Dataset Info

The Indian Premier league (IPL) is a professional twenty20 cricket

league in india usually contested March and may of every year by

eight teams representing eight different cities or state in india.

The league was founded by board of cricket control in india (BCCI)

in 2007. The IPL is the most- attended cricket league in the world

and the brand value of the IPL in 2019 was rs. 475 billion

importing the essential library

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

Reading the Csv file

```
In [2]: df1 = pd.read_csv('IPL Matches 2008-2020.csv')
        df1.head()
```

Out[2]:

	id	city	date	player_of_match	venue	neutral_venue	team1
0	335982	Bangalore	2008-04-18	BB McCullum	M Chinnaswamy Stadium	0	Royal Challengers Bangalore
1	335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium, Mohali	0	Kings XI Punjab
2	335984	Delhi	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	Delhi Daredevils
3	335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadium	0	Mumbai Indians
4	335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	0	Kolkata Knight Riders

In [3]:
```
df2 = pd.read_csv('IPL Ball-by-Ball 2008-2020.csv')
df2.head()
```

Out[3]:

	id	inning	over	ball	batsman	non_striker	bowler	batsman_runs	extra_runs	total
0	335982	1	6	5	RT Ponting	BB McCullum	AA Noffke	1	0	
1	335982	1	6	6	BB McCullum	RT Ponting	AA Noffke	1	0	
2	335982	1	7	1	BB McCullum	RT Ponting	Z Khan	0	0	
3	335982	1	7	2	BB McCullum	RT Ponting	Z Khan	1	0	
4	335982	1	7	3	RT Ponting	BB McCullum	Z Khan	1	0	

```
In [4]: df2 = df2.groupby('id').sum()
        df = pd.merge(df1,df2,on ='id')
        df
```

```
/tmp/ipykernel_6133/700340395.py:1: FutureWarning: The default valu
e of numeric_only in DataFrameGroupBy.sum is deprecated. In a futur
e version, numeric_only will default to False. Either specify numer
ic_only or select only columns which should be valid for the functi
on.
  df2 = df2.groupby('id').sum()
```

Out[4]:

	id	city	date	player_of_match	venue	neutral_venue	tea
0	335982	Bangalore	2008-04-18	BB McCullum	M Chinnaswamy Stadium	0	Rc Challeng Bangal
1	335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium, Mohali	0	Kings Pun
2	335984	Delhi	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	D Darede
3	335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadium	0	Mum Indi
4	335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	0	Kolk Kni Rid
...	...	...	...	...	...	...	
811	1216547	Dubai	2020-09-28	AB de Villiers	Dubai International Cricket Stadium	0	Rc Challeng Bangal
812	1237177	Dubai	2020-11-05	JJ Bumrah	Dubai International Cricket Stadium	0	Mum Indi
813	1237178	Abu Dhabi	2020-11-06	KS Williamson	Sheikh Zayed Stadium	0	Rc Challeng Bangal
814	1237180	Abu Dhabi	2020-11-08	MP Stoinis	Sheikh Zayed Stadium	0	D Capi
815	1237181	Dubai	2020-11-10	TA Boult	Dubai International Cricket Stadium	0	D Capi

816 rows × 25 columns

```
In [5]: # describing the dataset
```

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

Out[6]:

	id	neutral_venue	result_margin	inning	over	ball	bats
count	8.160000e+02	816.000000	799.000000	816.000000	816.000000	816.000000	
mean	7.563496e+05	0.094363	17.321652	351.403186	2175.810049	857.321078	
std	3.058943e+05	0.292512	22.068427	39.086664	316.109617	83.685974	
min	3.359820e+05	0.000000	1.000000	63.000000	88.000000	178.000000	
25%	5.012278e+05	0.000000	6.000000	348.000000	2131.750000	844.000000	
50%	7.292980e+05	0.000000	8.000000	365.000000	2295.500000	876.000000	
75%	1.082626e+06	0.000000	19.500000	371.000000	2351.000000	899.000000	
max	1.237181e+06	1.000000	146.000000	396.000000	2502.000000	996.000000	

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 816 entries, 0 to 815
Data columns (total 25 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   id               816 non-null    int64  
 1   city             803 non-null    object 
 2   date             816 non-null    object 
 3   player_of_match  812 non-null    object 
 4   venue            816 non-null    object 
 5   neutral_venue    816 non-null    int64  
 6   team1            816 non-null    object 
 7   team2            816 non-null    object 
 8   toss_winner      816 non-null    object 
 9   toss_decision    816 non-null    object 
 10  winner           812 non-null    object 
 11  result           812 non-null    object 
 12  result_margin    799 non-null    float64
 13  eliminator       812 non-null    object 
 14  method           19 non-null     object 
 15  umpire1          816 non-null    object 
 16  umpire2          816 non-null    object 
 17  inning           816 non-null    int64  
 18  over             816 non-null    int64  
 19  ball             816 non-null    int64  
 20  batsman_runs     816 non-null    int64  
 21  extra_runs       816 non-null    int64  
 22  total_runs       816 non-null    int64  
 23  non_boundary     816 non-null    int64  
 24  is_wicket        816 non-null    int64  
dtypes: float64(1), int64(10), object(14)
memory usage: 165.8+ KB
```

In [8]:
```
#checking for null values
```

```
In [9]: df.isnull().sum()
```

```
Out[9]: id                   0
        city                13
        date                 0
        player_of_match      4
        venue                0
        neutral_venue        0
        team1                0
        team2                0
        toss_winner          0
        toss_decision        0
        winner               4
        result               4
        result_margin       17
        eliminator           4
        method             797
        umpire1              0
        umpire2              0
        inning               0
        over                 0
        ball                 0
        batsman_runs         0
        extra_runs           0
        total_runs           0
        non_boundary         0
        is_wicket            0
        dtype: int64
```

```
In [10]: # dealing with the null values

         df['player_of_match']  = df.player_of_match.fillna('No P.O.M')
         df['winner']           = df.winner.fillna('No winner')
         df['result']           = df.result.fillna('No result')
         df['method']           = df['method'].fillna('Match Complete')
         df['eliminator']       = df['eliminator'].fillna('N')
         df['result_margin']    = df['result_margin'].fillna('no result')
         df['city']             = df['city'].fillna('not Available')
```

