Task 5- Exploratory Data Analysis - Sports

Exploratory Data Analysis - Sports (Level - Advanced) Assignment - 5

Author:Aniket Suresh Hendre

1. Importing the required Libraries

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
%matplotlib inline
from sklearn.cluster import KMeans
from sklearn import datasets
import warnings
warnings.filterwarnings("ignore")
import os
import mpl_toolkits
import json
print('Libraries are imported Successfully')
```

```
Libraries are imported Successfully
```

In [2]:
```
#Importing the dataset
df_deliveries=pd.read_csv('deliveries.csv',low_memory=False)
print('Data Read Successfully')
```

```
Data Read Successfully
```

In [3]:
```
df_deliveries
```

Out[3]:

	match_id	inning	batting_team	bowling_team	over	ball	batsman	non_striker	bowler	is_super_over	...	bye_runs	legbye_runs
0	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	1	DA Warner	S Dhawan	TS Mills	0	...	0	0
1	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	2	DA Warner	S Dhawan	TS Mills	0	...	0	0
2	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	3	DA Warner	S Dhawan	TS Mills	0	...	0	0
3	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	4	DA Warner	S Dhawan	TS Mills	0	...	0	0
4	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	5	DA Warner	S Dhawan	TS Mills	0	...	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
179073	11415	2	Chennai Super Kings	Mumbai Indians	20	2	RA Jadeja	SR Watson	SL Malinga	0	...	0	0
179074	11415	2	Chennai Super Kings	Mumbai Indians	20	3	SR Watson	RA Jadeja	SL Malinga	0	...	0	0
179075	11415	2	Chennai Super Kings	Mumbai Indians	20	4	SR Watson	RA Jadeja	SL Malinga	0	...	0	0
179076	11415	2	Chennai Super Kings	Mumbai Indians	20	5	SN Thakur	RA Jadeja	SL Malinga	0	...	0	0
179077	11415	2	Chennai Super Kings	Mumbai Indians	20	6	SN Thakur	RA Jadeja	SL Malinga	0	...	0	0

179078 rows × 21 columns

In [4]:
```
df_matches=pd.read_csv('matches.csv',low_memory=False)
print('Data Read Successfully')
```

```
Data Read Successfully
```

In [5]:
```
df_matches
```

Out[5]:

	id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_rur
0	1	2017	Hyderabad	2017-04-05	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	3
1	2	2017	Pune	2017-04-06	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	0	Rising Pune Supergiant	
2	3	2017	Rajkot	2017-04-07	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	0	Kolkata Knight Riders	
3	4	2017	Indore	2017-04-08	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	0	Kings XI Punjab	
4	5	2017	Bangalore	2017-04-08	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	0	Royal Challengers Bangalore	1
...	...	...	...	...	...	...	...	...	...	...	...	
751	11347	2019	Mumbai	05/05/19	Kolkata Knight Riders	Mumbai Indians	Mumbai Indians	field	normal	0	Mumbai Indians	
752	11412	2019	Chennai	07/05/19	Chennai Super Kings	Mumbai Indians	Chennai Super Kings	bat	normal	0	Mumbai Indians	
753	11413	2019	Visakhapatnam	08/05/19	Sunrisers Hyderabad	Delhi Capitals	Delhi Capitals	field	normal	0	Delhi Capitals	
754	11414	2019	Visakhapatnam	10/05/19	Delhi Capitals	Chennai Super Kings	Chennai Super Kings	field	normal	0	Chennai Super Kings	
755	11415	2019	Hyderabad	12/05/19	Mumbai Indians	Chennai Super Kings	Mumbai Indians	bat	normal	0	Mumbai Indians	

756 rows × 18 columns

In [6]:
```
#Pre processing of Data
df_matches.head()
```

Out[6]:

	id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_w
0	1	2017	Hyderabad	2017-04-05	Sunrisers Hyderabad	Royal Challengers Bangalore	Royal Challengers Bangalore	field	normal	0	Sunrisers Hyderabad	35	
1	2	2017	Pune	2017-04-06	Mumbai Indians	Rising Pune Supergiant	Rising Pune Supergiant	field	normal	0	Rising Pune Supergiant	0	
2	3	2017	Rajkot	2017-04-07	Gujarat Lions	Kolkata Knight Riders	Kolkata Knight Riders	field	normal	0	Kolkata Knight Riders	0	
3	4	2017	Indore	2017-04-08	Rising Pune Supergiant	Kings XI Punjab	Kings XI Punjab	field	normal	0	Kings XI Punjab	0	
4	5	2017	Bangalore	2017-04-08	Royal Challengers Bangalore	Delhi Daredevils	Royal Challengers Bangalore	bat	normal	0	Royal Challengers Bangalore	15	

In [7]:
```
df_matches.tail()
```

Out[7]:

	id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_
751	11347	2019	Mumbai	05/05/19	Kolkata Knight Riders	Mumbai Indians	Mumbai Indians	field	normal	0	Mumbai Indians	0	
752	11412	2019	Chennai	07/05/19	Chennai Super Kings	Mumbai Indians	Chennai Super Kings	bat	normal	0	Mumbai Indians	0	
753	11413	2019	Visakhapatnam	08/05/19	Sunrisers Hyderabad	Delhi Capitals	Delhi Capitals	field	normal	0	Delhi Capitals	0	
754	11414	2019	Visakhapatnam	10/05/19	Delhi Capitals	Chennai Super Kings	Chennai Super Kings	field	normal	0	Chennai Super Kings	0	
755	11415	2019	Hyderabad	12/05/19	Mumbai Indians	Chennai Super Kings	Mumbai Indians	bat	normal	0	Mumbai Indians	1	

In [8]:
```
df_matches['team1'].unique()
```

Out[8]:
```
array(['Sunrisers Hyderabad', 'Mumbai Indians', 'Gujarat Lions',
       'Rising Pune Supergiant', 'Royal Challengers Bangalore',
       'Kolkata Knight Riders', 'Delhi Daredevils', 'Kings XI Punjab',
       'Chennai Super Kings', 'Rajasthan Royals', 'Deccan Chargers',
       'Kochi Tuskers Kerala', 'Pune Warriors', 'Rising Pune Supergiants',
       'Delhi Capitals'], dtype=object)
```

In [9]:
```
df_matches['team2'].unique()
```

Out[9]:
```
array(['Royal Challengers Bangalore', 'Rising Pune Supergiant',
       'Kolkata Knight Riders', 'Kings XI Punjab', 'Delhi Daredevils',
       'Sunrisers Hyderabad', 'Mumbai Indians', 'Gujarat Lions',
       'Rajasthan Royals', 'Chennai Super Kings', 'Deccan Chargers',
       'Pune Warriors', 'Kochi Tuskers Kerala', 'Rising Pune Supergiants',
       'Delhi Capitals'], dtype=object)
```

In [10]:
```
df_deliveries['batting_team'].unique()
```

Out[10]:
```
array(['Sunrisers Hyderabad', 'Royal Challengers Bangalore',
       'Mumbai Indians', 'Rising Pune Supergiant', 'Gujarat Lions',
       'Kolkata Knight Riders', 'Kings XI Punjab', 'Delhi Daredevils',
       'Chennai Super Kings', 'Rajasthan Royals', 'Deccan Chargers',
       'Kochi Tuskers Kerala', 'Pune Warriors', 'Rising Pune Supergiants',
       'Delhi Capitals'], dtype=object)
```

In [11]:
```
df_deliveries['bowling_team'].unique()
```

Out[11]:
```
array(['Royal Challengers Bangalore', 'Sunrisers Hyderabad',
       'Rising Pune Supergiant', 'Mumbai Indians',
       'Kolkata Knight Riders', 'Gujarat Lions', 'Kings XI Punjab',
       'Delhi Daredevils', 'Chennai Super Kings', 'Rajasthan Royals',
       'Deccan Chargers', 'Kochi Tuskers Kerala', 'Pune Warriors',
       'Rising Pune Supergiants', 'Delhi Capitals'], dtype=object)
```

In [12]:
```
df_matches.replace(['Royal Challengers Bangalore', 'Sunrisers Hyderabad',
       'Rising Pune Supergiant', 'Mumbai Indians',
       'Kolkata Knight Riders', 'Gujarat Lions', 'Kings XI Punjab',
       'Delhi Daredevils', 'Chennai Super Kings', 'Rajasthan Royals',
       'Deccan Chargers', 'Kochi Tuskers Kerala', 'Pune Warriors',
       'Rising Pune Supergiants', 'Delhi Capitals'], ['RCB', 'SRH', 'PW', 'MI', 'KKR', 'GL', 'KXIP', 'DD','CSK'
```

In [14]:
```
df_deliveries.replace(['Royal Challengers Bangalore', 'Sunrisers Hyderabad',
       'Rising Pune Supergiant', 'Mumbai Indians',
       'Kolkata Knight Riders', 'Gujarat Lions', 'Kings XI Punjab',
       'Delhi Daredevils', 'Chennai Super Kings', 'Rajasthan Royals',
       'Deccan Chargers', 'Kochi Tuskers Kerala', 'Pune Warriors',
       'Rising Pune Supergiants', 'Delhi Capitals'], ['RCB', 'SRH', 'PW', 'MI', 'KKR', 'GL', 'KXIP', 'DD','CSK'
```

In [15]:
```
print('Total Matches played:',df_matches.shape[0])
print('\n Venues played at:', df_matches['city'].unique())
print('\n Teams:', df_matches['team1'].unique)
```

```
Total Matches played: 756

 Venues played at: ['Hyderabad' 'Pune' 'Rajkot' 'Indore' 'Bangalore' 'Mumbai' 'Kolkata'
 'Delhi' 'Chandigarh' 'Kanpur' 'Jaipur' 'Chennai' 'Cape Town'
 'Port Elizabeth' 'Durban' 'Centurion' 'East London' 'Johannesburg'
 'Kimberley' 'Bloemfontein' 'Ahmedabad' 'Cuttack' 'Nagpur' 'Dharamsala'
 'Kochi' 'Visakhapatnam' 'Raipur' 'Ranchi' 'Abu Dhabi' 'Sharjah' nan
 'Mohali' 'Bengaluru']

 Teams: <bound method Series.unique of 0      SRH
1       MI
2       GL
3       PW
4      RCB
      ...
751    KKR
752    CSK
753    SRH
754     DC
755     MI
Name: team1, Length: 756, dtype: object>
```

In [16]:
```
print('Total venues playes at:', df_matches['city'].nunique())
print('\n Total umpires:', df_matches['umpire1'].nunique())
```

```
Total venues playes at: 32

 Total umpires: 61
```

In [17]:
```
print((df_matches['player_of_match'].value_counts()).idxmax(), ':has most man of the match awards')
print((df_matches['winner'].value_counts()).idxmax(), ':has the highest number of match wins')
```

```
CH Gayle :has most man of the match awards
MI :has the highest number of match wins
```

In [18]:
```
df_matches.dtypes
```

Out[18]:
```
id                 int64
season             int64
city              object
date              object
team1             object
team2             object
toss_winner       object
toss_decision     object
result            object
dl_applied         int64
winner            object
win_by_runs        int64
win_by_wickets     int64
player_of_match   object
venue             object
umpire1           object
umpire2           object
umpire3           object
dtype: object
```

In [19]:
```
df_matches.nunique()
```

Out[19]:
```
id                756
season             12
city               32
date              546
team1              13
team2              13
toss_winner        13
toss_decision       2
result              3
dl_applied          2
winner             13
win_by_runs        89
win_by_wickets     11
player_of_match   226
venue              41
umpire1            61
umpire2            65
umpire3            25
dtype: int64
```

Full Data Summary

In [20]:
```
df_matches.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 756 entries, 0 to 755
Data columns (total 18 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   id               756 non-null    int64 
 1   season           756 non-null    int64 
 2   city             749 non-null    object
 3   date             756 non-null    object
 4   team1            756 non-null    object
 5   team2            756 non-null    object
 6   toss_winner      756 non-null    object
 7   toss_decision    756 non-null    object
 8   result           756 non-null    object
 9   dl_applied       756 non-null    int64 
 10  winner           752 non-null    object
 11  win_by_runs      756 non-null    int64 
 12  win_by_wickets   756 non-null    int64 
 13  player_of_match  752 non-null    object
 14  venue            756 non-null    object
 15  umpire1          754 non-null    object
 16  umpire2          754 non-null    object
 17  umpire3          119 non-null    object
dtypes: int64(5), object(13)
memory usage: 106.4+ KB
```

In [21]:
```
# Statistical Summary of Data
df_matches.describe()
```

Out[21]:

	id	season	dl_applied	win_by_runs	win_by_wickets
count	756.000000	756.000000	756.000000	756.000000	756.000000
mean	1792.178571	2013.444444	0.025132	13.283069	3.350529
std	3464.478148	3.366895	0.156630	23.471144	3.387963
min	1.000000	2008.000000	0.000000	0.000000	0.000000
25%	189.750000	2011.000000	0.000000	0.000000	0.000000
50%	378.500000	2013.000000	0.000000	0.000000	4.000000
75%	567.250000	2016.000000	0.000000	19.000000	6.000000
max	11415.000000	2019.000000	1.000000	146.000000	10.000000

Observations

In [22]:
```
# The .csv file has data of ipl matches starting from the season 2008 to 2019.
# The biggest margin of victory for the team batting first(win by runs) is 146.
# The biggest victory of the team batting second(win by wickets) is by 10 Wickets.
# 75% of the victorious teams that bat first won by the margin of 19 runs.
# 75% of the victorious teams that bat second won by a margin of 6 Wickets.
# There were 756 Ipl matches hosted from 2008 to 2019.
```

In [23]:
```
#Columns in the data
df_matches.columns
```

Out[23]:
```
Index(['id', 'season', 'city', 'date', 'team1', 'team2', 'toss_winner',
       'toss_decision', 'result', 'dl_applied', 'winner', 'win_by_runs',
       'win_by_wickets', 'player_of_match', 'venue', 'umpire1', 'umpire2',
       'umpire3'],
      dtype='object')
```

Getting Unique values of each column

In [24]:
```
for col in df_matches:
    print(df_matches[col].unique())
```

