Text(0, 0, 'T Curran'),
 Text(0, 0, 'J Archer'),
 Text(0, 0, 'M Lomror'),
 Text(0, 0, 'O Thomas')]
```


```
list_bowl = ['JJ Bumrah','RD Chahar','KA Pollard','HH Pandya','KH Pandya','DS Kulkarni','TA B
             'T Curran','S Gopal','M Lomror','O Thomas','AJ Tye','R Tewatia','M Markande','J

for i in list_bowl:
    z = boundaryrun_given[boundaryrun_given['bowler']==i]
    a = len(wicket_taken[wicket_taken['bowler']==i]) + len(dot_balls[dot_balls['bowler']== i]
    b = sum(z['total_runs'])/len(z)
    c = len(team_bowlers[team_bowlers['bowler']==i])
    bls = (a-b)/c
    print('BLS of' , i , 'is' ,bls)
```

```
BLS of JJ Bumrah is 0.4085010280783362
BLS of RD Chahar is 0.44773344773344775
BLS of KA Pollard is 0.3051537912411931
BLS of HH Pandya is 0.3581285336638798
BLS of KH Pandya is 0.3485701747970453
BLS of DS Kulkarni is 0.44566051325410616
BLS of TA Boult is 0.407156351545379
BLS of J Yadav is 0.3955071477195371
BLS of BA Stokes is 0.37657939882151115
BLS of T Curran is 0.246875
BLS of S Gopal is 0.4029921736498465
BLS of M Lomror is -0.1111111111111111
BLS of O Thomas is 0.2828282828282829
BLS of AJ Tye is 0.3567552548135073
BLS of R Tewatia is 0.39840172398311935
BLS of M Markande is 0.3354040839804429
BLS of J Archer is 0.44622183792093
BLS of VR Aaron is 0.4105504341588239
```

Ques5.Ans and Explanation for ans

From above calculation we can see that J Archer have the highest bls score.So our ans is Jofra Archer or Trent Boult

Ans5:Jofra Archer or Trent Boult

```
1.Made a graph for both team that I have group
average won matches for last 2 years.Also made
matches.Also calulated how many matches the bo
other
2.Made a graph for top 10 batsman.Then made fe
out to calculate BIS and then made a list for
BIS using BIS formula using a for loop iterati
3.Made a graph for batsman.Then made new dataf
then calculted the ratio and also made a count
4.Calculated no of runs made by both team in h
calculated the balls require to make 100 runs
subtract the both result.
5.Made few new dataframe by filtering out to c
list for bowler and calculated the BlS using B
iterating over the list of bowler and made few
visulization.
```

1.Made a graph for both team that I have group by season and calculated average won matches for last 2 years.Also made a graph for total won matches.Also calulated how many matches the both team won against each other 2.Made a graph for top 10 batsman.Then made few new dataframe by filtering out to calculate BIS and then made a list for batsman and calculated the BIS using BIS formula using a for loop iterating over the list of batsman. 3.Made a graph for batsman.Then made new dataframes using some condition then calculted the ratio and also made a countplot for visulization 4.Calculated no of runs made by both team in how many balls.Then calculated the balls require to make 100 runs for both team and then subtract the both result. 5.Made few new dataframe by filtering out to calculate BLS and then made a list for bowler and calculated the BIS using BIS formula using a for loop iterating over the list of bowler and made few countplots for better visulization.