```
In [11]: # In city the null values are depending on the venue

         city = []
         for i in df.values:
             if i[4] == 'Sharjah Cricket Stadium':
                 city.append(str(i[1]).replace('nan','Sharjah'))
             elif i[4] == 'Dubai International Cricket Stadium':
                 city.append(str(i[1]).replace('nan','Dubai'))
             else:
                 city.append(i[1])

         df['city'] = city
         df.head()
```

Out[11]:

	id	city	date	player_of_match	venue	neutral_venue	team1
0	335982	Bangalore	2008-04-18	BB McCullum	M Chinnaswamy Stadium	0	Royal Challengers Bangalore
1	335983	Chandigarh	2008-04-19	MEK Hussey	Punjab Cricket Association Stadium, Mohali	0	Kings XI Punjab
2	335984	Delhi	2008-04-19	MF Maharoof	Feroz Shah Kotla	0	Delhi Daredevils
3	335985	Mumbai	2008-04-20	MV Boucher	Wankhede Stadium	0	Mumbai Indians
4	335986	Kolkata	2008-04-20	DJ Hussey	Eden Gardens	0	Kolkata Knight Riders

5 rows × 25 columns

```
In [12]: df['date'] = pd.to_datetime(df['date'])
```

```
In [13]: year = []
         for i in df['date'].dt.year:
             year.append(i)

         df['year'] = year
```

In [14]:
```
df[df['ball'] >= df['ball'].quantile(0.99)]
```

Out[14]:

	id	city	date	player_of_match	venue	neutral_venue	tear
66	392190	Cape Town	2009-04-23	YK Pathan	Newlands	1	Kolka Knig Ride
151	419142	Chennai	2010-04-06	SK Raina	MA Chidambaram Stadium, Chepauk	0	Chenn Sup Kir
197	501221	Mumbai	2011-04-22	Harbhajan Singh	Wankhede Stadium	0	Muml India
456	734047	Mumbai	2014-05-30	V Sehwag	Wankhede Stadium	0	Chenn Sup Kir
474	829737	Bangalore	2015-04-19	Harbhajan Singh	M Chinnaswamy Stadium	0	Ro Challenge Bangalo
507	829805	Mumbai	2015-05-14	HH Pandya	Wankhede Stadium	0	Muml India
510	829811	Mumbai	2015-05-16	SR Watson	Brabourne Stadium	0	Rajasth Roy
743	1178423	Hyderabad	2019-04-29	DA Warner	Rajiv Gandhi International Stadium, Uppal	0	Sunrise Hyderab
786	1216522	Dubai	2020-10-17	AB de Villiers	Dubai International Cricket Stadium	0	Rajasth Roy

9 rows × 26 columns

Q1. what was the count of matches played in each season ?

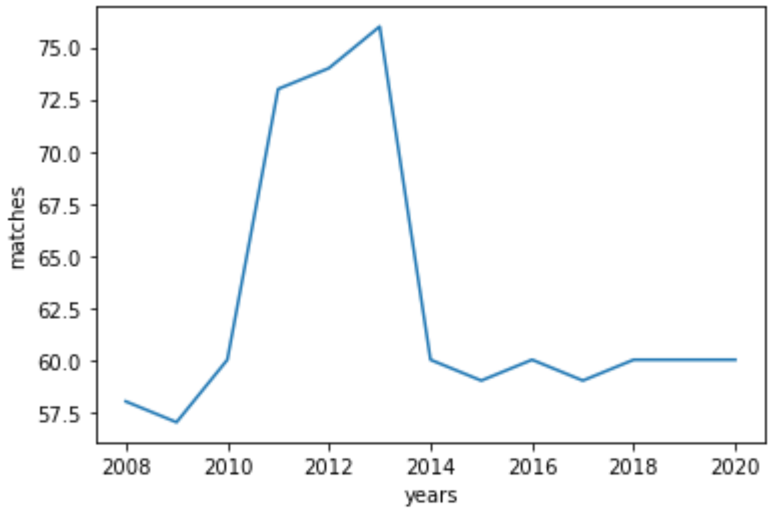
```
In [15]: years = []
         for i in df.values:
             years.append(i[25])

         years = list(set(years))

         matches = []
         for year in years:
             c = 0
             for i in df.values:
                 if (i[25] == year):
                     c+= 1
             matches.append([year,c])

         match_per_season = pd.DataFrame(matches,columns = ['years','matches']
         match_per_season = match_per_season.sort_values(by = 'years',ascendir

         match_per_season
```

Out[15]:

	years	matches
5	2008	58
6	2009	57
7	2010	60
8	2011	73
9	2012	74
10	2013	76
11	2014	60
12	2015	59
0	2016	60
1	2017	59
2	2018	60
3	2019	60
4	2020	60

```
In [16]: # visualization

         sns.lineplot(x = match_per_season['years'],y = match_per_season['matc
```

```
Out[16]: <AxesSubplot:xlabel='years', ylabel='matches'>
```



Q2. How many runs were scored in each season ?

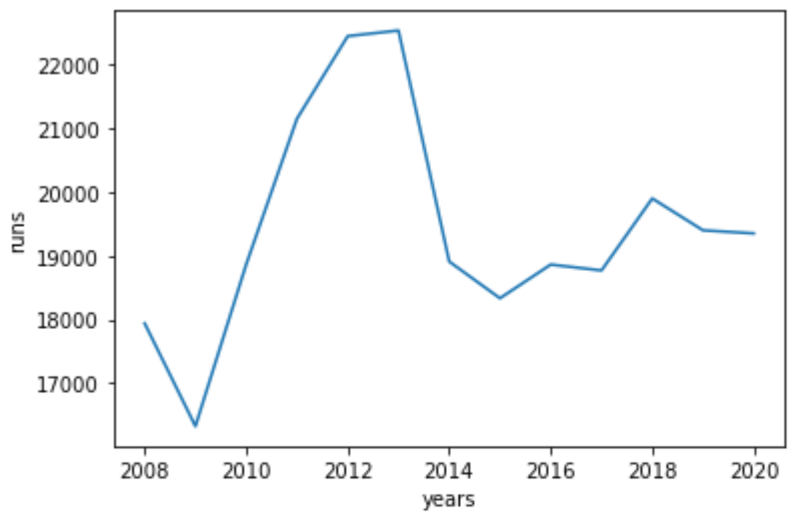
In [17]:
```
years = []
for i in df.values:
    years.append(i[25])