```
[    1     2     3     4     5     6     7     8     9    10    11    12
    13    14    15    16    17    18    19    20    21    22    23    24
    25    26    27    28    29    30    31    32    33    34    35    36
    37    38    39    40    41    42    43    44    45    46    47    48
    49    50    51    52    53    54    55    56    57    58    59    60
    61    62    63    64    65    66    67    68    69    70    71    72
    73    74    75    76    77    78    79    80    81    82    83    84
    85    86    87    88    89    90    91    92    93    94    95    96
    97    98    99   100   101   102   103   104   105   106   107   108
   109   110   111   112   113   114   115   116   117   118   119   120
   121   122   123   124   125   126   127   128   129   130   131   132
   133   134   135   136   137   138   139   140   141   142   143   144
   145   146   147   148   149   150   151   152   153   154   155   156
   157   158   159   160   161   162   163   164   165   166   167   168
   169   170   171   172   173   174   175   176   177   178   179   180
   181   182   183   184   185   186   187   188   189   190   191   192
   193   194   195   196   197   198   199   200   201   202   203   204
   205   206   207   208   209   210   211   212   213   214   215   216
```

```
   217   218   219   220   221   222   223   224   225   226   227   228
   229   230   231   232   233   234   235   236   237   238   239   240
   241   242   243   244   245   246   247   248   249   250   251   252
   253   254   255   256   257   258   259   260   261   262   263   264
   265   266   267   268   269   270   271   272   273   274   275   276
   277   278   279   280   281   282   283   284   285   286   287   288
   289   290   291   292   293   294   295   296   297   298   299   300
   301   302   303   304   305   306   307   308   309   310   311   312
   313   314   315   316   317   318   319   320   321   322   323   324
   325   326   327   328   329   330   331   332   333   334   335   336
   337   338   339   340   341   342   343   344   345   346   347   348
   349   350   351   352   353   354   355   356   357   358   359   360
   361   362   363   364   365   366   367   368   369   370   371   372
   373   374   375   376   377   378   379   380   381   382   383   384
   385   386   387   388   389   390   391   392   393   394   395   396
   397   398   399   400   401   402   403   404   405   406   407   408
   409   410   411   412   413   414   415   416   417   418   419   420
   421   422   423   424   425   426   427   428   429   430   431   432
   433   434   435   436   437   438   439   440   441   442   443   444
   445   446   447   448   449   450   451   452   453   454   455   456
   457   458   459   460   461   462   463   464   465   466   467   468
   469   470   471   472   473   474   475   476   477   478   479   480
   481   482   483   484   485   486   487   488   489   490   491   492
   493   494   495   496   497   498   499   500   501   502   503   504
   505   506   507   508   509   510   511   512   513   514   515   516
   517   518   519   520   521   522   523   524   525   526   527   528
   529   530   531   532   533   534   535   536   537   538   539   540
   541   542   543   544   545   546   547   548   549   550   551   552
   553   554   555   556   557   558   559   560   561   562   563   564
   565   566   567   568   569   570   571   572   573   574   575   576
   577   578   579   580   581   582   583   584   585   586   587   588
   589   590   591   592   593   594   595   596   597   598   599   600
   601   602   603   604   605   606   607   608   609   610   611   612
   613   614   615   616   617   618   619   620   621   622   623   624
   625   626   627   628   629   630   631   632   633   634   635   636
  7894  7895  7896  7897  7898  7899  7900  7901  7902  7903  7904  7905
  7906  7907  7908  7909  7910  7911  7912  7913  7914  7915  7916  7917
  7918  7919  7920  7921  7922  7923  7924  7925  7926  7927  7928  7929
  7930  7931  7932  7933  7934  7935  7936  7937  7938  7939  7940  7941
  7942  7943  7944  7945  7946  7947  7948  7949  7950  7951  7952  7953
 11137 11138 11139 11140 11141 11142 11143 11144 11145 11146 11147 11148
 11149 11150 11151 11152 11153 11309 11310 11311 11312 11313 11314 11315
 11316 11317 11318 11319 11320 11321 11322 11323 11324 11325 11326 11327
 11328 11329 11330 11331 11332 11333 11334 11335 11336 11337 11338 11339
 11340 11341 11342 11343 11344 11345 11346 11347 11412 11413 11414 11415]
[2017 2008 2009 2010 2011 2012 2013 2014 2015 2016 2018 2019]
['Hyderabad' 'Pune' 'Rajkot' 'Indore' 'Bangalore' 'Mumbai' 'Kolkata'
 'Delhi' 'Chandigarh' 'Kanpur' 'Jaipur' 'Chennai' 'Cape Town'
 'Port Elizabeth' 'Durban' 'Centurion' 'East London' 'Johannesburg'
 'Kimberley' 'Bloemfontein' 'Ahmedabad' 'Cuttack' 'Nagpur' 'Dharamsala'
 'Kochi' 'Visakhapatnam' 'Raipur' 'Ranchi' 'Abu Dhabi' 'Sharjah' nan
 'Mohali' 'Bengaluru']
['2017-04-05' '2017-04-06' '2017-04-07' '2017-04-08' '2017-04-09'
 '2017-04-10' '2017-04-11' '2017-04-12' '2017-04-13' '2017-04-14'
 '2017-04-15' '2017-04-16' '2017-04-17' '2017-04-18' '2017-04-19'
 '2017-04-20' '2017-04-21' '2017-04-22' '2017-04-23' '2017-04-24'
 '2017-04-26' '2017-04-27' '2017-04-28' '2017-04-29' '2017-04-30'
 '2017-05-01' '2017-05-02' '2017-05-03' '2017-05-04' '2017-05-05'
 '2017-05-06' '2017-05-07' '2017-05-08' '2017-05-09' '2017-05-10'
 '2017-05-11' '2017-05-12' '2017-05-13' '2017-05-14' '2017-05-16'
 '2017-05-17' '2017-05-19' '2017-05-21' '2008-04-18' '2008-04-19'
 '2008-04-20' '2008-04-21' '2008-04-22' '2008-04-23' '2008-04-24'
 '2008-04-25' '2008-04-26' '2008-04-27' '2008-04-28' '2008-04-29'
 '2008-04-30' '2008-05-01' '2008-05-02' '2008-05-25' '2008-05-03'
 '2008-05-04' '2008-05-05' '2008-05-06' '2008-05-07' '2008-05-08'
 '2008-05-09' '2008-05-28' '2008-05-10' '2008-05-11' '2008-05-12'
 '2008-05-13' '2008-05-14' '2008-05-15' '2008-05-16' '2008-05-17'
 '2008-05-18' '2008-05-19' '2008-05-20' '2008-05-21' '2008-05-23'
 '2008-05-24' '2008-05-26' '2008-05-27' '2008-05-30' '2008-05-31'
 '2008-06-01' '2009-04-18' '2009-04-19' '2009-04-20' '2009-04-21'
 '2009-04-22' '2009-04-23' '2009-04-24' '2009-04-25' '2009-04-26'
 '2009-04-27' '2009-04-28' '2009-04-29' '2009-04-30' '2009-05-01'
 '2009-05-02' '2009-05-03' '2009-05-04' '2009-05-05' '2009-05-06'
 '2009-05-07' '2009-05-08' '2009-05-09' '2009-05-10' '2009-05-11'
 '2009-05-12' '2009-05-13' '2009-05-14' '2009-05-15' '2009-05-16'
 '2009-05-17' '2009-05-18' '2009-05-19' '2009-05-20' '2009-05-21'
 '2009-05-22' '2009-05-23' '2009-05-24' '2010-03-12' '2010-03-13'
 '2010-03-14' '2010-03-15' '2010-03-16' '2010-03-17' '2010-03-18'
 '2010-03-19' '2010-03-20' '2010-03-21' '2010-03-22' '2010-03-23'
 '2010-03-24' '2010-03-25' '2010-03-26' '2010-03-27' '2010-03-28'
 '2010-03-29' '2010-03-30' '2010-03-31' '2010-04-01' '2010-04-02'
 '2010-04-03' '2010-04-04' '2010-04-05' '2010-04-06' '2010-04-07'
 '2010-04-08' '2010-04-09' '2010-04-10' '2010-04-11' '2010-04-12'
 '2010-04-13' '2010-04-14' '2010-04-15' '2010-04-16' '2010-04-17'
 '2010-04-18' '2010-04-19' '2010-04-21' '2010-04-22' '2010-04-24'
 '2010-04-25' '2011-04-08' '2011-04-09' '2011-04-10' '2011-04-11'
 '2011-04-12' '2011-04-13' '2011-04-14' '2011-04-15' '2011-04-16'
 '2011-04-17' '2011-04-18' '2011-04-19' '2011-04-20' '2011-04-21'
 '2011-04-22' '2011-04-23' '2011-04-24' '2011-04-25' '2011-04-26'
```