years = list(set(years))

runs = []
for year in years:
    c = 0
    for i in df.values:
        if (i[25] == year):
            c+= i[22]
    runs.append([year,c])

runs_per_season = pd.DataFrame(runs,columns = ['years','runs'])
runs_per_season = runs_per_season.sort_values(by = 'years',ascending
runs_per_season
```

Out[17]:

	years	runs
5	2008	17937
6	2009	16320
7	2010	18864
8	2011	21154
9	2012	22453
10	2013	22541
11	2014	18909
12	2015	18332
0	2016	18862
1	2017	18769
2	2018	19901
3	2019	19400
4	2020	19352

```
In [18]: sns.lineplot(x = runs_per_season['years'],y = runs_per_season['runs']
```

```
Out[18]: <AxesSubplot:xlabel='years', ylabel='runs'>
```



Q3.what were the runs scored per match in different seasons ?

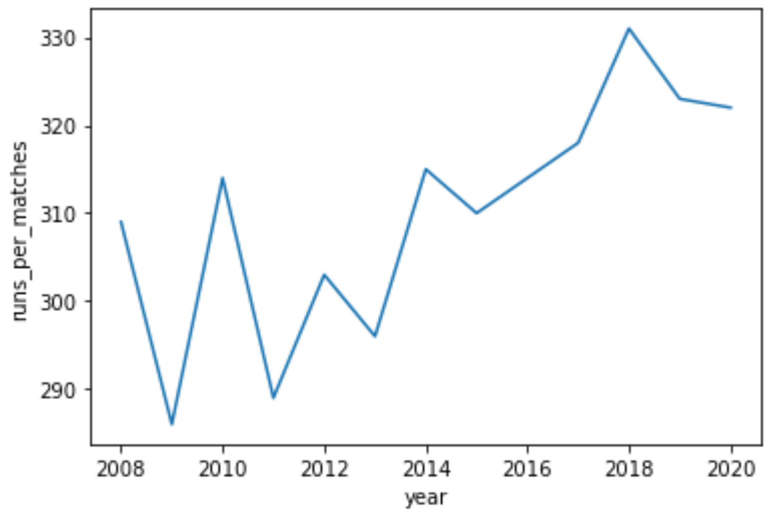
In [19]:
```
years = []
for i in df.values:
    years.append(i[25])

years = list(set(years))

runs_ = []
for year in years:
    run = 0
    c   = 0
    for i in df.values:
        if (i[25] == year):
            run += i[22]
            c   += 1
    runs_.append([year,run, c, run//c])

runs_per_match = pd.DataFrame(runs_, columns = ['year','total_runs',
runs_per_match =  runs_per_match.sort_values(by = 'year',ascending =
runs_per_match
```

Out[19]:

	year	total_runs	total_matches	runs_per_matches
5	2008	17937	58	309
6	2009	16320	57	286
7	2010	18864	60	314
8	2011	21154	73	289
9	2012	22453	74	303
10	2013	22541	76	296
11	2014	18909	60	315
12	2015	18332	59	310
0	2016	18862	60	314
1	2017	18769	59	318
2	2018	19901	60	331
3	2019	19400	60	323
4	2020	19352	60	322

```
In [20]: sns.lineplot(x = runs_per_match['year'], y = runs_per_match['runs_pe
```

```
Out[20]: <AxesSubplot:xlabel='year', ylabel='runs_per_matches'>
```



Q4. who has umpired the most ?

```
In [21]: ump = pd.concat([df['umpire1'],df['umpire2']]).value_counts()
         ump.head(10)
```

```
Out[21]: S Ravi               121
         HDPK Dharmasena       94
         AK Chaudhary          87
         C Shamshuddin         82
         M Erasmus             65
         CK Nandan             57
         Nitin Menon           57
         SJA Taufel            55
         Asad Rauf             51
         VA Kulkarni           50
         dtype: int64
```

Q5.which team has won the most tosses ?

```
In [22]: most_tosses = {}
         for i in df.values:
             if i[8] in most_tosses:
                 most_tosses[i[8]] += 1
             else:
                 most_tosses[i[8]]  = 1

         print('Most tosses Won By :',list(most_tosses.keys())[list(most_tosse
         print('Tosses won         :', max(most_tosses.values()))

         # thus mumbai indians have won 106 toss in total
```

```
Most tosses Won By : Mumbai Indians
Tosses won         : 106
```

Q6.what does the team decide after winning the toss ?

```
In [23]: bat   = 0
         field = 0
         for i in df.values:
             if (i[9] == 'bat'):
                 bat += 1
             else:
                 field += 1

         print('Number of times team decides to bat first        :',bat)
         print('Number of times team decides to bowl/field first :',field)

         # we can conclude that the toss winning captain decides to field firs
```

```
Number of times team decides to bat first        : 320
Number of times team decides to bowl/field first : 496
```

Q7.how does the toss decision vary across seasons ?