```
 '2011-04-27' '2011-04-28' '2011-04-29' '2011-04-30' '2011-05-01'
 '2011-05-02' '2011-05-03' '2011-05-04' '2011-05-05' '2011-05-06'
 '2011-05-07' '2011-05-08' '2011-05-09' '2011-05-10' '2011-05-11'
 '2011-05-12' '2011-05-13' '2011-05-14' '2011-05-15' '2011-05-16'
 '2011-05-17' '2011-05-18' '2011-05-19' '2011-05-20' '2011-05-21'
 '2011-05-22' '2011-05-24' '2011-05-25' '2011-05-27' '2011-05-28'
 '2012-04-04' '2012-04-05' '2012-04-06' '2012-04-07' '2012-04-08'
 '2012-04-09' '2012-04-10' '2012-04-11' '2012-04-12' '2012-04-13'
 '2012-04-19' '2012-04-14' '2012-04-15' '2012-04-16' '2012-04-17'
 '2012-04-18' '2012-05-10' '2012-04-20' '2012-04-21' '2012-04-22'
 '2012-04-23' '2012-04-24' '2012-04-25' '2012-04-26' '2012-04-27'
 '2012-04-28' '2012-04-29' '2012-04-30' '2012-05-01' '2012-05-02'
 '2012-05-03' '2012-05-04' '2012-05-05' '2012-05-06' '2012-05-07'
 '2012-05-08' '2012-05-09' '2012-05-11' '2012-05-12' '2012-05-13'
 '2012-05-14' '2012-05-15' '2012-05-16' '2012-05-17' '2012-05-18'
 '2012-05-19' '2012-05-20' '2012-05-22' '2012-05-23' '2012-05-25'
 '2012-05-27' '2013-04-03' '2013-04-04' '2013-04-05' '2013-04-06'
 '2013-04-07' '2013-04-08' '2013-04-09' '2013-04-10' '2013-04-11'
 '2013-04-12' '2013-04-13' '2013-04-14' '2013-04-15' '2013-04-16'
 '2013-04-17' '2013-04-18' '2013-04-19' '2013-04-20' '2013-04-21'
 '2013-04-22' '2013-04-23' '2013-05-16' '2013-04-24' '2013-04-25'
 '2013-04-26' '2013-04-27' '2013-04-28' '2013-04-29' '2013-04-30'
 '2013-05-01' '2013-05-02' '2013-05-03' '2013-05-04' '2013-05-14'
 '2013-05-05' '2013-05-07' '2013-05-08' '2013-05-09' '2013-05-10'
 '2013-05-11' '2013-05-12' '2013-05-13' '2013-05-15' '2013-05-06'
 '2013-05-17' '2013-05-18' '2013-05-19' '2013-05-21' '2013-05-22'
 '2013-05-24' '2013-05-26' '2014-04-16' '2014-04-17' '2014-04-18'
 '2014-04-19' '2014-04-20' '2014-04-21' '2014-04-22' '2014-04-23'
 '2014-04-24' '2014-04-25' '2014-04-26' '2014-04-27' '2014-04-28'
 '2014-04-29' '2014-04-30' '2014-05-02' '2014-05-03' '2014-05-04'
 '2014-05-05' '2014-05-06' '2014-05-07' '2014-05-08' '2014-05-09'
 '2014-05-10' '2014-05-11' '2014-05-12' '2014-05-13' '2014-05-14'
 '2014-05-15' '2014-05-18' '2014-05-19' '2014-05-20' '2014-05-21'
 '2014-05-22' '2014-05-23' '2014-05-24' '2014-05-25' '2014-05-27'
 '2014-05-28' '2014-05-30' '2014-06-01' '2015-04-08' '2015-04-09'
 '2015-04-10' '2015-04-11' '2015-04-12' '2015-04-13' '2015-04-14'
 '2015-04-30' '2015-04-15' '2015-04-16' '2015-04-17' '2015-04-18'
 '2015-04-19' '2015-04-20' '2015-04-21' '2015-04-22' '2015-04-23'
 '2015-04-24' '2015-04-25' '2015-04-26' '2015-04-27' '2015-05-07'
 '2015-04-29' '2015-04-28' '2015-05-01' '2015-05-02' '2015-05-03'
 '2015-05-04' '2015-05-05' '2015-05-06' '2015-05-08' '2015-05-09'
 '2015-05-10' '2015-05-11' '2015-05-12' '2015-05-13' '2015-05-14'
 '2015-05-15' '2015-05-16' '2015-05-17' '2015-05-19' '2015-05-20'
 '2015-05-22' '2015-05-24' '2016-04-09' '2016-04-10' '2016-04-11'
 '2016-04-12' '2016-04-13' '2016-04-14' '2016-04-15' '2016-04-16'
 '2016-04-17' '2016-04-18' '2016-04-19' '2016-04-20' '2016-04-21'
 '2016-04-22' '2016-04-23' '2016-04-24' '2016-04-25' '2016-04-26'
 '2016-04-27' '2016-04-28' '2016-04-29' '2016-04-30' '2016-05-01'
 '2016-05-02' '2016-05-03' '2016-05-04' '2016-05-05' '2016-05-06'
 '2016-05-07' '2016-05-08' '2016-05-09' '2016-05-10' '2016-05-11'
 '2016-05-12' '2016-05-13' '2016-05-14' '2016-05-15' '2016-05-16'
 '2016-05-17' '2016-05-18' '2016-05-19' '2016-05-20' '2016-05-21'
 '2016-05-22' '2016-05-24' '2016-05-25' '2016-05-27' '2016-05-29'
 '07/04/18' '08/04/18' '09/04/18' '10/04/18' '11/04/18' '12/04/18'
 '13/04/18' '14/04/18' '15/04/18' '16/04/18' '17/04/18' '18/04/18'
 '19/04/18' '20/04/18' '21/04/18' '22/04/18' '23/04/18' '24/04/18'
 '25/04/18' '26/04/18' '27/04/18' '28/04/18' '29/04/18' '30/04/18'
 '01/05/18' '02/05/18' '03/05/18' '04/05/18' '05/05/18' '06/05/18'
 '07/05/18' '08/05/18' '09/05/18' '10/05/18' '11/05/18' '12/05/18'
 '13/05/18' '14/05/18' '15/05/18' '16/05/18' '17/05/18' '18/05/18'
 '19/05/18' '20/05/18' '22/05/18' '23/05/18' '25/05/18' '27/05/18'
 '23/03/19' '24/03/19' '25/03/19' '26/03/19' '27/03/19' '28/03/19'
 '29/03/19' '30/03/19' '31/03/19' '01/04/19' '02/04/19' '03/04/19'
 '04/04/19' '05/04/19' '06/04/19' '07/04/19' '08/04/19' '09/04/19'
 '10/04/19' '11/04/19' '12/04/19' '13/04/19' '14/04/19' '15/04/19'
 '16/04/19' '17/04/19' '18/04/19' '19/04/19' '20/04/19' '21/04/19'
 '22/04/19' '23/04/19' '24/04/19' '25/04/19' '26/04/19' '27/04/19'
 '28/04/19' '29/04/19' '30/04/19' '01/05/19' '02/05/19' '03/05/19'
 '04/05/19' '05/05/19' '07/05/19' '08/05/19' '10/05/19' '12/05/19']
['SRH' 'MI' 'GL' 'PW' 'RCB' 'KKR' 'DD' 'KXIP' 'CSK' 'RR' 'DC' 'KTK' 'RPS']
['RCB' 'PW' 'KKR' 'KXIP' 'DD' 'SRH' 'MI' 'GL' 'RR' 'CSK' 'DC' 'KTK' 'RPS']
['RCB' 'PW' 'KKR' 'KXIP' 'SRH' 'MI' 'GL' 'DD' 'CSK' 'RR' 'DC' 'KTK' 'RPS']
['field' 'bat']
['normal' 'tie' 'no result']
[0 1]
['SRH' 'PW' 'KKR' 'KXIP' 'RCB' 'MI' 'DD' 'GL' 'CSK' 'RR' 'DC' 'KTK' nan
 'RPS']
[ 35   0  15  97  17  51  27   5  21  14  26  82   3  61  48  19  12 146
   7   9  10  20   1 140  33   6  66  13  45  29  18  23  41  65  25 105
  75  92  11  24  38   8  78  16  53   2   4  31  55  98  34  36  39  40
  67  63  37  57  22  85  32  76 111  43  58  28  74  42  59  46  47  86
  44  87 130  60  77  30  50  93  72  62 138  71 144  80  64 102 118]
[ 0  7 10  6  9  4  8  5  2  3  1]
['Yuvraj Singh' 'SPD Smith' 'CA Lynn' 'GJ Maxwell' 'KM Jadhav'
 'Rashid Khan' 'N Rana' 'AR Patel' 'SV Samson' 'JJ Bumrah' 'SP Narine'
 'KA Pollard' 'AJ Tye' 'RV Uthappa' 'CJ Anderson' 'BA Stokes'
 'NM Coulter-Nile' 'B Kumar' 'CH Gayle' 'KS Williamson' 'JC Buttler'
 'SK Raina' 'MJ McClenaghan' 'MS Dhoni' 'HM Amla' 'G Gambhir'
 'LH Ferguson' 'KH Pandya' 'Sandeep Sharma' 'DA Warner' 'RG Sharma'
```

```
 'Mohammed Shami' 'RA Tripathi' 'RR Pant' 'JD Unadkat' 'LMP Simmons'
 'DR Smith' 'S Dhawan' 'MM Sharma' 'SS Iyer' 'WP Saha' 'KK Nair'
 'Mohammed Siraj' 'AT Rayudu' 'HV Patel' 'Washington Sundar' 'KV Sharma'
 'BB McCullum' 'MEK Hussey' 'MF Maharoof' 'MV Boucher' 'DJ Hussey'
 'SR Watson' 'V Sehwag' 'ML Hayden' 'YK Pathan' 'KC Sangakkara' 'JDP Oram'
 'AC Gilchrist' 'SM Katich' 'ST Jayasuriya' 'GD McGrath' 'SE Marsh'
 'SA Asnodkar' 'R Vinay Kumar' 'IK Pathan' 'SM Pollock' 'Sohail Tanvir'
 'S Sreesanth' 'A Nehra' 'SC Ganguly' 'CRD Fernando' 'L Balaji'
 'Shoaib Akhtar' 'A Mishra' 'DPMD Jayawardene' 'GC Smith' 'DJ Bravo'
 'M Ntini' 'SP Goswami' 'A Kumble' 'KD Karthik' 'JA Morkel' 'P Kumar'
 'Umar Gul' 'SR Tendulkar' 'R Dravid' 'DL Vettori' 'RP Singh'
 'M Muralitharan' 'AB de Villiers' 'RS Bopara' 'PP Ojha' 'TM Dilshan'
 'HH Gibbs' 'DP Nannes' 'JP Duminy' 'SB Jakati' 'JH Kallis' 'A Singh'
 'S Badrinath' 'LRPL Taylor' 'Harbhajan Singh' 'R Bhatia' 'SK Warne'
 'B Lee' 'BJ Hodge' 'LR Shukla' 'MK Pandey' 'AD Mathews' 'MK Tiwary'
 'WPUJC Vaas' 'A Symonds' 'AA Jhunjhunwala' 'J Theron' 'AC Voges'
 'NV Ojha' 'SL Malinga' 'M Vijay' 'KP Pietersen' 'PD Collingwood'
 'MJ Lumb' 'TL Suman' 'RJ Harris' 'PP Chawla' 'Harmeet Singh' 'R Ashwin'
 'R McLaren' 'M Kartik' 'DE Bollinger' 'S Anirudha' 'SK Trivedi' 'SB Wagh'
 'PC Valthaty' 'MD Mishra' 'DW Steyn' 'S Sohal' 'MM Patel' 'V Kohli'
 'I Sharma' 'J Botha' 'Iqbal Abdulla' 'P Parameswaran' 'R Sharma'
 'MR Marsh' 'BA Bhatt' 'S Aravind' nan 'JEC Franklin' 'RE Levi'
 'AM Rahane' 'RA Jadeja' 'MN Samuels' 'M Morkel' 'F du Plessis'
 'AD Mascarenhas' 'Shakib Al Hasan' 'JD Ryder' 'S Nadeem'
 'KMDN Kulasekara' 'CL White' 'Mandeep Singh' 'P Negi' 'Azhar Mahmood'
 'BW Hilfenhaus' 'A Chandila' 'UT Yadav' 'MS Bisla' 'M Vohra' 'GH Vihari'
 'AJ Finch' 'JP Faulkner' 'MS Gony' 'DA Miller' 'DJG Sammy' 'MG Johnson'
 'KK Cooper' 'PA Patel' 'AP Tare' 'LJ Wright' 'YS Chahal' 'PV Tambe'
 'DJ Hooda' 'GJ Bailey' 'AD Russell' 'MA Agarwal' 'MA Starc' 'VR Aaron'
 'TA Boult' 'EJG Morgan' 'HH Pandya' 'MC Henriques' 'Z Khan' 'Q de Kock'
 'Mustafizur Rahman' 'SA Yadav' 'AB Dinda' 'CH Morris' 'CR Brathwaite'
 'MP Stoinis' 'A Zampa' 'BCJ Cutting' 'KL Rahul' 'SW Billings' 'JJ Roy'
 'B Stanlake' 'J Archer' 'AS Rajpoot' 'TG Southee' 'AS Yadav'
 'M Ur Rahman' 'Ishan Kishan' 'Kuldeep Yadav' 'S Gopal' 'L Ngidi' 'P Shaw'
 'J Bairstow' 'S Curran' 'A Joseph' 'K Rabada' 'H Gurney' 'DL Chahar'
 'Imran Tahir' 'K Paul' 'K Ahmed' 'S Gill' 'S Hetmyer']
['Rajiv Gandhi International Stadium, Uppal'
 'Maharashtra Cricket Association Stadium'
 'Saurashtra Cricket Association Stadium' 'Holkar Cricket Stadium'
 'M Chinnaswamy Stadium' 'Wankhede Stadium' 'Eden Gardens'
 'Feroz Shah Kotla' 'Punjab Cricket Association IS Bindra Stadium, Mohali'
 'Green Park' 'Punjab Cricket Association Stadium, Mohali'
 'Sawai Mansingh Stadium' 'MA Chidambaram Stadium, Chepauk'
 'Dr DY Patil Sports Academy' 'Newlands' "St George's Park" 'Kingsmead'
 'SuperSport Park' 'Buffalo Park' 'New Wanderers Stadium'
 'De Beers Diamond Oval' 'OUTsurance Oval' 'Brabourne Stadium'
 'Sardar Patel Stadium, Motera' 'Barabati Stadium'
 'Vidarbha Cricket Association Stadium, Jamtha'
 'Himachal Pradesh Cricket Association Stadium' 'Nehru Stadium'
 'Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket Stadium'
 'Subrata Roy Sahara Stadium'
 'Shaheed Veer Narayan Singh International Stadium'
 'JSCA International Stadium Complex' 'Sheikh Zayed Stadium'
 'Sharjah Cricket Stadium' 'Dubai International Cricket Stadium'
 'M. A. Chidambaram Stadium' 'Feroz Shah Kotla Ground'
 'M. Chinnaswamy Stadium' 'Rajiv Gandhi Intl. Cricket Stadium'
 'IS Bindra Stadium' 'ACA-VDCA Stadium']
['AY Dandekar' 'A Nand Kishore' 'Nitin Menon' 'AK Chaudhary' nan
 'A Deshmukh' 'KN Ananthapadmanabhan' 'YC Barde' 'S Ravi' 'CB Gaffaney'
 'M Erasmus' 'NJ Llong' 'CK Nandan' 'Asad Rauf' 'MR Benson' 'Aleem Dar'
 'SJ Davis' 'BF Bowden' 'IL Howell' 'DJ Harper' 'RE Koertzen'
 'BR Doctrove' 'AV Jayaprakash' 'BG Jerling' 'HDPK Dharmasena' 'S Asnani'
 'GAV Baxter' 'SS Hazare' 'K Hariharan' 'SL Shastri' 'SK Tarapore'
 'SJA Taufel' 'S Das' 'AM Saheba' 'PR Reiffel' 'JD Cloete' 'VA Kulkarni'
 'BNJ Oxenford' 'C Shamshuddin' 'RK Illingworth' 'RM Deshpande'
 'K Srinath' 'SD Fry' 'PG Pathak' 'K Bharatan' 'Chris Gaffaney'
 'Rod Tucker' 'Nigel Llong' 'Anil Chaudhary' 'K Ananthapadmanabhan'
 'O Nandan' 'A Nanda Kishore' 'Vineet Kulkarni' 'Bruce Oxenford'
 'Marais Erasmus' 'Kumar Dharmasena' 'Anil Dandekar' 'Yeshwant Barde'
 'Ian Gould' 'Ulhas Gandhe' 'Nanda Kishore' 'Sundaram Ravi']
['NJ Llong' 'S Ravi' 'CK Nandan' 'C Shamshuddin' nan 'AK Chaudhary'
 'Nitin Menon' 'A Deshmukh' 'VK Sharma' 'M Erasmus' 'CB Gaffaney'
 'A Nand Kishore' 'RE Koertzen' 'SL Shastri' 'GA Pratapkumar' 'DJ Harper'
 'K Hariharan' 'RB Tiffin' 'AM Saheba' 'MR Benson' 'IL Howell'
 'AV Jayaprakash' 'I Shivram' 'BR Doctrove' 'BG Jerling' 'SJ Davis'
 'SD Ranade' 'SJA Taufel' 'TH Wijewardene' 'SK Tarapore' 'HDPK Dharmasena'
 'SS Hazare' 'PR Reiffel' 'AL Hill' 'RJ Tucker' 'VA Kulkarni' 'JD Cloete'
 'BNJ Oxenford' 'S Asnani' 'S Das' 'K Srinath' 'Subroto Das'
 'RK Illingworth' 'PG Pathak' 'K Srinivasan' 'SD Fry' 'A Nanda Kishore'
 'K Ananthapadmanabhan' 'A.D Deshmukh' 'Vineet Kulkarni' 'Chris Gaffaney'
 'Rod Tucker' 'Nigel Llong' 'Anil Chaudhary' 'O Nandan'
 'Virender Kumar Sharma' 'Yeshwant Barde' 'Anil Dandekar'
 'Kumar Dharmasena' 'KN Anantapadmanabhan' 'Ulhas Gandhe' 'Nanda Kishore'
 'Bruce Oxenford' 'Nand Kishore' 'KN Ananthapadmanabhan' 'Ian Gould']
[nan 'Anil Chaudhary' 'Nitin Menon' 'S Ravi' 'O Nandan' 'A Nanda Kishore'
 'Vineet Kulkarni' 'C Shamshuddin' 'Rod Tucker' 'Chris Gaffaney'
 'A.D Deshmukh' 'Nigel Llong' 'K Ananthapadmanabhan' 'Anil Dandekar'
 'Virender Kumar Sharma' 'Yeshwant Barde' 'Bruce Oxenford'
 'Marais Erasmus' 'Kumar Dharmasena' 'KN Anantapadmanabhan' 'Ulhas Gandhe'
```

```
'Nanda Kishore' 'Ian Gould' 'Sundaram Ravi' 'KN Ananthapadmanabhan'
'Chettithody Shamshuddin']
```

Finding out Null values in Each Columns

In [25]:
```
df_matches.isnull().sum()
```

Out[25]:
```
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
```

Dropping of Columns having signigicant number of Null values

In [28]:
```
df_matches1 =df_matches.drop(columns=['umpire3'], axis =1)
df_matches1
```

Out[28]:

	id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wi
0	1	2017	Hyderabad	2017-04-05	SRH	RCB	RCB	field	normal	0	SRH	35	
1	2	2017	Pune	2017-04-06	MI	PW	PW	field	normal	0	PW	0	
2	3	2017	Rajkot	2017-04-07	GL	KKR	KKR	field	normal	0	KKR	0	
3	4	2017	Indore	2017-04-08	PW	KXIP	KXIP	field	normal	0	KXIP	0	
4	5	2017	Bangalore	2017-04-08	RCB	DD	RCB	bat	normal	0	RCB	15	
...	...	...	...	...	...	...	...	...	...	...	...	...	
751	11347	2019	Mumbai	05/05/19	KKR	MI	MI	field	normal	0	MI	0	
752	11412	2019	Chennai	07/05/19	CSK	MI	CSK	bat	normal	0	MI	0	
753	11413	2019	Visakhapatnam	08/05/19	SRH	DC	DC	field	normal	0	DC	0	
754	11414	2019	Visakhapatnam	10/05/19	DC	CSK	CSK	field	normal	0	CSK	0	
755	11415	2019	Hyderabad	12/05/19	MI	CSK	MI	bat	normal	0	MI	1	

756 rows × 17 columns

In [30]:
```
df_matches.isnull().sum()
```

Out[30]:
```
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
```

In [32]:
```
df_matches.fillna(0,inplace=True)
df_matches
```

Out[32]:

	id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wi
0	1	2017	Hyderabad	2017-04-05	SRH	RCB	RCB	field	normal	0	SRH	35	
1	2	2017	Pune	2017-04-06	MI	PW	PW	field	normal	0	PW	0	
2	3	2017	Rajkot	2017-04-07	GL	KKR	KKR	field	normal	0	KKR	0	
3	4	2017	Indore	2017-04-08	PW	KXIP	KXIP	field	normal	0	KXIP	0	
4	5	2017	Bangalore	2017-04-08	RCB	DD	RCB	bat	normal	0	RCB	15	
...	...	...	...	...	...	...	...	...	...	...	...	...	
751	11347	2019	Mumbai	05/05/19	KKR	MI	MI	field	normal	0	MI	0	
752	11412	2019	Chennai	07/05/19	CSK	MI	CSK	bat	normal	0	MI	0	
753	11413	2019	Visakhapatnam	08/05/19	SRH	DC	DC	field	normal	0	DC	0	
754	11414	2019	Visakhapatnam	10/05/19	DC	CSK	CSK	field	normal	0	CSK	0	
755	11415	2019	Hyderabad	12/05/19	MI	CSK	MI	bat	normal	0	MI	1	

756 rows × 18 columns

In [33]:
```
df_matches.isnull().sum()
```

Out[33]:
```
id                0
season            0
city              0
date              0
team1             0
team2             0
toss_winner       0
toss_decision     0
result            0
dl_applied        0
winner            0
win_by_runs       0
win_by_wickets    0
player_of_match   0
venue             0
umpire1           0
umpire2           0
umpire3           0
dtype: int64
```

In [35]:
```
# We have successfully replace the Null values with 'Zeros'
```

In [36]:
```
df_deliveries.dtypes
```

Out[36]:
```
match_id             int64
inning               int64
batting_team        object
bowling_team        object
over                 int64
ball                 int64
batsman             object
non_striker         object
bowler              object
is_super_over        int64
wide_runs            int64
bye_runs             int64
legbye_runs          int64
noball_runs          int64
penalty_runs         int64
batsman_runs         int64
extra_runs           int64
total_runs           int64
player_dismissed    object
dismissal_kind      object
fielder             object
dtype: object
```

In [37]:
```
df_deliveries.shape
```

Out[37]:
```
(179078, 21)
```

In [38]:
```
df_deliveries.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
```

In [39]:
```
df_deliveries.describe()
```

Out[39]:

	match_id	inning	over	ball	is_super_over	wide_runs	bye_runs	legbye_runs	noball_r
count	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.000000	179078.000
mean	1802.252957	1.482952	10.162488	3.615587	0.000452	0.036721	0.004936	0.021136	0.004
std	3472.322805	0.502074	5.677684	1.806966	0.021263	0.251161	0.116480	0.194908	0.070
min	1.000000	1.000000	1.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000
25%	190.000000	1.000000	5.000000	2.000000	0.000000	0.000000	0.000000	0.000000	0.000
50%	379.000000	1.000000	10.000000	4.000000	0.000000	0.000000	0.000000	0.000000	0.000
75%	567.000000	2.000000	15.000000	5.000000	0.000000	0.000000	0.000000	0.000000	0.000
max	11415.000000	5.000000	20.000000	9.000000	1.000000	5.000000	4.000000	5.000000	5.000

In [40]:
```
df_deliveries.columns
```

Out[40]:
```
Index(['match_id', 'inning', 'batting_team', 'bowling_team', 'over', 'ball',
       'batsman', 'non_striker', 'bowler', 'is_super_over', 'wide_runs',
       'bye_runs', 'legbye_runs', 'noball_runs', 'penalty_runs',
       'batsman_runs', 'extra_runs', 'total_runs', 'player_dismissed',
       'dismissal_kind', 'fielder'],
      dtype='object')
```

In [41]:
```
df_deliveries.isnull().sum()
```

Out[41]:
```
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
```

Total number of Nan values in the dataset

In [42]:
```
df_deliveries.isnull().sum().sum()
```

Out[42]:
```
513118
```

Analysing the data

Which Team had won by maximum runs?

In [43]:
```
df_matches.iloc[df_matches['win_by_runs'].idxmax()]
```

Out[43]:
```
id                              44
season                        2017
city                         Delhi
date                    2017-05-06
team1                           MI
team2                           DD
toss_winner                     DD
toss_decision                field
result                      normal
dl_applied                       0
winner                          MI
win_by_runs                    146
win_by_wickets                   0
player_of_match        LMP Simmons
venue              Feroz Shah Kotla
umpire1                Nitin Menon
umpire2                  CK Nandan
umpire3                          0
Name: 43, dtype: object
```

Which Team had won by maximum wicket?

In [44]:
```
df_matches.iloc[df_matches['win_by_wickets'].idxmax()]
```

Out[44]:
```
id                                                3
season                                         2017
city                                          Rajkot
date                                      2017-04-07
team1                                             GL
team2                                            KKR
toss_winner                                      KKR
toss_decision                                  field
result                                        normal
dl_applied                                         0
winner                                           KKR
win_by_runs                                        0
win_by_wickets                                    10
player_of_match                           CA Lynn
venue              Saurashtra Cricket Association Stadium
umpire1                                   Nitin Menon
umpire2                                     CK Nandan
umpire3                                            0
Name: 2, dtype: object
```

Which Team had won by minimum Margin?

In [45]:
```
df_matches.iloc[df_matches[df_matches['win_by_runs'].ge(1)].win_by_runs.idxmin()]
```

Out[45]:
```
id                                                    59
season                                              2017
city                                           Hyderabad
date                                          2017-05-21
team1                                                 MI
team2                                                 PW
toss_winner                                           MI
toss_decision                                        bat
result                                            normal
dl_applied                                             0
winner                                                MI
win_by_runs                                            1
win_by_wickets                                         0
player_of_match                                KH Pandya
venue              Rajiv Gandhi International Stadium, Uppal
umpire1                                        NJ Llong
umpire2                                          S Ravi
umpire3                                                0
Name: 58, dtype: object
```

Which Team had won by Minimum Wickets?

In [46]:
```
df_matches.iloc[df_matches[df_matches['win_by_wickets'].ge(1)].win_by_wickets.idxmin()]
```

Out[46]:
```
id                           560
season                      2015
city                     Kolkata
date                  2015-05-09
team1                       KXIP
team2                        KKR
toss_winner                 KXIP
toss_decision                bat
result                    normal
dl_applied                     0
winner                       KKR
win_by_runs                    0
win_by_wickets                 1
player_of_match        AD Russell
venue                Eden Gardens
umpire1              AK Chaudhary
umpire2           HDPK Dharmasena
umpire3                        0
Name: 559, dtype: object
```

In [49]:
```
len(df_matches['season'].unique())
```

Out[49]:
```
12
```

In [50]:
```
df_deliveries.fillna(0,inplace=True)
df_deliveries
```

Out[50]:

	match_id	inning	batting_team	bowling_team	over	ball	batsman	non_striker	bowler	is_super_over	...	bye_runs	legbye_runs
0	1	1	SRH	RCB	1	1	DA Warner	S Dhawan	TS Mills	0	...	0	0
1	1	1	SRH	RCB	1	2	DA Warner	S Dhawan	TS Mills	0	...	0	0
2	1	1	SRH	RCB	1	3	DA Warner	S Dhawan	TS Mills	0	...	0	0
3	1	1	SRH	RCB	1	4	DA Warner	S Dhawan	TS Mills	0	...	0	0
4	1	1	SRH	RCB	1	5	DA Warner	S Dhawan	TS Mills	0	...	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
179073	11415	2	CSK	MI	20	2	RA Jadeja	SR Watson	SL Malinga	0	...	0	0
179074	11415	2	CSK	MI	20	3	SR Watson	RA Jadeja	SL Malinga	0	...	0	0
179075	11415	2	CSK	MI	20	4	SR Watson	RA Jadeja	SL Malinga	0	...	0	0
179076	11415	2	CSK	MI	20	5	SN Thakur	RA Jadeja	SL Malinga	0	...	0	0
179077	11415	2	CSK	MI	20	6	SN Thakur	RA Jadeja	SL Malinga	0	...	0	0

179078 rows × 21 columns

In [51]:
```
df_deliveries.isnull().sum()
```

Out[51]:
```
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    0
dismissal_kind      0
fielder             0
dtype: int64
```

The team with most number of Wins per season

In [52]:
```
teams_per_season =df_matches.groupby('season')['winner'].value_counts()
teams_per_season
```

Out[52]:
```
season  winner
2008    RR        13
        KXIP      10
        CSK        9
        DD         7
        MI         7
                  ..
2019    KXIP       6
        SRH        6
        RCB        5
        RR         5
        0          1
Name: winner, Length: 103, dtype: int64
```

In [53]:
```
"""
for i, w in wins_per_season.iteritmes():
     print(i, w)
for items in win_per_season.iteritems():
     print(items)
"""
year = 2008
win_per_season_df_matches=pd.DataFrame(columns=['year', 'team', 'wins'])
for items in teams_per_season.iteritems():
    if items[0][0]==year:
        print(items)
        win_series =pd.DataFrame({
            'year': [items[0][0]],
            'team':[items[0][1]],
            'wins':[items[1]]
        })
        win_per_season_df_matches = win_per_season_df_matches.append(win_series)
        year +=1
```