```
In [24]: years = []
         for i in df.values:
             years.append(i[25])

         years = list(set(years))

         toss_per_seasons = []
         for year in years:
             bat  = 0
             bowl = 0
             for i in df.values:
                 if (i[25] == year):
                     if (i[9] == 'bat'):
                         bat += 1
                     else:
                         bowl += 1
             toss_per_seasons.append([year,bat,bowl])
         toss_per_season = pd.DataFrame(toss_per_seasons,columns = ['year','ba
         toss_per_season = toss_per_season.sort_values(by='year',ascending = T
         toss_per_season
```

Out[24]:

	year	bat_toss	bowl_toss
5	2008	26	32
6	2009	35	22
7	2010	39	21
8	2011	25	48
9	2012	37	37
10	2013	45	31
11	2014	19	41
12	2015	25	34
0	2016	11	49
1	2017	11	48
2	2018	10	50
3	2019	10	50
4	2020	27	33

Q8.does winning the toss imply winning the game ?

In [25]:
```
c = 0
for i in df.values:
    if (i[8] != i[10]):
        c += 1
print('probablity of toss winning team won matches:',100-(c/len(df))*

# we can conclude that toss can imply that there is 50 percentage cer
# the game
# that there is 51:49 of same team winning the game : opposite team w
```

```
probablity of toss winning team won matches: 51.22549019607843
```

Q9.how many times has the chasing team won the matches ?

In [26]:
```
c = 0
for i in df.values:
    if i[11] == 'wickets':
        c += 1

print('Number of times chasing team won the matches :',c)

# 435 times chasing team has won the matches
```

```
Number of times chasing team won the matches : 435
```

Q10.which all teams had won this tournament ?

```
In [27]: tour = []
         for year in df['year'].unique():
             team = ''
             for i in df.values:
                 if i[25] == year:
                     team += i[10] + ','


             tour.append([year,team.split(',')[-2]])

         tour_winner = pd.DataFrame(tour,columns = ['year','Team'])
         tour_winner
```

Out[27]:

	year	Team
0	2008	Rajasthan Royals
1	2009	Deccan Chargers
2	2010	Chennai Super Kings
3	2011	Chennai Super Kings
4	2012	Kolkata Knight Riders
5	2013	Mumbai Indians
6	2014	Kolkata Knight Riders
7	2015	Mumbai Indians
8	2016	Sunrisers Hyderabad
9	2017	Mumbai Indians
10	2018	Chennai Super Kings
11	2019	Mumbai Indians
12	2020	Mumbai Indians

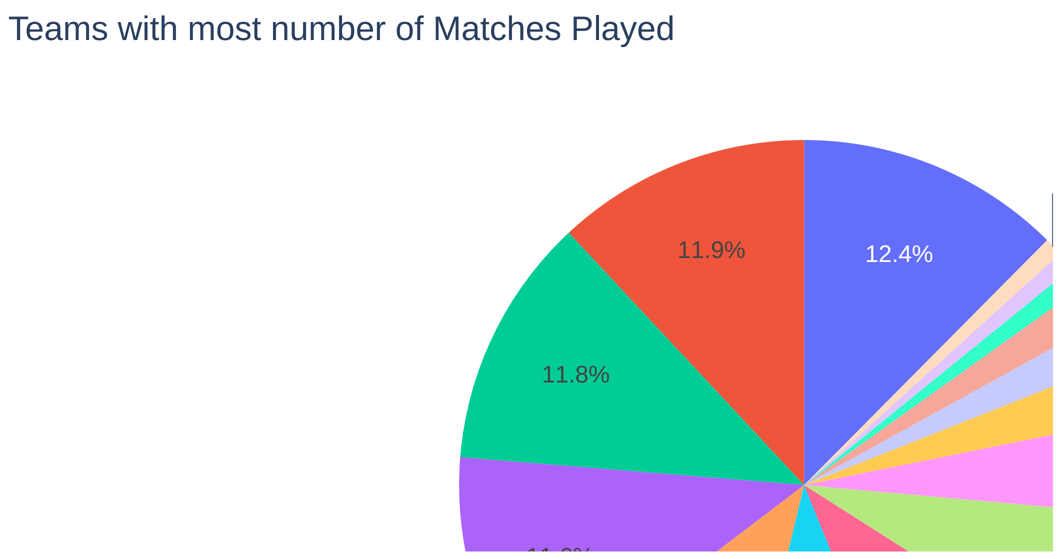
Q11.which team has played the most number of matches ?

```
In [28]: match_played=pd.concat([df['team1'],df['team2']],axis=1)
         teams=(match_played['team1'].value_counts()+match_played['team2'].val
         teams.columns=['Team Name','Total Matches played']
         teams.sort_values(by=['Total Matches played'],ascending=False)
```

```
Out[28]:
```

	Team Name	Total Matches played
8	Mumbai Indians	203
13	Royal Challengers Bangalore	195
7	Kolkata Knight Riders	192
5	Kings XI Punjab	190
0	Chennai Super Kings	178
3	Delhi Daredevils	161
10	Rajasthan Royals	161
14	Sunrisers Hyderabad	124
1	Deccan Chargers	75
9	Pune Warriors	46
2	Delhi Capitals	33
4	Gujarat Lions	30
11	Rising Pune Supergiant	16
6	Kochi Tuskers Kerala	14
12	Rising Pune Supergiants	14

```
In [29]: px.pie(teams,names = 'Team Name',values = 'Total Matches played',titl
```



Q12.which team has won the most number of times ?

```
In [30]: def most_winner(data):
             winner = {}
             for i in df['winner']:
                 if i not in winner:
                     winner[i] = 1
                 else:
                     winner[i] += 1
             return (list(winner.keys())[list(winner.values()).index(max(winne
         team,matches = most_winner(df)
         print('Most Number of times team Won :',team)
         print('Number of times team won      :',matches)

         # print('Name of the team         :',list(winner.keys())[list(winner.
         # print('Number of times team won :',max(winner.values()))

         # Thus we can conclude that mumbai indians have won the most number c
```

```
Most Number of times team Won : Mumbai Indians
Number of times team won      : 120
```

Q13.which team has the highest winning percentage ?

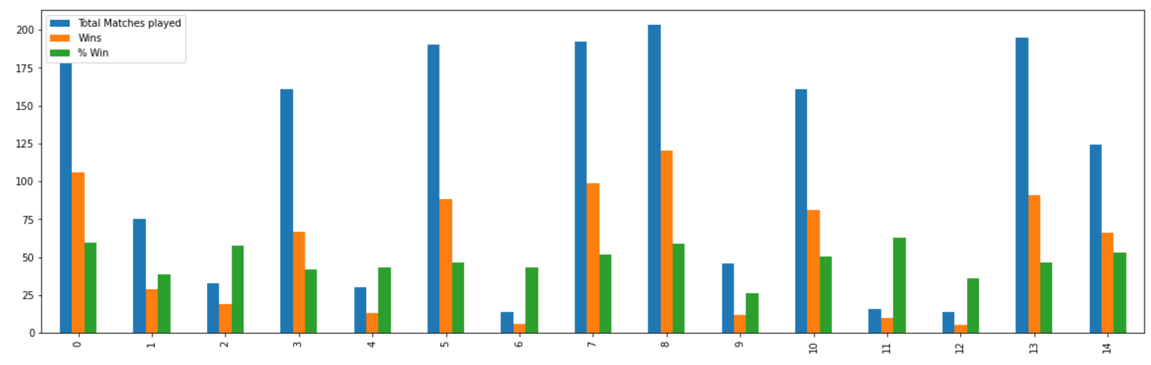
```
In [31]: wins=pd.DataFrame(df['winner'].value_counts()).reset_index()
         wins.columns=['Team Name','Wins']
         played=teams.merge(wins,left_on='Team Name',right_on='Team Name',how=
         played['% Win']=(played['Wins']/played['Total Matches played'])*100
         played.sort_values(by = '% Win',ascending = False)
```

Out[31]:

	Team Name	Total Matches played	Wins	% Win
11	Rising Pune Supergiant	16	10	62.500000
0	Chennai Super Kings	178	106	59.550562
8	Mumbai Indians	203	120	59.113300
2	Delhi Capitals	33	19	57.575758
14	Sunrisers Hyderabad	124	66	53.225806
7	Kolkata Knight Riders	192	99	51.562500
10	Rajasthan Royals	161	81	50.310559
13	Royal Challengers Bangalore	195	91	46.666667
5	Kings XI Punjab	190	88	46.315789
4	Gujarat Lions	30	13	43.333333
6	Kochi Tuskers Kerala	14	6	42.857143
3	Delhi Daredevils	161	67	41.614907
1	Deccan Chargers	75	29	38.666667
12	Rising Pune Supergiants	14	5	35.714286
9	Pune Warriors	46	12	26.086957

```
In [32]: played.plot.bar(figsize = (20,6))
```

```
Out[32]: <AxesSubplot:>
```



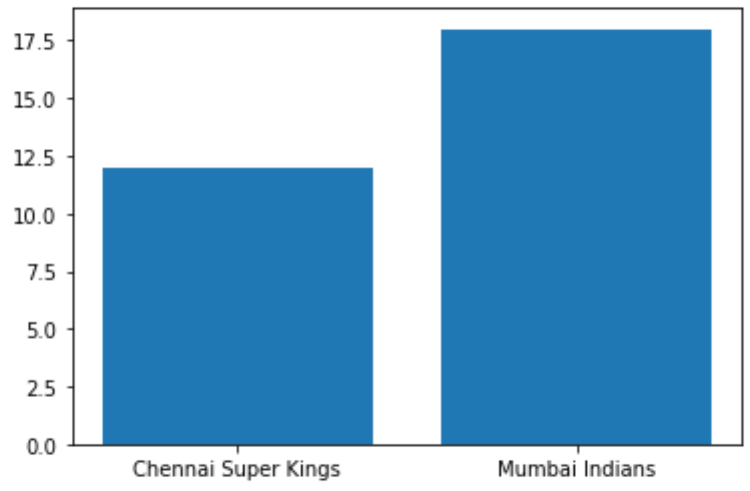
Q14. is there any lucky venue for a particular team ?

```
In [33]: new_df = pd.read_csv('IPL_data.csv')
         def luck_venue(dataframe,team_name):
             return dataframe[dataframe['winner'] == team_name]['venue'].value
```

```
In [34]: luck_venue(new_df,'Chennai Super Kings')

Out[34]: MA Chidambaram Stadium, Chepauk             40
         Wankhede Stadium                             7
         Dubai International Cricket Stadium          6
         Feroz Shah Kotla                             6
         Maharashtra Cricket Association Stadium      5
         Name: venue, dtype: int64
```

Q15.innings wise comparison between teams ?

```
In [35]: def compare(data,team1,team2):
             win_team = data[((data['team1'] == team1) | (data['team2'] == tea
             dct = {}
             for i in win_team:
                 if (i) not in dct:
                     dct[i] = 1
                 else:
                     dct[i] += 1

             plt.bar(dct.keys(),dct.values())
             plt.show()

         compare(new_df,'Mumbai Indians','Chennai Super Kings')
```



Q16.which team has scored the most number of 200+ scores ?

In [36]:
```
df2 = pd.read_csv('IPL Ball-by-Ball 2008-2020.csv')
high_scores = df2.groupby(['id', 'inning','batting_team','bowling_tea
team_200 = high_scores[high_scores['total_runs']>=200]
team_200.head(5)
```

Out[36]:

	id	inning	batting_team	bowling_team	total_runs
0	335982	1	Kolkata Knight Riders	Royal Challengers Bangalore	222
2	335983	1	Chennai Super Kings	Kings XI Punjab	240
3	335983	2	Kings XI Punjab	Chennai Super Kings	207
14	335989	1	Chennai Super Kings	Mumbai Indians	208
15	335989	2	Mumbai Indians	Chennai Super Kings	202

In [37]:
```
team_200['batting_team'].value_counts()
```

Out[37]:
```
Royal Challengers Bangalore    18
Chennai Super Kings            17
Kings XI Punjab                14
Mumbai Indians                 14
Kolkata Knight Riders          12
Sunrisers Hyderabad            12
Rajasthan Royals                9
Delhi Daredevils                5
Delhi Capitals                  2
Deccan Chargers                 1
Gujarat Lions                   1
Name: batting_team, dtype: int64
```

Q17.which team has conceded 200+ scores the most ?

In [38]:
```
team_200['bowling_team'].value_counts()
```

Out[38]:
```
Kings XI Punjab                20
Royal Challengers Bangalore    17
Chennai Super Kings            12
Delhi Daredevils               11
Rajasthan Royals               10
Kolkata Knight Riders          10
Mumbai Indians                  8
Sunrisers Hyderabad             7
Gujarat Lions                   3
Delhi Capitals                  3
Deccan Chargers                 2
Pune Warriors                   1
Rising Pune Supergiant          1
Name: bowling_team, dtype: int64
```

Q18.what was the highest run scored by a team in a single match ?