```
((2008, 'RR'), 13)
((2009, 'DD'), 10)
((2010, 'MI'), 11)
((2011, 'CSK'), 11)
((2012, 'KKR'), 12)
((2013, 'MI'), 13)
((2014, 'KXIP'), 12)
((2015, 'CSK'), 10)
((2016, 'SRH'), 11)
((2017, 'MI'), 12)
((2018, 'CSK'), 11)
((2019, 'MI'), 11)
```

In [54]:
```
win_per_season_df_matches
```

Out[54]:

	year	team	wins
0	2008	RR	13
0	2009	DD	10
0	2010	MI	11
0	2011	CSK	11
0	2012	KKR	12
0	2013	MI	13
0	2014	KXIP	12
0	2015	CSK	10
0	2016	SRH	11
0	2017	MI	12
0	2018	CSK	11
0	2019	MI	11

DATA Visualising

In [55]:
```
venue_ser=df_matches['venue'].value_counts()
```

In [56]:
```
venue_df_matches =pd.DataFrame(columns=['venue', 'matches'])
for items in venue_ser.iteritems():
    temp_df_matches =pd.DataFrame({
        'venue':[items[0]],
        'matches':[items[1]]
    })

    venue_df_matches = venue_df_matches.append(temp_df_matches,ignore_index=True)
```

Number of Matches played and venue

In [58]:
```
venue_df_matches
```

Out[58]:

	venue	matches
0	Eden Gardens	77
1	M Chinnaswamy Stadium	73
2	Wankhede Stadium	73
3	Feroz Shah Kotla	67
4	Rajiv Gandhi International Stadium, Uppal	56
5	MA Chidambaram Stadium, Chepauk	49
6	Sawai Mansingh Stadium	47
7	Punjab Cricket Association Stadium, Mohali	35
8	Maharashtra Cricket Association Stadium	21
9	Subrata Roy Sahara Stadium	17
10	Dr DY Patil Sports Academy	17
11	Kingsmead	15
12	Punjab Cricket Association IS Bindra Stadium, ...	14
13	SuperSport Park	12
14	Sardar Patel Stadium, Motera	12
15	Dr. Y.S. Rajasekhara Reddy ACA-VDCA Cricket St...	11
16	Brabourne Stadium	11
17	Saurashtra Cricket Association Stadium	10
18	Holkar Cricket Stadium	9
19	Himachal Pradesh Cricket Association Stadium	9
20	Rajiv Gandhi Intl. Cricket Stadium	8
21	M. A. Chidambaram Stadium	8
22	New Wanderers Stadium	8
23	Feroz Shah Kotla Ground	7
24	Barabati Stadium	7
25	M. Chinnaswamy Stadium	7
26	St George's Park	7
27	Newlands	7
28	JSCA International Stadium Complex	7
29	Sheikh Zayed Stadium	7
30	Dubai International Cricket Stadium	7
31	IS Bindra Stadium	7
32	Shaheed Veer Narayan Singh International Stadium	6
33	Sharjah Cricket Stadium	6
34	Nehru Stadium	5
35	Green Park	4
36	De Beers Diamond Oval	3
37	Vidarbha Cricket Association Stadium, Jamtha	3
38	Buffalo Park	3
39	OUTsurance Oval	2
40	ACA-VDCA Stadium	2

The most Successful IPL team

In [59]:
```
team_wins_ser= df_matches['winner'].value_counts()
team_wins_df_matches=pd.DataFrame(columns=['team', 'wins'])
for items in team_wins_ser.iteritems():
    temp_df1 =pd.DataFrame({
        'team':[items[0]],
        'wins':[items[1]]
    })
    team_wins_df_matches= team_wins_df_matches.append(temp_df1,ignore_index=True)
```

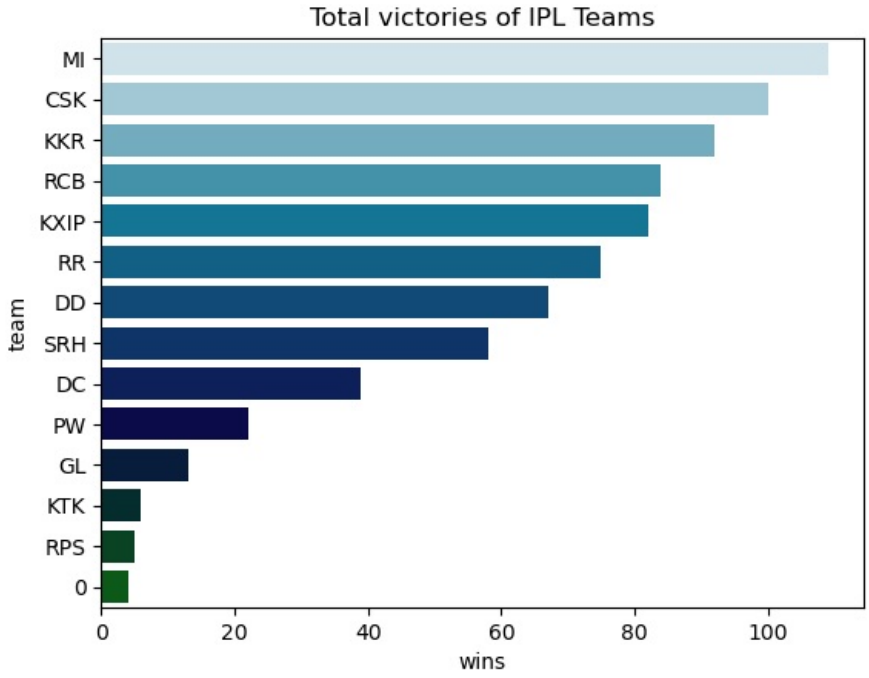
In [60]:
```
team_wins_df_matches
```

Out[60]:

	team	wins
0	MI	109
1	CSK	100
2	KKR	92
3	RCB	84
4	KXIP	82
5	RR	75
6	DD	67
7	SRH	58
8	DC	39
9	PW	22
10	GL	13
11	KTK	6
12	RPS	5
13	0	4

IPL Victories by team

In [61]:
```
plt.title('Total victories of IPL Teams')
sns.barplot(x='wins', y='team', data =team_wins_df_matches, palette ='ocean_r');
```



Most valuable players

In [70]:
```
mpv_ser=df_matches['player_of_match'].value_counts()

mvp_10_df_matches=pd.DataFrame(columns=['player','wins'])
count = 0
for items in mpv_ser.iteritems():
    if count>9:
        break
    else:
            temp_df2=pd.DataFrame({
                'player':[items[0]],
                'wins':[items[1]]
            })
            mvp_10_df_matches =mvp_10_df_matches.append(temp_df2,ignore_index=True)
            count += 1
```

Top 10 Most valuable players

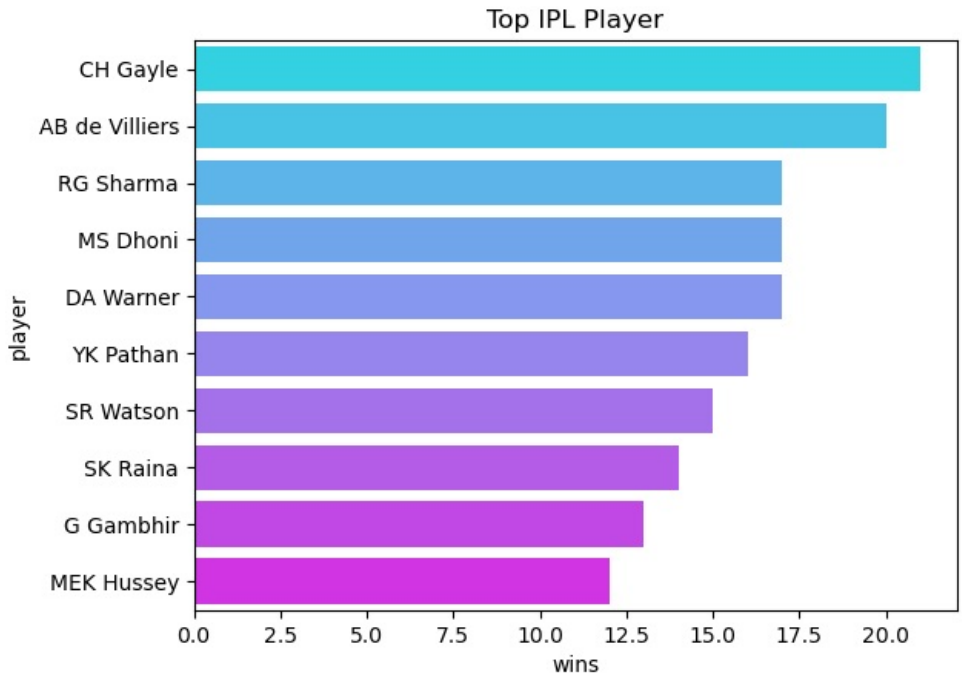
In [71]:
```
mvp_10_df_matches
```

Out[71]:

	player	wins
0	CH Gayle	21
1	AB de Villiers	20
2	RG Sharma	17
3	MS Dhoni	17
4	DA Warner	17
5	YK Pathan	16
6	SR Watson	15
7	SK Raina	14
8	G Gambhir	13
9	MEK Hussey	12

In [72]:
```
plt.title("Top IPL Player")
sns.barplot(x='wins', y='player', data =mvp_10_df_matches, palette ='cool')
```

Out[72]:
```
<AxesSubplot:title={'center':'Top IPL Player'}, xlabel='wins', ylabel='player'>
```



Team that won the most number of toss

In [73]:
```
toss_ser =df_matches['toss_winner'].value_counts()
toss_df_matches=pd.DataFrame(columns=['team','wins'])

for items in toss_ser.iteritems():
    temp_df3=pd.DataFrame({
        'team':[items[0]],
        'wins':[items[1]]
    })
    toss_df_matches = toss_df_matches.append(temp_df3,ignore_index=True)
```

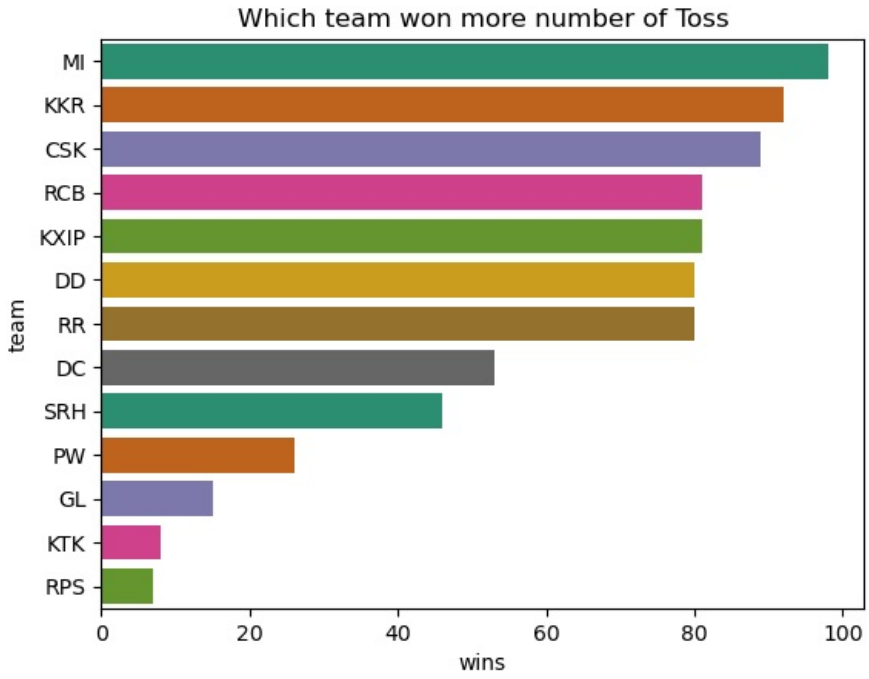
In [74]:
```
### Count of Number of toss wins and teams
toss_df_matches
```

Out[74]:

	team	wins
0	MI	98
1	KKR	92
2	CSK	89
3	RCB	81
4	KXIP	81
5	DD	80
6	RR	80
7	DC	53
8	SRH	46
9	PW	26
10	GL	15
11	KTK	8
12	RPS	7

In [76]:
```
### Teams which has won More number of Toss
plt.title('Which team won more number of Toss')
sns.barplot(x='wins', y='team', data=toss_df_matches, palette='Dark2')
```

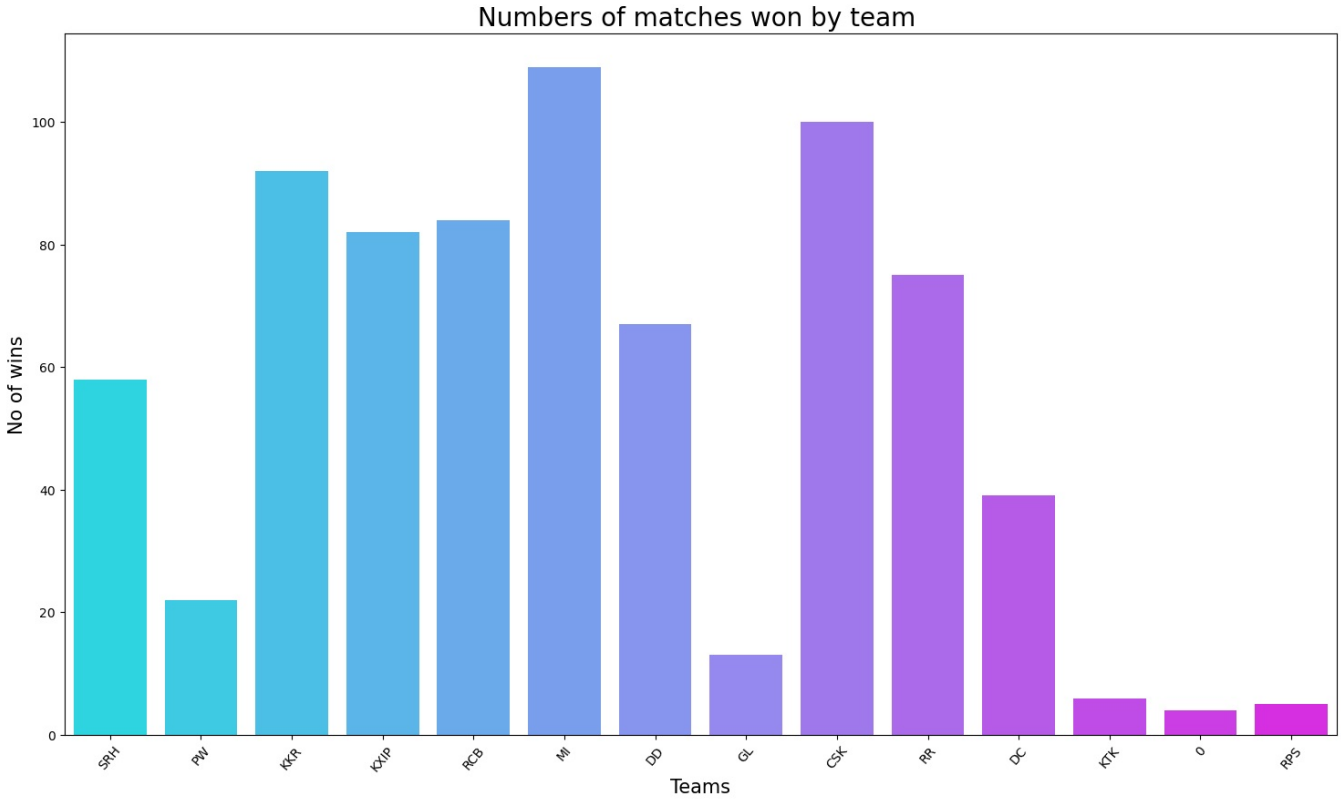
Out[76]:
```
<AxesSubplot:title={'center':'Which team won more number of Toss'}, xlabel='wins', ylabel='team'>
```



Observations

Mumbai Indians has won the most toss(till 2019) in ipl history.

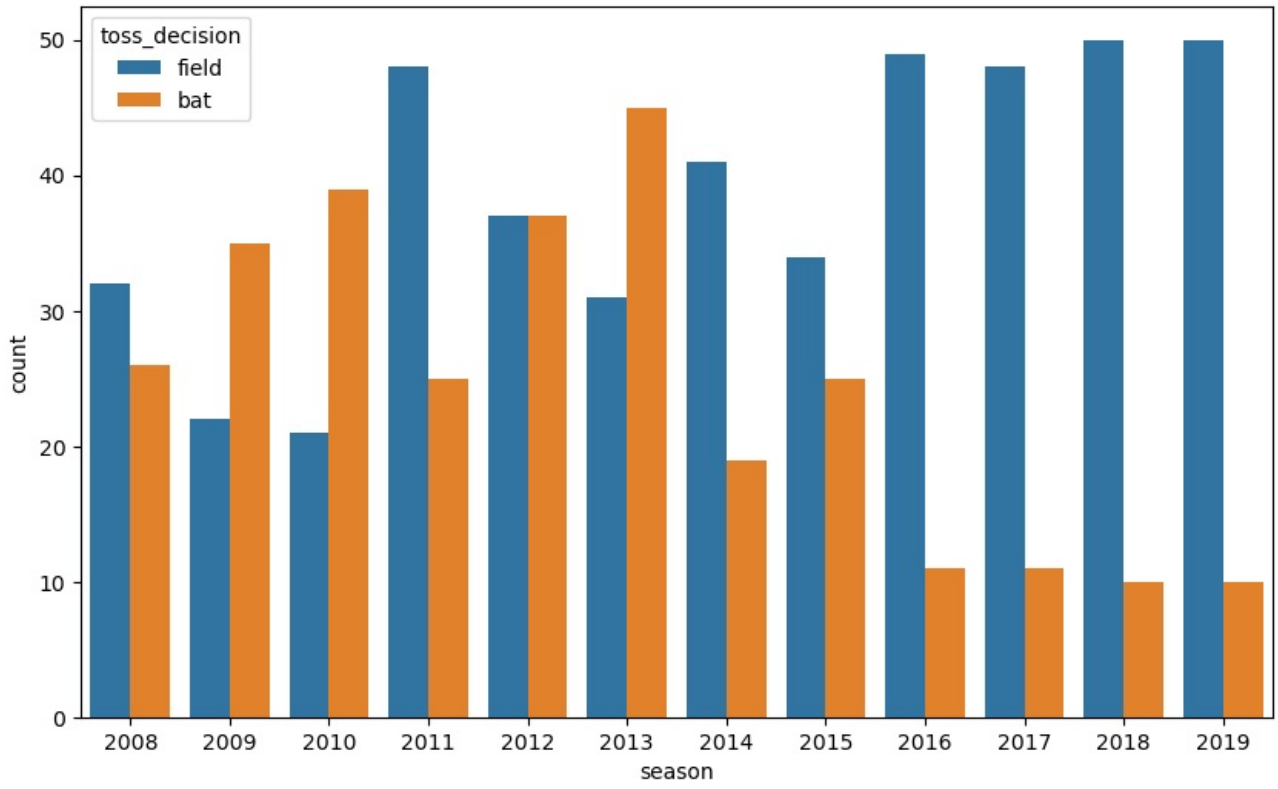
In [77]:
```
#Numbebr of Matches won by team
plt.figure(figsize = (18,10))
sns.countplot(x='winner',data=df_matches, palette='cool')
plt.title("Numbers of matches won by team ",fontsize=20)
plt.xticks(rotation=50)
plt.xlabel("Teams",fontsize=15)
plt.ylabel("No of wins",fontsize=15)
plt.show()
```



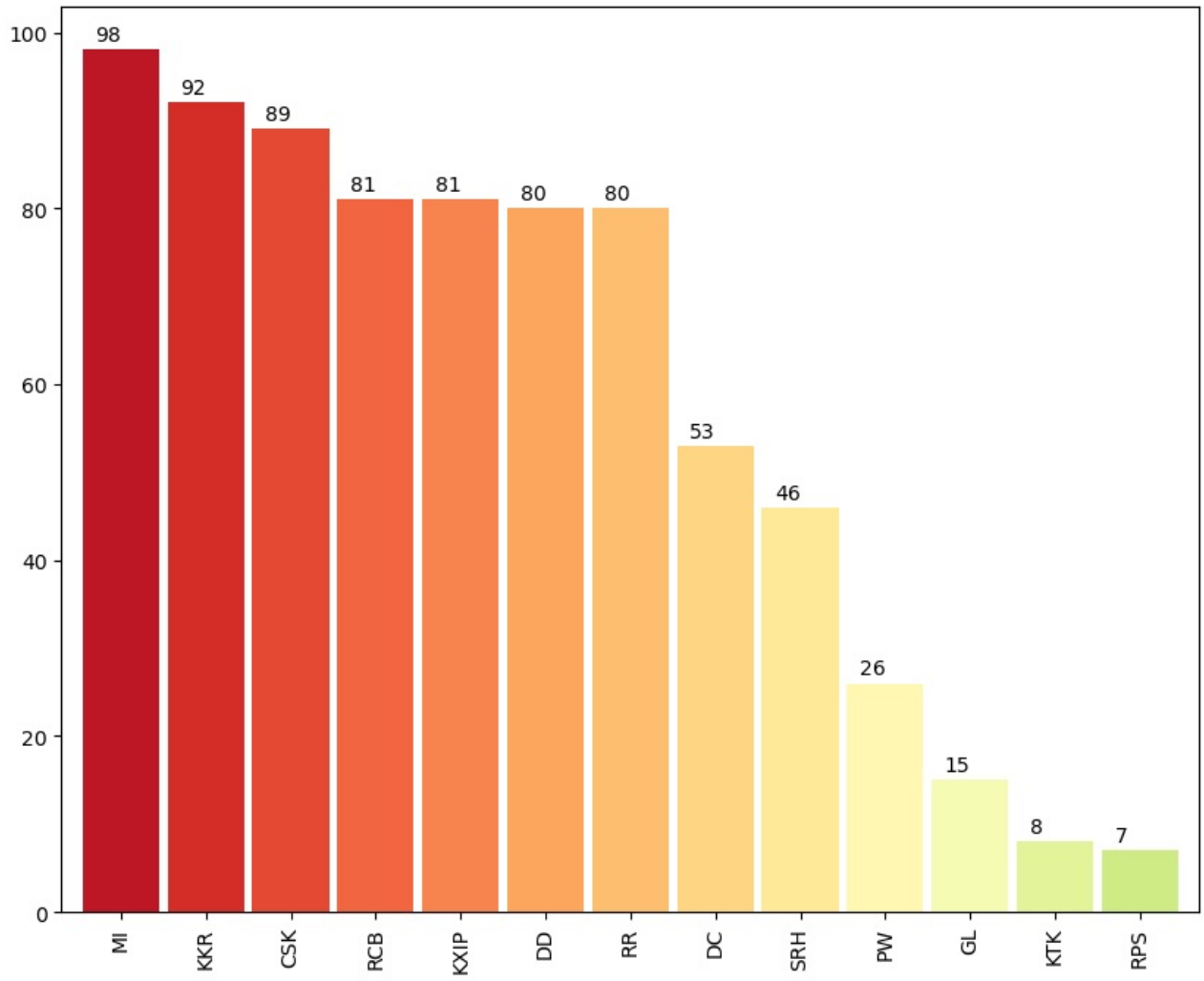
In [78]:
```
df_matches.result.value_counts()
```

Out[78]:
```
normal       743
tie            9
no result      4
Name: result, dtype: int64
```

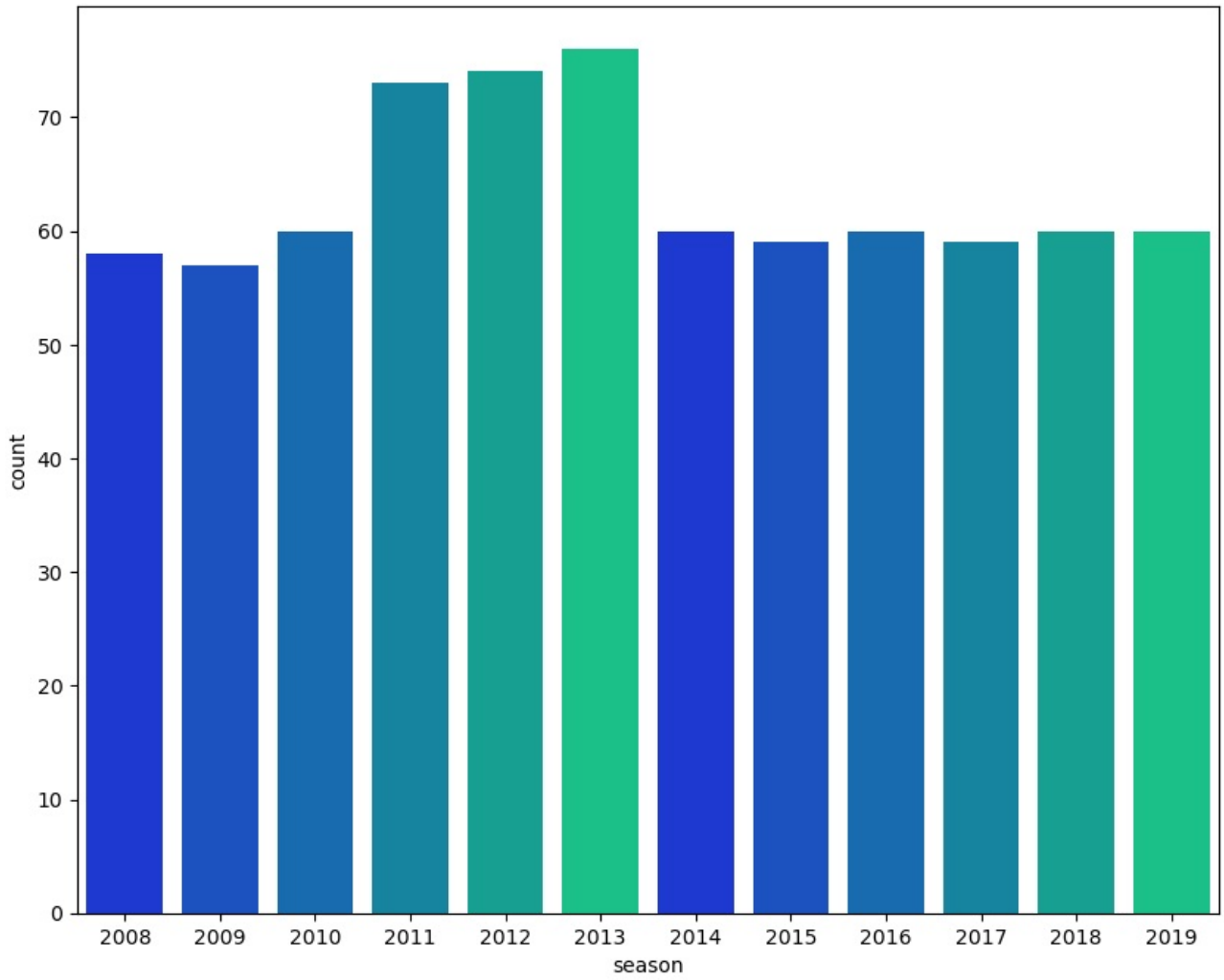
In [79]:
```
plt.subplots(figsize=(10,6))
sns.countplot(x='season', hue='toss_decision', data=df_matches)
plt.show()
```



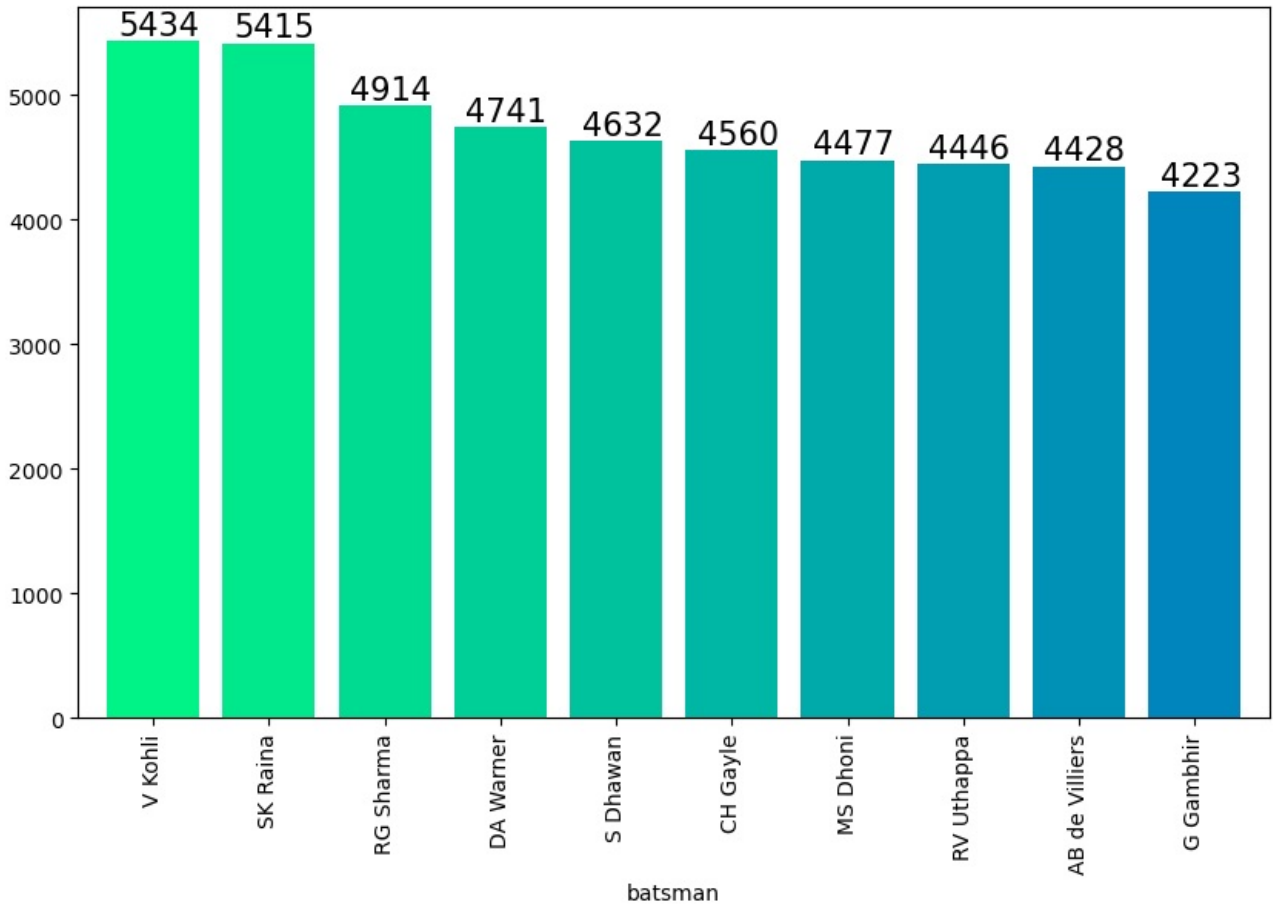
In [80]:
```
plt.subplots(figsize=(10,8))
ax=df_matches['toss_winner'].value_counts().plot.bar(width=0.9,color=sns.color_palette('RdYlGn', 20))
for p in ax.patches:
        ax.annotate(format(p.get_height()),(p.get_x()+0.15, p.get_height()+1))
plt.show()
```



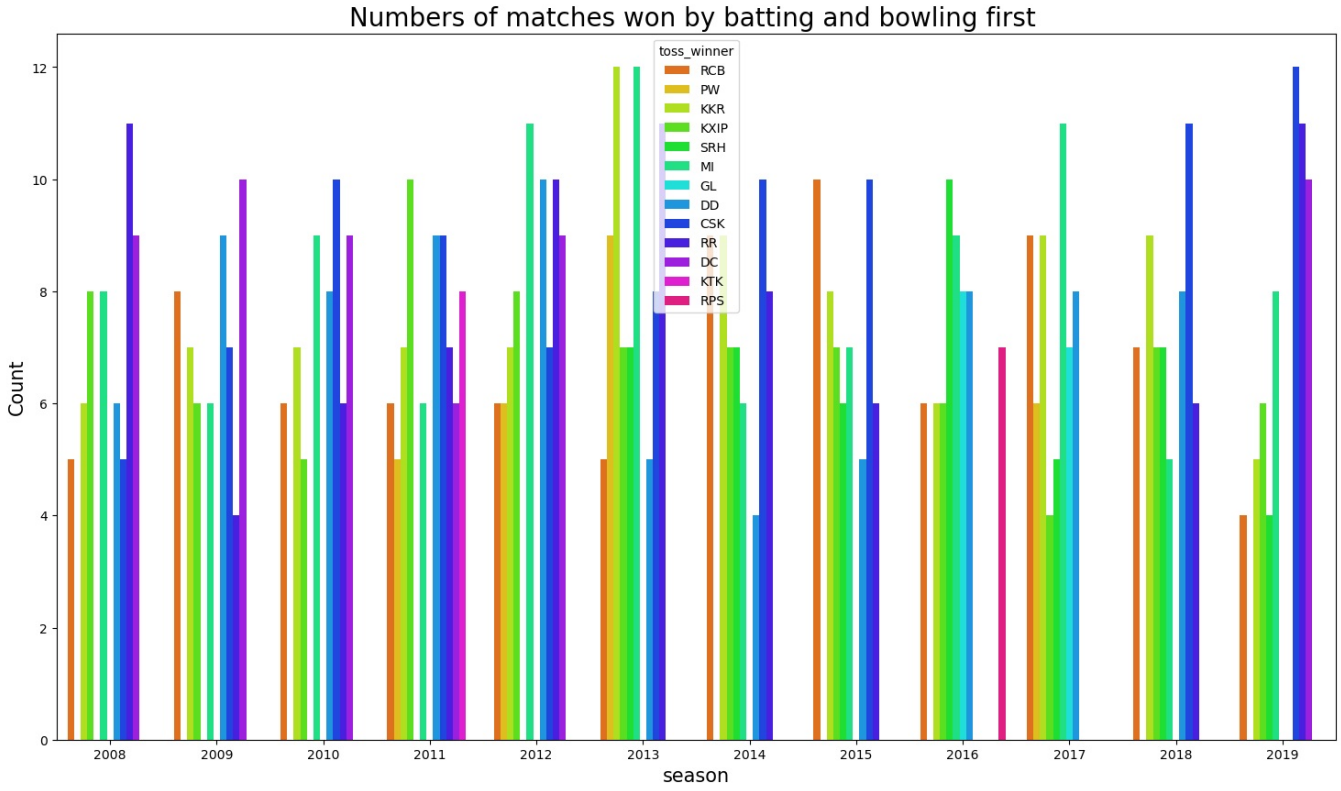
In [81]:
```
#Matches played across each seasons
plt.subplots(figsize=(10,8))
sns.countplot(x='season', data=df_matches,palette=sns.color_palette('winter'))
plt.show()
```



In [82]:
```
#Top 10 Batsman from the dataset
plt.subplots(figsize=(10,6))
max_runs=df_deliveries.groupby(['batsman'])['batsman_runs'].sum()
ax=max_runs.sort_values(ascending=False)[:10].plot.bar(width=0.8,color=sns.color_palette('winter_r',20))
for p in ax.patches:
    ax.annotate(format(p.get_height()),(p.get_x()+0.1, p.get_height()+50),fontsize=15)
plt.show()
```



In [83]:
```
#Number of matches won by Toss winning side
plt.figure(figsize = (18,10))
sns.countplot('season',hue='toss_winner',data=df_matches,palette='hsv')
plt.title("Numbers of matches won by batting and bowling first ",fontsize=20)
plt.xlabel("season",fontsize=15)
plt.ylabel("Count",fontsize=15)
plt.show()
```



In [84]:
```
df_matches
```

Out[84]:

	id	season	city	date	team1	team2	toss_winner	toss_decision	result	dl_applied	winner	win_by_runs	win_by_wi
0	1	2017	Hyderabad	2017-04-05	SRH	RCB	RCB	field	normal	0	SRH	35	
1	2	2017	Pune	2017-04-06	MI	PW	PW	field	normal	0	PW	0	
2	3	2017	Rajkot	2017-04-07	GL	KKR	KKR	field	normal	0	KKR	0	
3	4	2017	Indore	2017-04-08	PW	KXIP	KXIP	field	normal	0	KXIP	0	
4	5	2017	Bangalore	2017-04-08	RCB	DD	RCB	bat	normal	0	RCB	15	
...	...	...	...	...	...	...	...	...	...	...	...	...	
751	11347	2019	Mumbai	05/05/19	KKR	MI	MI	field	normal	0	MI	0	
752	11412	2019	Chennai	07/05/19	CSK	MI	CSK	bat	normal	0	MI	0	
753	11413	2019	Visakhapatnam	08/05/19	SRH	DC	DC	field	normal	0	DC	0	
754	11414	2019	Visakhapatnam	10/05/19	DC	CSK	CSK	field	normal	0	CSK	0	
755	11415	2019	Hyderabad	12/05/19	MI	CSK	MI	bat	normal	0	MI	1	

756 rows × 18 columns

In [85]:
```
# we will print winner season wise
final_matches=df_matches.drop_duplicates(subset=['season'], keep='last')