```
In [39]: data = pd.read_csv('IPL_data.csv')
         high_score = 0
         for i in data.values:
             if (i[17] > high_score):
                 high_score = i[17]
             elif i[18] > high_score:
                 high_score = i[17]

         print('Highest runs scored by a team :',high_score)
```

```
Highest runs scored by a team : 263
```

Q19.which is the biggest win in terms of run margin ?

```
In [40]: run_margin = []
         for i in df.values:
             if i[11] == 'runs':
                 run_margin.append(i[12])

         print('Biggest win in terms of run margin :',max(run_margin))
```

```
Biggest win in terms of run margin : 146.0
```

Q20.which batsmen have played the most number of balls ?

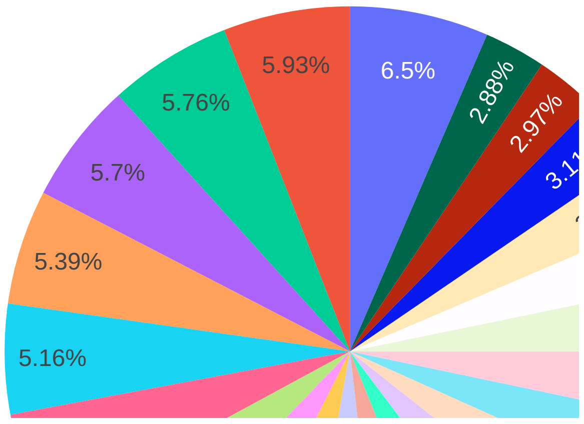
In [41]:
```
df2 = pd.read_csv('IPL Ball-by-Ball 2008-2020.csv')

most_ball = {}
for i in df2.values:
    if (i[4]) not in most_ball:
        most_ball[i[4]] = 1
    else:
        most_ball[i[4]] += 1

most_balls = pd.DataFrame()
most_balls['batsman'] = most_ball.keys()
most_balls['balls played'] = most_ball.values()
most_balls_ = most_balls[most_balls['balls played']>2000].sort_values
most_balls_
```

Out[41]:

	batsman	balls played
15	V Kohli	4609
31	S Dhawan	4208
61	RG Sharma	4088
19	SK Raina	4041
186	DA Warner	3819
45	RV Uthappa	3658
32	G Gambhir	3524
24	MS Dhoni	3493
163	CH Gayle	3342
85	AM Rahane	3325
107	AB de Villiers	3264
89	KD Karthik	3023
208	AT Rayudu	2970
37	SR Watson	2888
96	MK Pandey	2772
22	PA Patel	2442
30	YK Pathan	2330
13	JH Kallis	2291
1	BB McCullum	2267
185	M Vijay	2208
26	Yuvraj Singh	2205
223	KA Pollard	2107
133	SR Tendulkar	2044

```
In [44]: # batsman who have played most number of ball where balls played is g
         fig = px.pie(most_balls_, names = 'batsman', values = 'balls played',
         fig.show()
```

Most number of balls played by any batsman



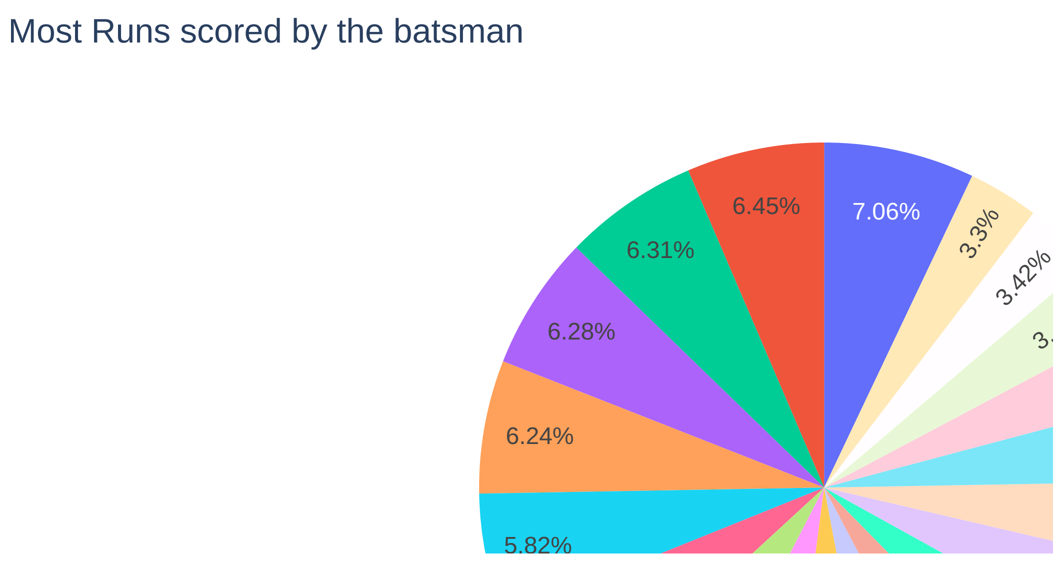
Q21.who are the leading run-scorers of all time ?

```
In [45]: lead_scorer = {}
         for i in df2.values:
             if i[4] not in lead_scorer:
                 lead_scorer[i[4]] = i[7]
             elif i[4] in lead_scorer:
                 lead_scorer[i[4]] += i[7]

         lead_scorers = pd.DataFrame()
         lead_scorers['Batsmen'] = lead_scorer.keys()
         lead_scorers['Runs']    = lead_scorer.values()
         lead_scorers_ = lead_scorers.sort_values(by = 'Runs',ascending = Fals
         # top 20 batsman
         lead_scorers_
```

Out[45]:

	Batsmen	Runs
15	V Kohli	5878
19	SK Raina	5368
186	DA Warner	5254
61	RG Sharma	5230
31	S Dhawan	5197
107	AB de Villiers	4849
163	CH Gayle	4772
24	MS Dhoni	4632
45	RV Uthappa	4607
32	G Gambhir	4217
85	AM Rahane	3933
37	SR Watson	3874
89	KD Karthik	3823
208	AT Rayudu	3659
96	MK Pandey	3268
30	YK Pathan	3204
223	KA Pollard	3023
1	BB McCullum	2880
22	PA Patel	2848
26	Yuvraj Singh	2750

```
In [46]: fig = px.pie(lead_scorers_,names = 'Batsmen', values = 'Runs', title
         fig.show()
```



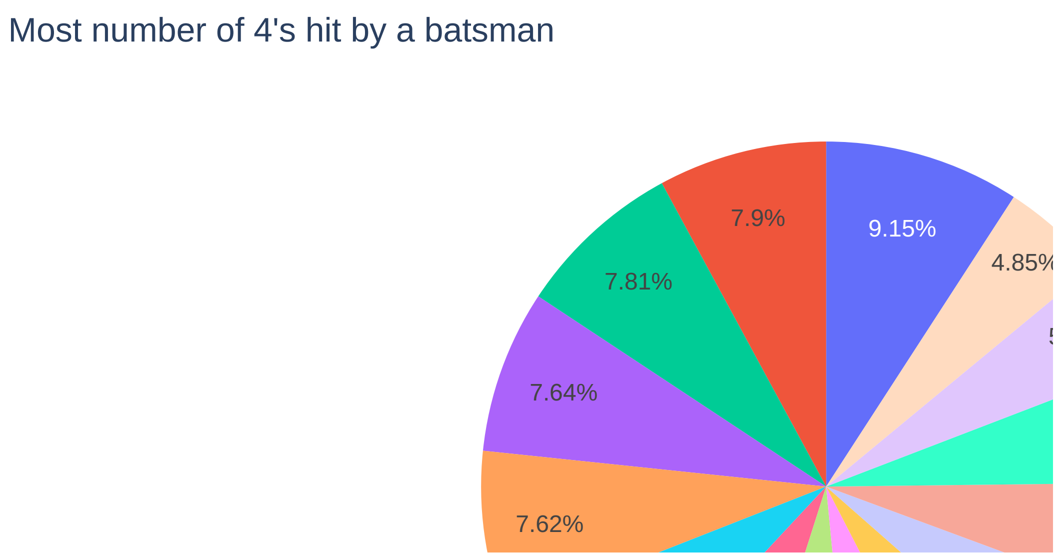
Q22.who has hit the most number of 4's ?

```
In [47]: four_bat = {}
         for i in df2.values:
             if i[4] not in four_bat:
                 four_bat[i[4]] = 0
             elif i[7] == 4 and i[4] in four_bat:
                 four_bat[i[4]] += 1

         batsman_4 = pd.DataFrame()
         batsman_4['name'] = four_bat.keys()
         batsman_4['number of fours'] = four_bat.values()
         batsman_4 = batsman_4[batsman_4['number of fours'] >= batsman_4['numb
         batsman_4.head(1)
```

Out[47]:

	name	number of fours
31	S Dhawan	591

In [48]:
```
# Top 15 four hitting batsman
batsman_4 = batsman_4.head(15)

# plotting the graph
fig = px.pie(batsman_4,values = 'number of fours',names = 'name' ,tit
fig.show()
```



Q23.who has hit the most number of 6's ?

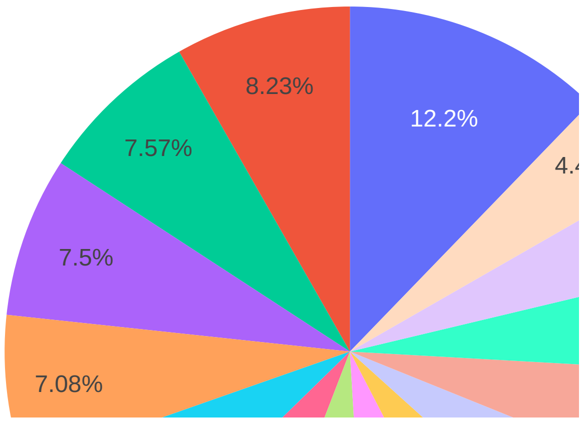
In [49]:
```
six_bat = {}
for i in df2.values:
    if i[4] not in six_bat:
        six_bat[i[4]] = 0
    elif i[7] == 6 and i[4] in six_bat:
        six_bat[i[4]] += 1

batsman_6 = pd.DataFrame()
batsman_6['name'] = six_bat.keys()
batsman_6['number of sixes'] = six_bat.values()
batsman_6 = batsman_6[batsman_6['number of sixes'] >= batsman_6['numb
batsman_6.head(1)
```

Out[49]:

	name	number of sixes
163	CH Gayle	349

```
In [50]: #top 15 batsman
         batsman_6 = batsman_6.head(15)

         # plotting the graph
         fig = px.pie(batsman_6,values = 'number of sixes',names = 'name' ,tit
         fig.show()
```

Most number of 6's hit by a batsman



Q24.who has the highest strike rate ?

In [51]:
```
player = pd.concat([lead_scorers,most_balls.iloc[:,1]],axis=1)
player['highest_strike_rate'] = player['Runs']/player['balls played']
player.sort_values(by ='highest_strike_rate',ascending = False)

# top 10 batsman who has played greater than 100 balls of all times
player[player['balls played'] >= 100].sort_values(by = 'highest_stri
```

Out[51]:

	Batsmen	Runs	balls played	highest_strike_rate
334	AD Russell	1517	882	171.995465
465	K Gowtham	186	113	164.601770
386	BCJ Cutting	238	146	163.013699
495	N Pooran	521	323	161.300310
315	SP Narine	892	573	155.671902
483	MM Ali	309	199	155.276382
350	CH Morris	551	360	153.055556
476	JC Archer	195	128	152.343750
408	CR Brathwaite	181	120	150.833333
227	Bipul Sharma	187	124	150.806452

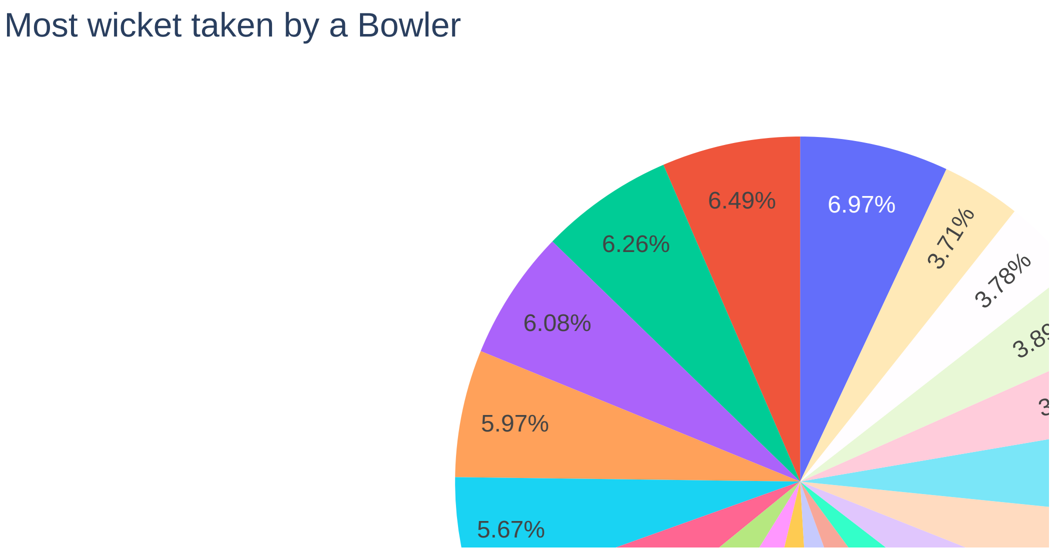
Q25.who is the leading wicket-taker ?

In [52]:
```
wick = {}
for i in df2.values:
    if i[6] not in wick:
        wick[i[6]] = 0
    elif i[11] == 1 and i[6] in wick:
        wick[i[6]] += 1

bowler = pd.DataFrame()
bowler['Names'] = wick.keys()
bowler['Wickets_Taken'] = wick.values()
bowler = bowler[bowler['Wickets_Taken'] >= bowler['Wickets_Taken'].me
bowler.head(1)
```

Out[52]:

	Names	Wickets_Taken
100	SL Malinga	188

```
In [53]: # top 20 wicket taking bowler
         bowler = bowler.head(20)

         # ploting using plotly
         fig = px.pie(bowler, values = 'Wickets_Taken', names = 'Names',title
         fig.show()
```



Q26.Which stadium haas hosted the most number of matches ?

```
In [54]: venue = {}
         for i in df.values:
             if i[4] not in venue:
                 venue[i[4]] = 1
             else:
                 venue[i[4]] += 1

         print('Venue Name               :',list(venue.keys())[list(venue.value
         print('Number of times hosted   :', max(venue.values()))
```

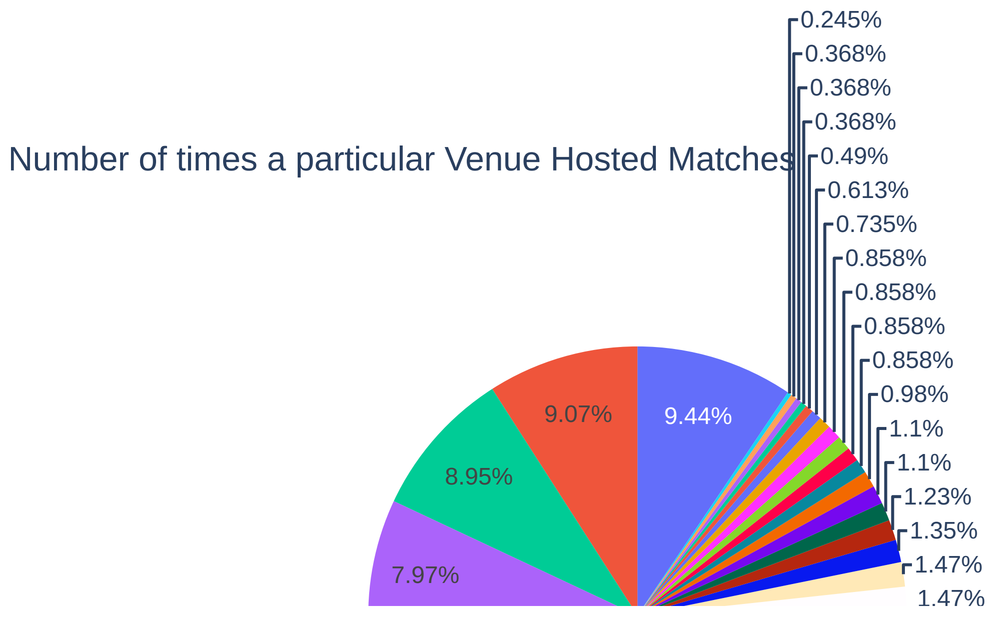
```
Venue Name               : Eden Gardens
Number of times hosted   : 77
```

```
In [55]: venues = pd.DataFrame()
         venues['V_name'] = venue.keys()
         venues['Times_hosted'] = venue.values()
         venues.sort_values(by='Times_hosted',ascending = False).head(10)
```

```
Out[55]:
```

	V_name	Times_hosted
4	Eden Gardens	77
2	Feroz Shah Kotla	74
3	Wankhede Stadium	73
0	M Chinnaswamy Stadium	65
6	Rajiv Gandhi International Stadium, Uppal	64
7	MA Chidambaram Stadium, Chepauk	57
5	Sawai Mansingh Stadium	47
1	Punjab Cricket Association Stadium, Mohali	35
30	Dubai International Cricket Stadium	33
28	Sheikh Zayed Stadium	29

```
In [56]: fig = px.pie(venues, values = 'Times_hosted', names = 'V_name',title
         fig.show()
```


Number of times a particular Venue Hosted Matches