final_matches[['season','winner']].reset_index(drop=True).sort_values('season')
```

Out[85]:

	season	winner
1	2008	RR
2	2009	DC
3	2010	CSK
4	2011	CSK
5	2012	KKR
6	2013	MI
7	2014	KKR
8	2015	MI
9	2016	SRH
0	2017	MI
10	2018	CSK
11	2019	MI

In [86]:
```
# we will print number of season won by teams
final_matches["winner"].value_counts()
```

Out[86]:
```
MI     4
CSK    3
KKR    2
RR     1
DC     1
SRH    1
Name: winner, dtype: int64
```

In [87]:
```
# we will print toss winner, toss decision, winner in final matches.
final_matches[['toss_winner','toss_decision','winner']].reset_index(drop=True)
```

Out[87]:

	toss_winner	toss_decision	winner
0	MI	bat	MI
1	RR	field	RR
2	RCB	field	DC
3	CSK	bat	CSK
4	CSK	bat	CSK
5	CSK	bat	KKR
6	MI	bat	MI
7	KKR	field	KKR
8	CSK	field	MI
9	SRH	bat	SRH
10	CSK	field	CSK
11	MI	bat	MI

In [88]:
```
# we will print man of the match
final_matches[['winner','player_of_match']].reset_index(drop=True)
```

Out[88]:

	winner	player_of_match
0	MI	KH Pandya
1	RR	YK Pathan
2	DC	A Kumble
3	CSK	SK Raina
4	CSK	M Vijay
5	KKR	MS Bisla
6	MI	KA Pollard
7	KKR	MK Pandey
8	MI	RG Sharma
9	SRH	BCJ Cutting
10	CSK	SR Watson
11	MI	JJ Bumrah

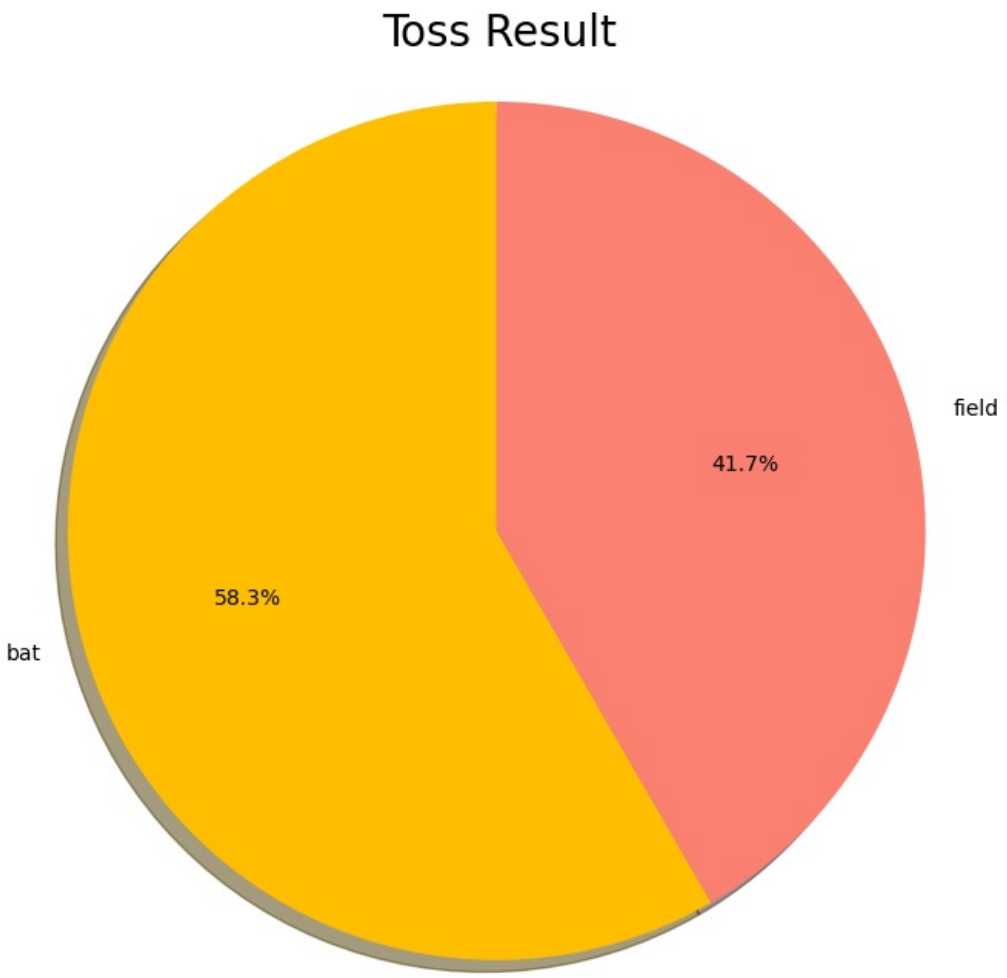
In [90]:
```
# we will print numbers of fours hit by team
season_data=df_matches[['id','season','winner']]
complete_data=df_deliveries.merge(season_data,how='inner',left_on='match_id',right_on='id')
four_data=complete_data[complete_data['batsman_runs']==4]
four_data.groupby('batting_team')['batsman_runs'].agg([('runs by fours','sum'),('fours','count')])
```

Out[90]:

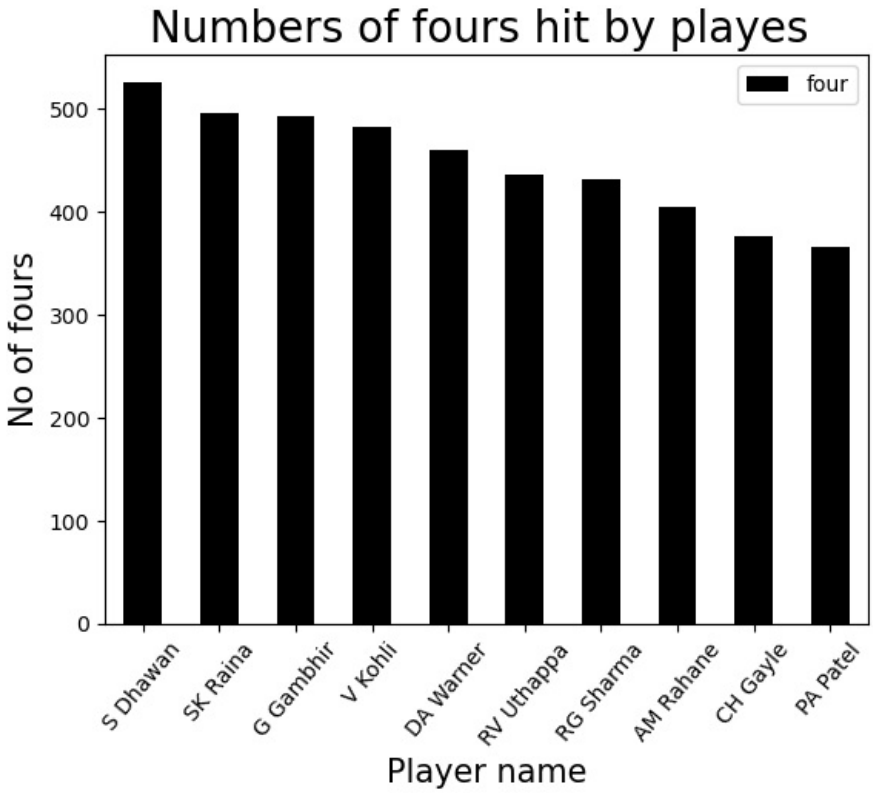
batting_team	runs by fours	fours
CSK	8772	2193
DC	4796	1199
DD	8632	2158
GL	1840	460
KKR	9736	2434
KTK	680	170
KXIP	9832	2458
MI	10352	2588
PW	2888	722
RCB	9440	2360
RPS	684	171
RR	8140	2035
SRH	5776	1444

In [91]:
```
Toss=final_matches.toss_decision.value_counts()
labels=np.array(Toss.index)
sizes = Toss.values
colors = ['#FFBF00', '#FA8072']
plt.figure(figsize = (10,8))
plt.pie(sizes, labels=labels, colors=colors,
        autopct='%1.1f%%', shadow=True,startangle=90)
plt.title('Toss Result', fontsize=20)
```

```
plt.axis('equal')
plt.show()
```

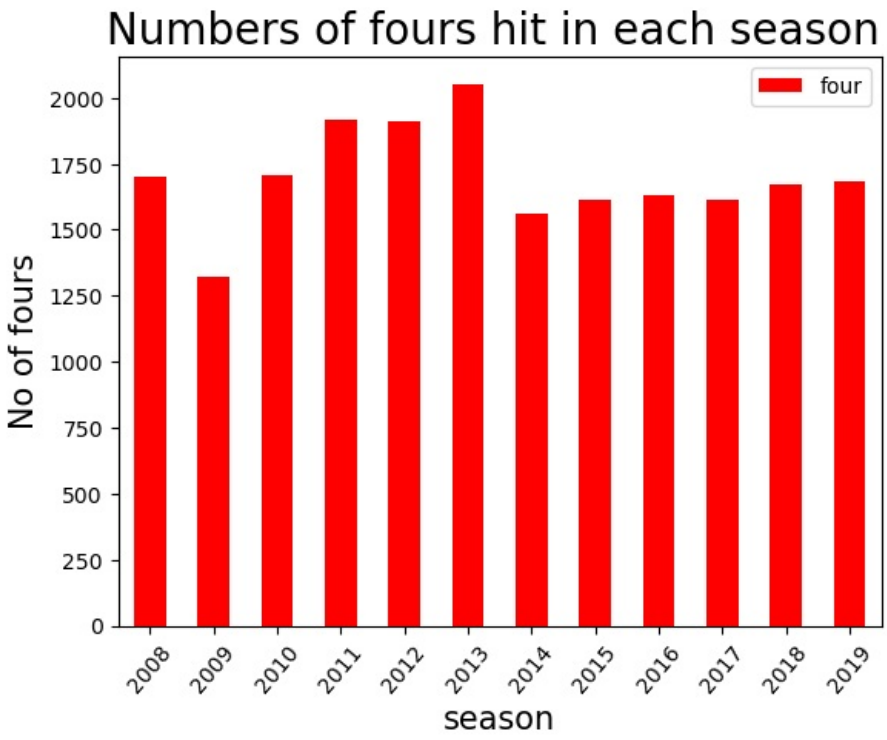


In [92]:
```
# we will plot graph on four hit by players
batsman_four=four_data.groupby('batsman')['batsman_runs'].agg([('four','count')]).reset_index().sort_values('fo
ax=batsman_four.iloc[:10,:].plot('batsman','four',kind='bar',color='black')
plt.title("Numbers of fours hit by playes ",fontsize=20)
plt.xticks(rotation=50)
plt.xlabel("Player name",fontsize=15)
plt.ylabel("No of fours",fontsize=15)
plt.show()
```



In [93]:
```
# we will plot graph on no of four hit in each season
```

```
# we will plot graph on no of four hit in each season
ax=four_data.groupby('season')['batsman_runs'].agg([('four','count')]).reset_index().plot('season','four',kind=
plt.title("Numbers of fours hit in each season ",fontsize=20)
plt.xticks(rotation=50)
plt.xlabel("season",fontsize=15)
plt.ylabel("No of fours",fontsize=15)
plt.show()
```



In [94]:
```
# we will print no of sixes hit by team
six_data=complete_data[complete_data['batsman_runs']==6]
six_data.groupby('batting_team')['batsman_runs'].agg([('runs by six','sum'),('sixes','count
```

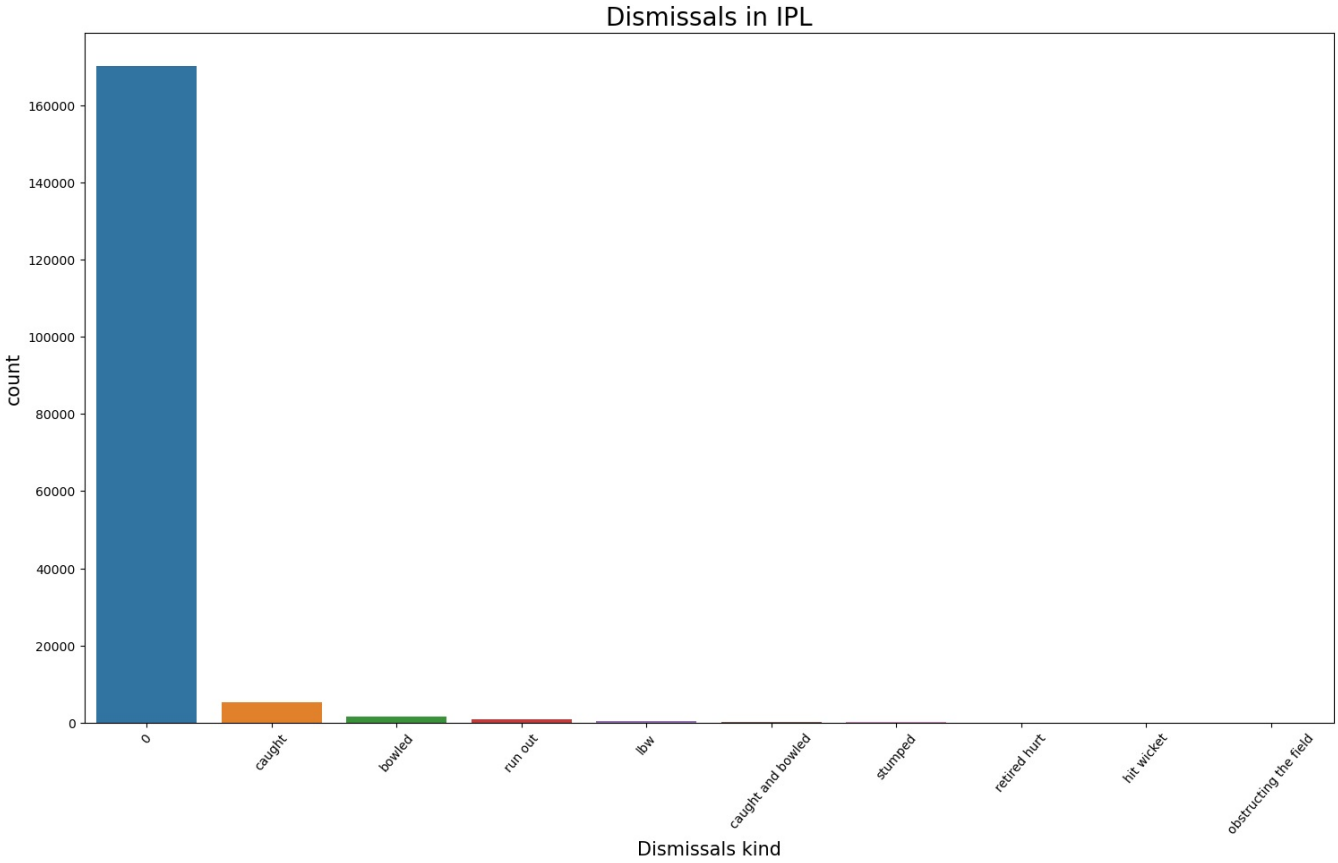
```
  File "C:\Users\ANIKET\AppData\Local\Temp\ipykernel_9588\2721299461.py", line 3
    six_data.groupby('batting_team')['batsman_runs'].agg([('runs by six','sum'),('sixes','count
                                                                                             ^
SyntaxError: EOL while scanning string literal
```

In [97]:
```
# we will print no of matches played by batsman
No_Matches_player= df_deliveries[["match_id","player_dismissed"]]
No_Matches_player =No_Matches_player .groupby("player_dismissed")["match_id"].count().reset_index().sort_values
No_Matches_player.columns=["batsman","No_of Matches"]
No_Matches_player .head(10)
```

Out[97]:

	batsman	No_of Matches
0	0	170244
1	SK Raina	162
2	RG Sharma	155
3	RV Uthappa	153
4	V Kohli	143
5	S Dhawan	137
6	G Gambhir	136
7	KD Karthik	135
8	PA Patel	126
9	AM Rahane	116

In [98]:
```
# Dismissals in IPL
plt.figure(figsize=(18,10))
ax=sns.countplot(df_deliveries.dismissal_kind)
plt.title("Dismissals in IPL",fontsize=20)
plt.xlabel("Dismissals kind",fontsize=15)
plt.ylabel("count",fontsize=15)
plt.xticks(rotation=50)
plt.show()
```



In [100…
```
wicket_data=df_deliveries.dropna(subset=['dismissal_kind'])
wicket_data=wicket_data[~wicket_data['dismissal_kind'].isin(['run out','retired hurt','obstructing the field'])
```

In [102…
```
# we will print ipl most wicket taking bowlers
wicket_data.groupby('bowler')['dismissal_kind'].agg(['count']).reset_index().sort_values('count',ascending=Fals
```

Out[102]:

	bowler	count
0	Harbhajan Singh	3440
1	A Mishra	3163
2	PP Chawla	3150
3	R Ashwin	3003
4	SL Malinga	2956
5	B Kumar	2699
6	DJ Bravo	2690
7	P Kumar	2625
8	UT Yadav	2588
9	SP Narine	2585

Conslusion :

The highest number of match played in IPL season was 2013,2014,2015.

The highest number of match won by Mumbai Indians i.e 4 match out of 12 matches.

Teams which Bowl first has higher chances of winning then the team which bat first.

After winning toss more teams decide to do fielding first.

In finals teams which decide to do fielding first win the matches more then the team which bat first.

In finals most teams after winning toss decide to do fielding first.

Top player of match winning are CH gayle, AB de villers.

It is interesting that out of 12 IPL finals,9 times the team that won the toss was also the winner of IPL.

The highest number of four hit by player is Shikar Dhawan.

The highest number of six hit by player is CH gayle.

Top leading run scorer in IPL are Virat kholi, SK Raina, RG Sharma.

Dismissals in IPL was most by Catch out.

The IPL most wicket taken blower is Harbajan Singh.

The highest number of matches played by player name are SK Raina, RG Sharma.

In [ ]: