# IPL_Data_Analysis

In [5]:

```
!pip install jovian --upgrade --quiet
```

In [6]:

```
project_name = 'ipl_data_analysis_and_visualization'
import jovian
jovian.commit(project = project_name, files = ['matches.csv'])
```

```
<IPython.core.display.Javascript object>

[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Uploading additional files...
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[6]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

# Section-1: Data Preparation and Cleaning ¶

In this data Analysis We will be using various Libraries such as pandas, Numpy, Seaborn & Matplotlib

In [7]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('dark_background')
```

In [8]:

```
ipl_df = pd.read_csv('matches.csv')
```

The Dataset I am using is downloaded from kaggle and contains around 6 csv's but For the current analysis we will be using only matches Played Data i.e matches.csv

In [9]:

```
ipl_df.head()
```

Out[9]:

| | id | Season | city | date | team1 | team2 | toss_winner | toss_decision | result | d |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | IPL-2017 | Hyderabad | 05-04-2017 | Sunrisers Hyderabad | Royal Challengers Bangalore | Royal Challengers Bangalore | field | normal | |
| **1** | 2 | IPL-2017 | Pune | 06-04-2017 | Mumbai Indians | Rising Pune Supergiant | Rising Pune Supergiant | field | normal | |
| **2** | 3 | IPL-2017 | Rajkot | 07-04-2017 | Gujarat Lions | Kolkata Knight Riders | Kolkata Knight Riders | field | normal | |
| **3** | 4 | IPL-2017 | Indore | 08-04-2017 | Rising Pune Supergiant | Kings XI Punjab | Kings XI Punjab | field | normal | |
| **4** | 5 | IPL-2017 | Bangalore | 08-04-2017 | Royal Challengers Bangalore | Delhi Daredevils | Royal Challengers Bangalore | bat | normal | |

In [10]:

```
ipl_df.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    object
 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(4), object(14)
memory usage: 106.4+ KB
```

to know number of rows and columns of dataset we will use the .shape method

In [11]:

```
ipl_df.shape
```

Out[11]:

```
(756, 18)
```

In [12]:

```
ipl_df.describe
```

Out[12]:

```
<bound method NDFrame.describe of         id    Season           city
date                        team1  \
0        1  IPL-2017      Hyderabad  05-04-2017          Sunrisers Hyderabad
1        2  IPL-2017           Pune  06-04-2017              Mumbai Indians
2        3  IPL-2017         Rajkot  07-04-2017               Gujarat Lions
3        4  IPL-2017         Indore  08-04-2017       Rising Pune Supergiant
4        5  IPL-2017      Bangalore  08-04-2017  Royal Challengers Bangalore
..     ...       ...            ...         ...                          ...
751  11347  IPL-2019         Mumbai  05-05-2019        Kolkata Knight Riders
752  11412  IPL-2019        Chennai  07-05-2019          Chennai Super Kings
753  11413  IPL-2019  Visakhapatnam  08-05-2019          Sunrisers Hyderabad
754  11414  IPL-2019  Visakhapatnam  10-05-2019              Delhi Capitals
755  11415  IPL-2019      Hyderabad  12-05-2019              Mumbai Indians

                         team2                 toss_winner toss_decision
\
0    Royal Challengers Bangalore  Royal Challengers Bangalore         field
1         Rising Pune Supergiant       Rising Pune Supergiant         field
2          Kolkata Knight Riders        Kolkata Knight Riders         field
3                Kings XI Punjab              Kings XI Punjab         field
4              Delhi Daredevils  Royal Challengers Bangalore           bat
..                          ...                         ...           ...
751               Mumbai Indians              Mumbai Indians         field
752               Mumbai Indians         Chennai Super Kings           bat
753               Delhi Capitals              Delhi Capitals         field
754          Chennai Super Kings         Chennai Super Kings         field
755          Chennai Super Kings              Mumbai Indians           bat

     result  dl_applied                       winner  win_by_runs  \
0    normal           0          Sunrisers Hyderabad           35
1    normal           0       Rising Pune Supergiant            0
2    normal           0        Kolkata Knight Riders            0
3    normal           0              Kings XI Punjab            0
4    normal           0  Royal Challengers Bangalore           15
..      ...         ...                          ...          ...
751  normal           0               Mumbai Indians            0
752  normal           0               Mumbai Indians            0
753  normal           0               Delhi Capitals            0
754  normal           0          Chennai Super Kings            0
755  normal           0               Mumbai Indians            1

     win_by_wickets player_of_match  \
0                 0    Yuvraj Singh
1                 7       SPD Smith
2                10         CA Lynn
3                 6      GJ Maxwell
4                 0       KM Jadhav
..              ...             ...
751               9       HH Pandya
752               6        AS Yadav
753               2         RR Pant
754               6    F du Plessis
755               0       JJ Bumrah

                                                venue        umpire1  \
```

```
0    Rajiv Gandhi International Stadium, Uppal     AY Dandekar
1      Maharashtra Cricket Association Stadium  A Nand Kishore
2       Saurashtra Cricket Association Stadium     Nitin Menon
3                         Holkar Cricket Stadium    AK Chaudhary
4                          M Chinnaswamy Stadium             NaN
..                                           ...             ...
751                           Wankhede Stadium   Nanda Kishore
752                  M. A. Chidambaram Stadium     Nigel Llong
753                           ACA-VDCA Stadium             NaN
754                           ACA-VDCA Stadium   Sundaram Ravi
755           Rajiv Gandhi Intl. Cricket Stadium     Nitin Menon

            umpire2                  umpire3
0          NJ Llong                      NaN
1            S Ravi                      NaN
2         CK Nandan                      NaN
3     C Shamshuddin                      NaN
4               NaN                      NaN
..              ...                      ...
751        O Nandan                   S Ravi
752     Nitin Menon                Ian Gould
753             NaN                      NaN
754  Bruce Oxenford  Chettithody Shamshuddin
755       Ian Gould              Nigel Llong

[756 rows x 18 columns]>
```

# Data Cleaning and Processing

In [13]:

```
ipl_df
```

Out[13]:

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

756 rows × 18 columns

We wont be using the Umpires Columns ('umpire1', 'umpire2', 'umpire3') in this analysis so we will remove those fields using .drop() method

In [14]:

```
#inplace argument is used to make permanent changes in the dataframe
ipl_df.drop(columns=['umpire1','umpire2','umpire3'],inplace=True)
```

In [15]:

```
# Exploring all column names in the data frame
ipl_df.columns
```

Out[15]:

```
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'],
      dtype='object')
```

In [16]:

```
# we use .unique() method to list the unique items from the selected column
ipl_df.Season.unique()
```

Out[16]:

```
array(['IPL-2017', 'IPL-2008', 'IPL-2009', 'IPL-2010', 'IPL-2011',
       'IPL-2012', 'IPL-2013', 'IPL-2014', 'IPL-2015', 'IPL-2016',
       'IPL-2018', 'IPL-2019'], dtype=object)
```

In [17]:

```
# Now Lets see all the teams that have played so far
ipl_df.team1.unique()
```

Out[17]:

```
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 [18]:

```
ipl_df.city.unique()
```

Out[18]:

```
array(['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'], dtype=object)
```

# From the Above Observations some Data cleaning is required

1.Pune was represented by various Team Names as 'Rising Pune Supergiant', 'Pune Warriors' & 'Rising Pune Supergiants' so as a convinience we will changes these with the recent team representing Pune 'Rising Pune Supergiant' in all columns involving this name i.e 'team1','team2','winner' & 'toss_winner' columns, similarly 2nd

Change is in team name of Delhi.

2.Earlier the team name for delhi was 'Delhi Daredevils' but later it was changed to 'Delhi Capitals' so we will replace the "delhi Daredevils' with 'Delhi Capitals'.

3.Bangalore was Renamed as Bengaluru in 2014 so we will change the Name for City Bangalore to Bengaluru to avoid Errors in Data Analysis.

In [19]:

```
# We will use the .replace() method for the above mentioned cleaning
ipl_df.team1.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant', 'Delhi Daredevi
ipl_df.team2.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant', 'Delhi Daredevi
ipl_df.toss_winner.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant', 'Delhi Da
ipl_df.winner.replace({'Rising Pune Supergiants' : 'Rising Pune Supergiant', 'Delhi Daredev
ipl_df.city.replace({'Bangalore' : 'Bengaluru'},inplace=True)
```

In [20]:

```
ipl_df.team1.unique()
```

Out[20]:

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

In [21]:

```
ipl_df.team2.unique()
```

Out[21]:

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

In [22]:

```
ipl_df.city.unique()
```

Out[22]:

```
array(['Hyderabad', 'Pune', 'Rajkot', 'Indore', 'Bengaluru', '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'], dtype=object)
```



So We Have Cleaned With Replication And Misspelled Data

In [23]:

```
jovian.commit(project=project_name, files = ['matches.csv'])
```

```
<IPython.core.display.Javascript object>

[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Uploading additional files...
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[23]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

# Lets Check For Missing Values

In [24]:

```
# we can use .isnull() to set Null values to True and then use .sum() to calculate all the
ipl_df.isnull().sum().sum()
```

Out[24]:

```
15
```

Above Result Shows we have 15 Null values in our data set. Now we will search For them.

In [25]:

```
null_df = ipl_df[ipl_df.isna().any(axis=1)]
```

In [26]:

```
null_df
```

Out[26]:

| | id | Season | city | date | team1 | team2 | toss_winner | toss_decision | res |
|---|---|---|---|---|---|---|---|---|---|
| **300** | 301 | IPL-2011 | Delhi | 21-05-2011 | Delhi Capitals | Rising Pune Supergiant | Delhi Capitals | bat | res |
| **461** | 462 | IPL-2014 | NaN | 19-04-2014 | Mumbai Indians | Royal Challengers Bangalore | Royal Challengers Bangalore | field | norn |
| **462** | 463 | IPL-2014 | NaN | 19-04-2014 | Kolkata Knight Riders | Delhi Capitals | Kolkata Knight Riders | bat | norn |
| **466** | 467 | IPL-2014 | NaN | 23-04-2014 | Chennai Super Kings | Rajasthan Royals | Rajasthan Royals | field | norn |
| **468** | 469 | IPL-2014 | NaN | 25-04-2014 | Sunrisers Hyderabad | Delhi Capitals | Sunrisers Hyderabad | bat | norn |
| **469** | 470 | IPL-2014 | NaN | 25-04-2014 | Mumbai Indians | Chennai Super Kings | Mumbai Indians | bat | norn |
| **474** | 475 | IPL-2014 | NaN | 28-04-2014 | Royal Challengers Bangalore | Kings XI Punjab | Kings XI Punjab | field | norn |
| **476** | 477 | IPL-2014 | NaN | 30-04-2014 | Sunrisers Hyderabad | Mumbai Indians | Mumbai Indians | field | norn |
| **545** | 546 | IPL-2015 | Bengaluru | 29-04-2015 | Royal Challengers Bangalore | Rajasthan Royals | Rajasthan Royals | field | res |
| **570** | 571 | IPL-2015 | Bengaluru | 17-05-2015 | Delhi Capitals | Royal Challengers Bangalore | Royal Challengers Bangalore | field | res |
| **744** | 11340 | IPL-2019 | Bengaluru | 30-04-2019 | Royal Challengers Bangalore | Rajasthan Royals | Rajasthan Royals | field | res |

From Above Observations We can See NaN values in various Columns like 'city', 'winner' ,'Player Of The Match'. But By Observation it is clear the NaN Values for columns like 'Winner' and 'Player Of The Match' are only for Case When Match had "No Result" so we Can assume the Match might have been a Draw or Cancelled Due to Some Weather Or Technical Conditions. While The Other Case 'City' Column has NaN values for Rows where Stadium Location is dubai. So we will Replace These NaN values and Insert "Dubai" as City in its Place.

We Can See this values are at index 461,462,466,468,469,474,476

In [27]:

```
ipl_df.loc[460:470]
```

Out[27]:

| | id | Season | city | date | team1 | team2 | toss_winner | toss_decision | result |
|---|---|---|---|---|---|---|---|---|---|
| **460** | 461 | IPL-2014 | Abu Dhabi | 18-04-2014 | Sunrisers Hyderabad | Rajasthan Royals | Rajasthan Royals | field | normal |
| **461** | 462 | IPL-2014 | NaN | 19-04-2014 | Mumbai Indians | Royal Challengers Bangalore | Royal Challengers Bangalore | field | normal |
| **462** | 463 | IPL-2014 | NaN | 19-04-2014 | Kolkata Knight Riders | Delhi Capitals | Kolkata Knight Riders | bat | normal |
| **463** | 464 | IPL-2014 | Sharjah | 20-04-2014 | Rajasthan Royals | Kings XI Punjab | Kings XI Punjab | field | normal |
| **464** | 465 | IPL-2014 | Abu Dhabi | 21-04-2014 | Chennai Super Kings | Delhi Capitals | Chennai Super Kings | bat | normal |
| **465** | 466 | IPL-2014 | Sharjah | 22-04-2014 | Kings XI Punjab | Sunrisers Hyderabad | Sunrisers Hyderabad | field | normal |
| **466** | 467 | IPL-2014 | NaN | 23-04-2014 | Chennai Super Kings | Rajasthan Royals | Rajasthan Royals | field | normal |
| **467** | 468 | IPL-2014 | Sharjah | 24-04-2014 | Kolkata Knight Riders | Royal Challengers Bangalore | Royal Challengers Bangalore | field | normal |
| **468** | 469 | IPL-2014 | NaN | 25-04-2014 | Sunrisers Hyderabad | Delhi Capitals | Sunrisers Hyderabad | bat | normal |
| **469** | 470 | IPL-2014 | NaN | 25-04-2014 | Mumbai Indians | Chennai Super Kings | Mumbai Indians | bat | normal |
| **470** | 471 | IPL-2014 | Abu Dhabi | 26-04-2014 | Royal Challengers Bangalore | Rajasthan Royals | Rajasthan Royals | field | normal |

Now We will Replace it With "Dubai"

In [28]:

```
ipl_df.loc[[461,462,466,468,469,474,476],'city'] = "Dubai"
```

In [29]:

```
#Lets See the Changed Values
ipl_df.loc[461:480]
```

Out[29]:

| | id | Season | city | date | team1 | team2 | toss_winner | toss_decision | result |
|---|---|---|---|---|---|---|---|---|---|
| **461** | 462 | IPL-2014 | Dubai | 19-04-2014 | Mumbai Indians | Royal Challengers Bangalore | Royal Challengers Bangalore | field | norma |
| **462** | 463 | IPL-2014 | Dubai | 19-04-2014 | Kolkata Knight Riders | Delhi Capitals | Kolkata Knight Riders | bat | norma |
| **463** | 464 | IPL-2014 | Sharjah | 20-04-2014 | Rajasthan Royals | Kings XI Punjab | Kings XI Punjab | field | norma |
| **464** | 465 | IPL-2014 | Abu Dhabi | 21-04-2014 | Chennai Super Kings | Delhi Capitals | Chennai Super Kings | bat | norma |
| **465** | 466 | IPL-2014 | Sharjah | 22-04-2014 | Kings XI Punjab | Sunrisers Hyderabad | Sunrisers Hyderabad | field | norma |
| **466** | 467 | IPL-2014 | Dubai | 23-04-2014 | Chennai Super Kings | Rajasthan Royals | Rajasthan Royals | field | norma |
| **467** | 468 | IPL-2014 | Sharjah | 24-04-2014 | Kolkata Knight Riders | Royal Challengers Bangalore | Royal Challengers Bangalore | field | norma |
| **468** | 469 | IPL-2014 | Dubai | 25-04-2014 | Sunrisers Hyderabad | Delhi Capitals | Sunrisers Hyderabad | bat | norma |
| **469** | 470 | IPL-2014 | Dubai | 25-04-2014 | Mumbai Indians | Chennai Super Kings | Mumbai Indians | bat | norma |
| **470** | 471 | IPL-2014 | Abu Dhabi | 26-04-2014 | Royal Challengers Bangalore | Rajasthan Royals | Rajasthan Royals | field | norma |
| **471** | 472 | IPL-2014 | Abu Dhabi | 26-04-2014 | Kings XI Punjab | Kolkata Knight Riders | Kolkata Knight Riders | field | norma |
| **472** | 473 | IPL-2014 | Sharjah | 27-04-2014 | Mumbai Indians | Delhi Capitals | Mumbai Indians | bat | norma |
| **473** | 474 | IPL-2014 | Sharjah | 27-04-2014 | Sunrisers Hyderabad | Chennai Super Kings | Sunrisers Hyderabad | bat | norma |
| **474** | 475 | IPL-2014 | Dubai | 28-04-2014 | Royal Challengers Bangalore | Kings XI Punjab | Kings XI Punjab | field | norma |

| | id | Season | city | date | team1 | team2 | toss_winner | toss_decision | result |
|---|---|---|---|---|---|---|---|---|---|
| **475** | 476 | IPL-2014 | Abu Dhabi | 29-04-2014 | Rajasthan Royals | Kolkata Knight Riders | Rajasthan Royals | bat | tie |
| **476** | 477 | IPL-2014 | Dubai | 30-04-2014 | Sunrisers Hyderabad | Mumbai Indians | Mumbai Indians | field | norma |
| **477** | 478 | IPL-2014 | Ranchi | 02-05-2014 | Chennai Super Kings | Kolkata Knight Riders | Chennai Super Kings | bat | norma |
| **478** | 479 | IPL-2014 | Mumbai | 03-05-2014 | Kings XI Punjab | Mumbai Indians | Kings XI Punjab | bat | norma |
| **479** | 480 | IPL-2014 | Delhi | 03-05-2014 | Delhi Capitals | Rajasthan Royals | Rajasthan Royals | field | norma |
| **480** | 481 | IPL-2014 | Bengaluru | 04-05-2014 | Sunrisers Hyderabad | Royal Challengers Bangalore | Royal Challengers Bangalore | field | norma |

In [30]:

```
#Now Lets Confirm if we have any NaN values in City Field
ipl_df.city.isnull().any()
```

Out[30]:

```
False
```

Now Lets Check For Total Remaining NaN Values

In [31]:

```
# Lets Check if any any other COlumns Have NaN values
ipl_df.isna().any()[lambda x: x]
```

Out[31]:

```
winner             True
player_of_match    True
dtype: bool
```

From Above Results It is clear That we have have No NaN values Other than Those in Columns Of Winner and Player Of The Match

So We Have Now Completed With Our Data Cleaning Part and Can Move with Further Steps

## Let's Commit the Work Completed Until now

In [33]:

```
jovian.commit(project=project_name)
```

```
<IPython.core.display.Javascript object>
```

```
[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[33]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

# Section-2: Exploratory Analysis and Visualization

Now We Will Analyse the Data For Different types of Queries

In [34]:

```
#Lets find total Number of Matches Played from 2008 - 2019
ipl_df.id.count()
```

Out[34]:

```
756
```

We can see 756 Matches Have Been Played in 11 Seasons (08 - 19)

In [35]:

```
#Now Lets Find Total Number Of Matches Where Result Was normal i.e Not A Tie
regular_matches = ipl_df[ipl_df.result == 'normal'].count()
```

In [36]:

```
regular_matches.result
```

Out[36]:

```
743
```

We can See From 756 Matches Played Only 13 seem not to have a normal result

## Total Matches Played in Each City

In [37]:

```
#Lets See About Cities Where Matches have Been Played
ipl_df.city.unique()
```

Out[37]:

```
array(['Hyderabad', 'Pune', 'Rajkot', 'Indore', 'Bengaluru', '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',
       'Dubai', 'Mohali'], dtype=object)
```

Now Lets See Match count played in each of the above city

In [38]:

```
cities = ipl_df.groupby('city')[['id']].count()
```

In [39]:

```
cities
```

Out[39]:

| city | id |
|---|---|
| Abu Dhabi | 7 |
| Ahmedabad | 12 |
| Bengaluru | 80 |
| Bloemfontein | 2 |
| Cape Town | 7 |
| Centurion | 12 |
| Chandigarh | 46 |
| Chennai | 57 |
| Cuttack | 7 |
| Delhi | 74 |
| Dharamsala | 9 |
| Dubai | 7 |
| Durban | 15 |
| East London | 3 |
| Hyderabad | 64 |
| Indore | 9 |
| Jaipur | 47 |
| Johannesburg | 8 |
| Kanpur | 4 |
| Kimberley | 3 |
| Kochi | 5 |
| Kolkata | 77 |
| Mohali | 10 |
| Mumbai | 101 |
| Nagpur | 3 |
| Port Elizabeth | 7 |
| Pune | 38 |
| Raipur | 6 |
| Rajkot | 10 |
| Ranchi | 7 |
| Sharjah | 6 |
| Visakhapatnam | 13 |

Lets Arrange this data In a More Organised manner

In [40]:

```
plt.figaspect
cities.rename(columns={'id':'matches'},inplace=True)
cities = cities.sort_values('matches',ascending=True).reset_index()
cities
```

Out[40]:

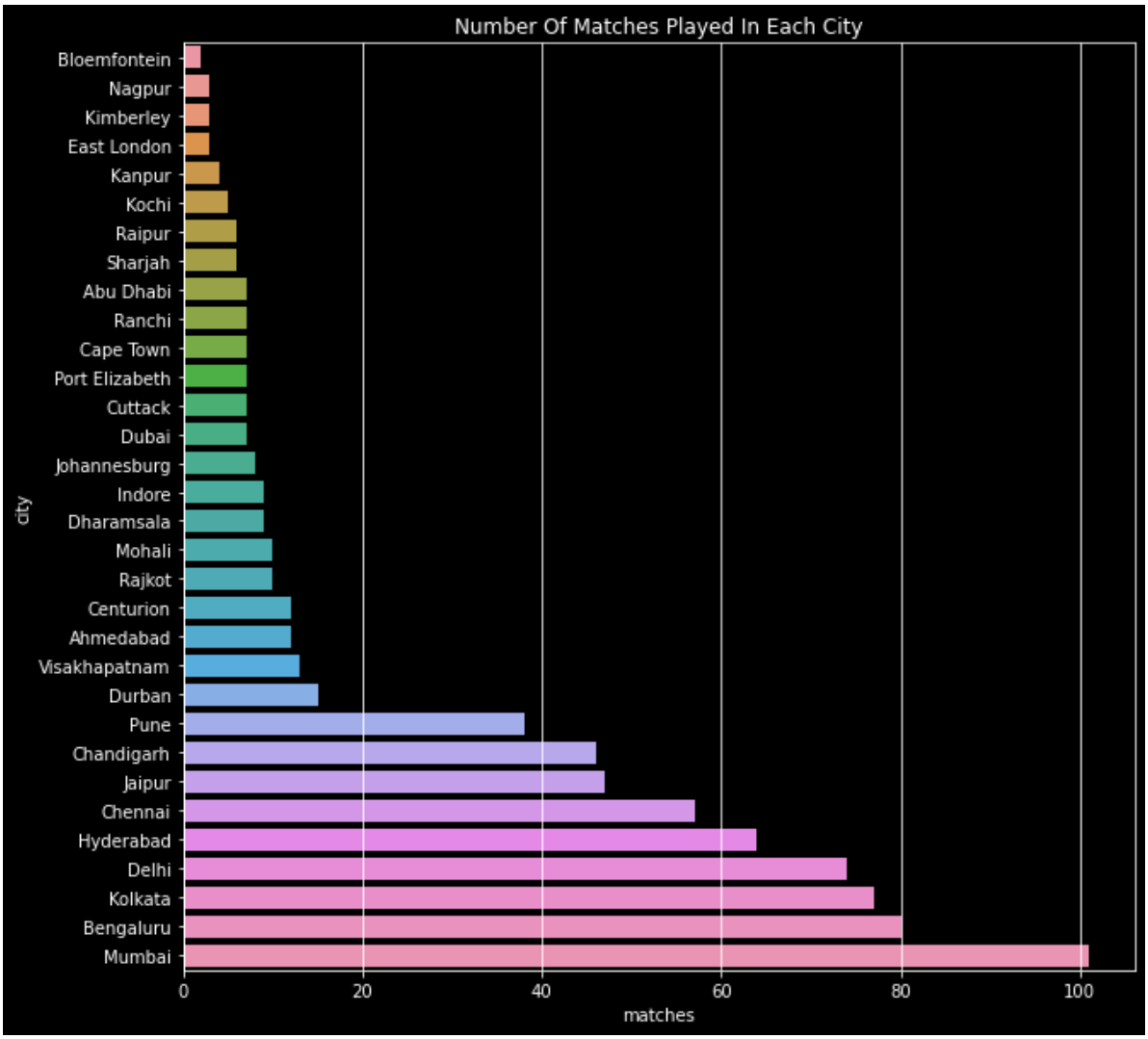
| | city | matches |
|---|---|---|
| 0 | Bloemfontein | 2 |
| 1 | Nagpur | 3 |
| 2 | Kimberley | 3 |
| 3 | East London | 3 |
| 4 | Kanpur | 4 |
| 5 | Kochi | 5 |
| 6 | Raipur | 6 |
| 7 | Sharjah | 6 |
| 8 | Abu Dhabi | 7 |
| 9 | Ranchi | 7 |
| 10 | Cape Town | 7 |
| 11 | Port Elizabeth | 7 |
| 12 | Cuttack | 7 |
| 13 | Dubai | 7 |
| 14 | Johannesburg | 8 |
| 15 | Indore | 9 |
| 16 | Dharamsala | 9 |
| 17 | Mohali | 10 |
| 18 | Rajkot | 10 |
| 19 | Centurion | 12 |
| 20 | Ahmedabad | 12 |
| 21 | Visakhapatnam | 13 |
| 22 | Durban | 15 |
| 23 | Pune | 38 |
| 24 | Chandigarh | 46 |
| 25 | Jaipur | 47 |
| 26 | Chennai | 57 |
| 27 | Hyderabad | 64 |
| 28 | Delhi | 74 |
| 29 | Kolkata | 77 |
| 30 | Bengaluru | 80 |
| 31 | Mumbai | 101 |

We can See IPL has Altogether 32 Official Locations where matches have been Played since 2008 till 2019. As we all might know this year's IPL is again being Held at UAE.

Lets Plot the Cities in a bar Chart

In [48]:

```
plt.figure(figsize=(10,10))
plt.grid()
plt.title('Number Of Matches Played In Each City')
sns.barplot(x='matches',y='city',data=cities);
```




It seems Mumbai has Been the favourite Location followed by Bengaluru and Kolkata

Now Lets See Matches Won by Each Team

## Total Matches Won By Each Team

In [42]:

```
ipl_df.winner.unique()
```

Out[42]:

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

In [43]:

```
winner_df = ipl_df.groupby('winner')[['id']].count()
winner_df = winner_df.sort_values('id', ascending=False).reset_index()

winner_df.rename(columns = {'id':'wins','winner':'Teams'},inplace=True)
winner_df
```

Out[43]:

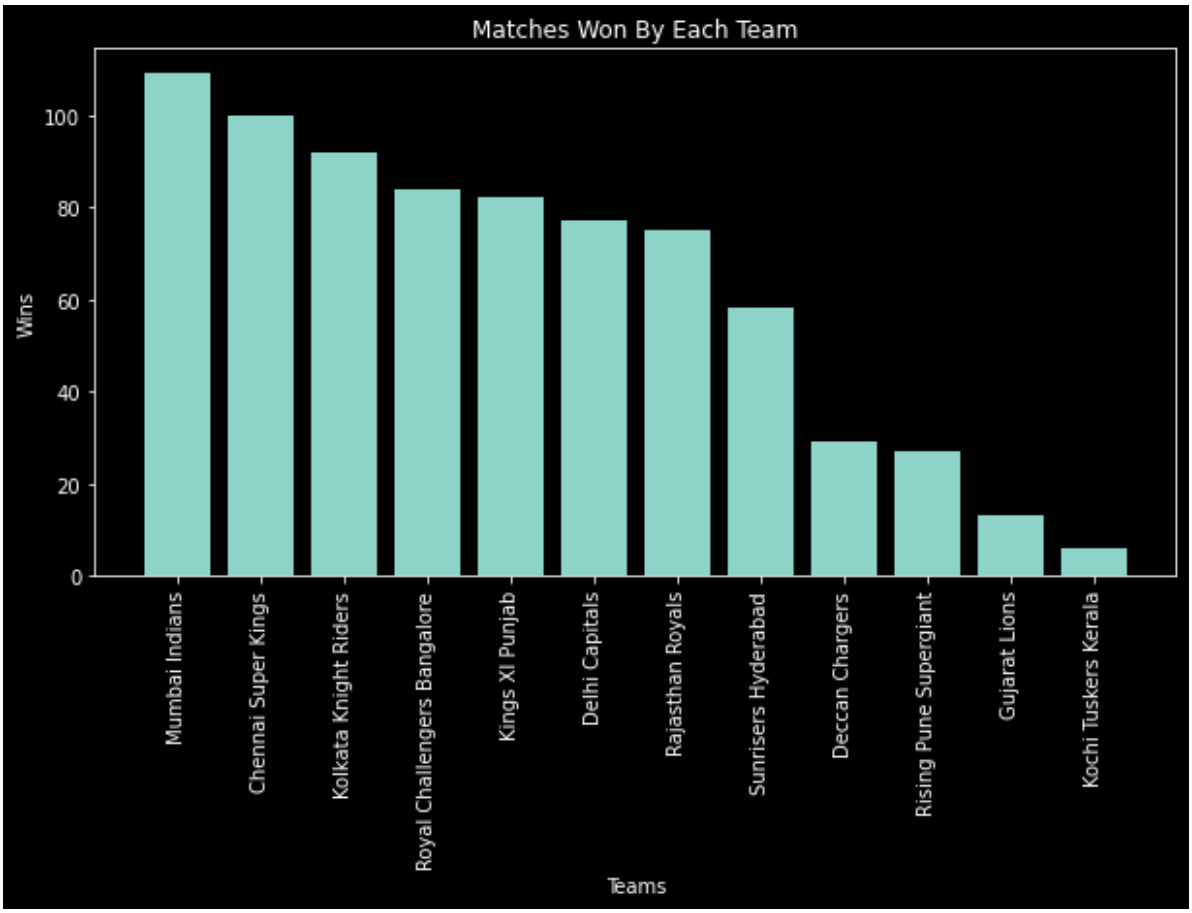
| | Teams | wins |
|---|---|---|
| **0** | Mumbai Indians | 109 |
| **1** | Chennai Super Kings | 100 |
| **2** | Kolkata Knight Riders | 92 |
| **3** | Royal Challengers Bangalore | 84 |
| **4** | Kings XI Punjab | 82 |
| **5** | Delhi Capitals | 77 |
| **6** | Rajasthan Royals | 75 |
| **7** | Sunrisers Hyderabad | 58 |
| **8** | Deccan Chargers | 29 |
| **9** | Rising Pune Supergiant | 27 |
| **10** | Gujarat Lions | 13 |
| **11** | Kochi Tuskers Kerala | 6 |

Seems Mumbai Indians Have won the Most matches in IPL Till Date. Followed by Chennai Super Kings.

Now Lets Plot These Wins

In [44]:

```
#Plotting Wins vs Teams
plt.figure(figsize=(10,5))
plt.xlabel('Teams')
plt.ylabel('Wins')
plt.xticks(rotation=90)
plt.title('Matches Won By Each Team');
plt.bar(winner_df.Teams,winner_df.wins);
```

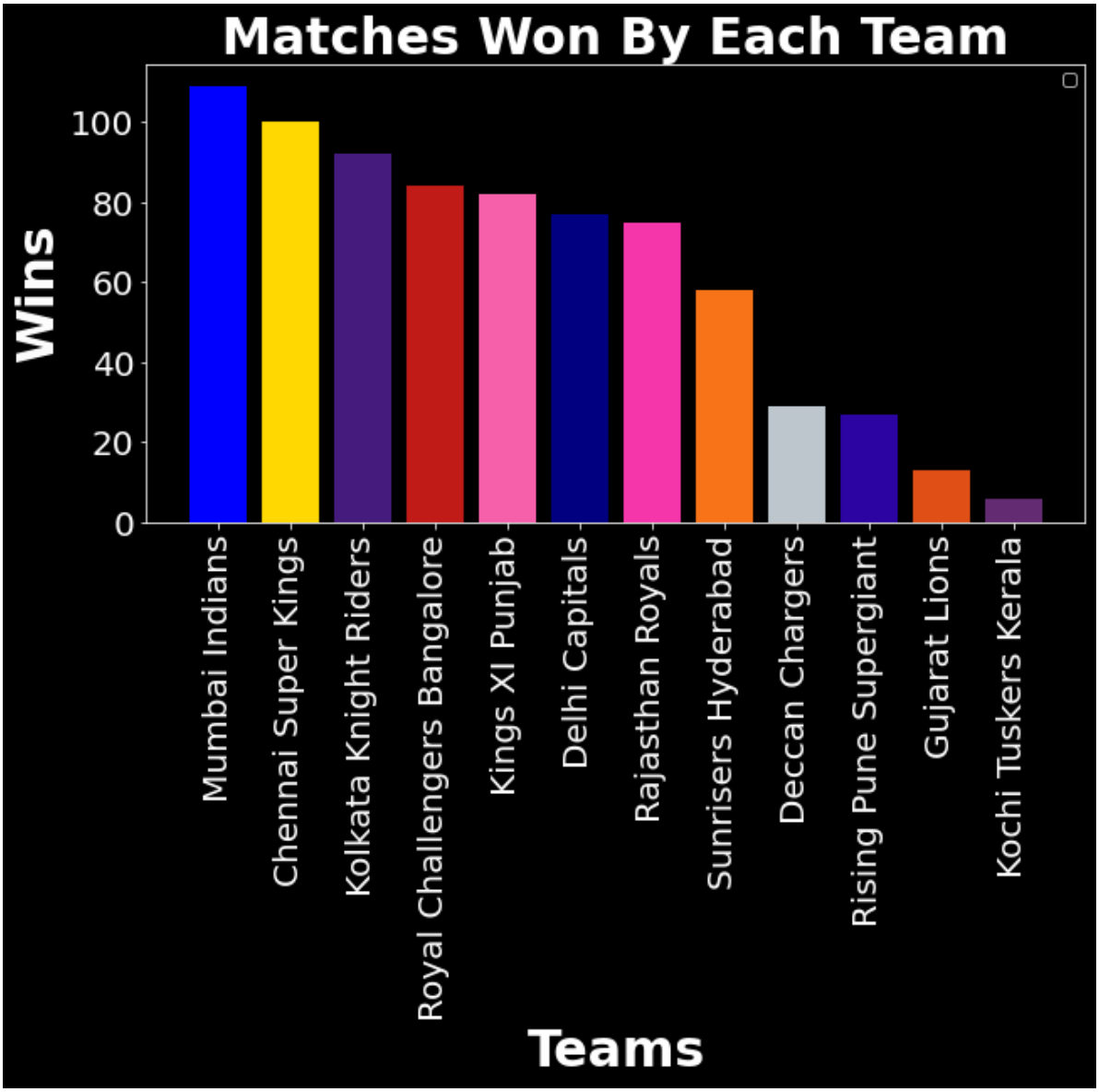



Lets Add Colour To Each Team so That we Get A Clear Idea

We can do this by using color argument of the bar() Function

In [46]:

```
#Plotting Wins vs Teams
#We will be using colour code of teams jersey to make it easily understandable
plt.figure(figsize=(10,5))
plt.legend(winner_df.Teams,loc=1)
plt.xlabel('Teams',fontweight='bold',fontsize=30)
plt.ylabel('Wins',fontweight='bold',fontsize=30)
plt.tick_params(labelsize=20)
plt.xticks(rotation=90)
plt.title('Matches Won By Each Team',fontweight='bold',fontsize=30);
plt.bar(winner_df.Teams, winner_df.wins, color = ['blue','#FFD801','#461B7E','#C11B17','#F6
```



In [49]:

```
jovian.commit(project=project_name)
```

```
<IPython.core.display.Javascript object>

[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[49]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

Now lets See Season with Most Number Of Matches

In [55]:

```
season_df = ipl_df.groupby('Season')[['id']].count()
season_df = season_df.sort_values('Season', ascending=False).reset_index()
season_df.rename(columns = {'id':'Matches','Season':'Year'},inplace = True)
```

In [56]:

```
season_df
```

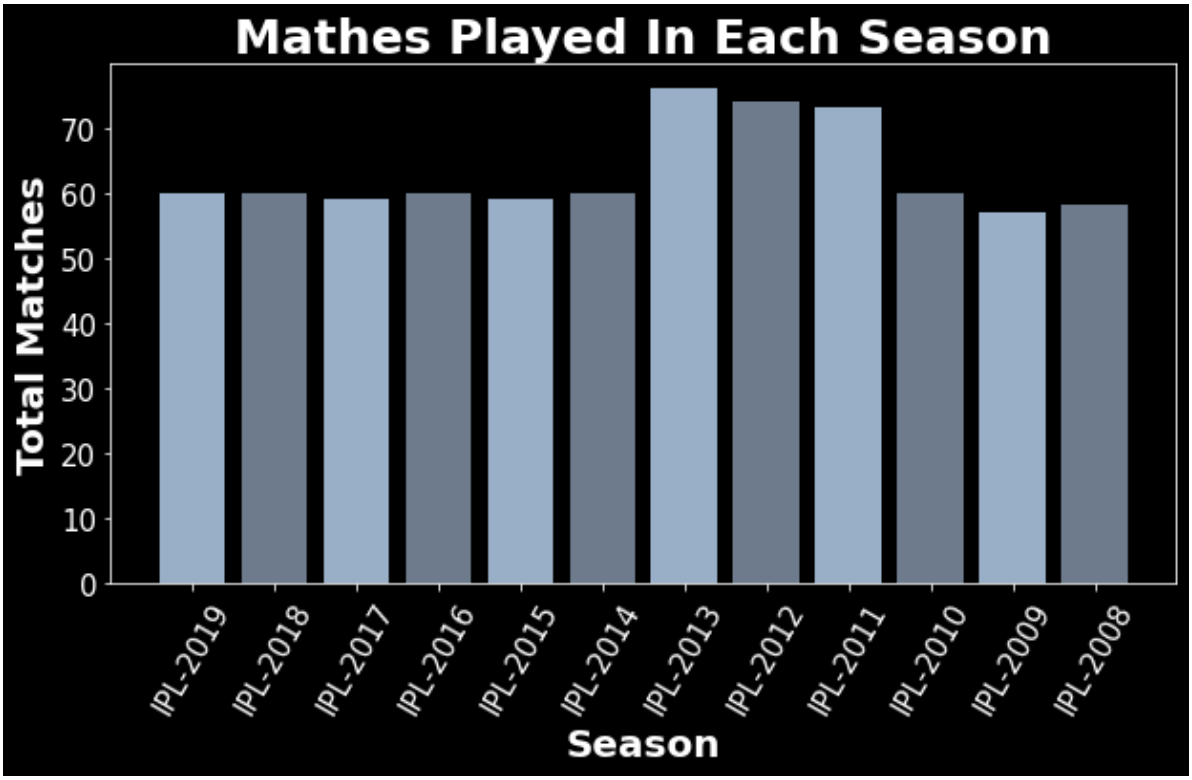
Out[56]:

| | Year | Matches |
|---|---|---|
| **0** | IPL-2019 | 60 |
| **1** | IPL-2018 | 60 |
| **2** | IPL-2017 | 59 |
| **3** | IPL-2016 | 60 |
| **4** | IPL-2015 | 59 |
| **5** | IPL-2014 | 60 |
| **6** | IPL-2013 | 76 |
| **7** | IPL-2012 | 74 |
| **8** | IPL-2011 | 73 |
| **9** | IPL-2010 | 60 |
| **10** | IPL-2009 | 57 |
| **11** | IPL-2008 | 58 |

Now Lets plot this Information

In [61]:

```
#To make it look more neat we will rotate the x-axis name with an angle of 60 using .xticks
# Also will make the font bold and increase its size for readability
plt.figure(figsize=(10,5))
plt.title("Mathes Played In Each Season",fontweight='bold',fontsize=25)
plt.xlabel('Season',fontweight='bold',fontsize=20)
plt.ylabel('Total Matches',fontweight='bold',fontsize=20)
plt.xticks(rotation='60')
plt.tick_params(labelsize=15)
plt.bar(season_df.Year,season_df.Matches,color=['#98AFC7','#6D7B8D']);
```




From The Above Graph its Clear the Season 2013 had most number of matches played (76)

Lets Commit our work and Move further with analysis

In [62]:

```
jovian.commit(project=project_name)
```

```
<IPython.core.display.Javascript object>

[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[62]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

# Section 3: Asking Interesting Questions on data

I will be asking following Questions:

1.What was the most preferred Decision On winning Toss i.e. Choose To Bat / Choose To Field 2.Which Decision has proved most beneficial i.e Field / Bat 3.Which Venue has hosted the Most Number Of Ipl Matches 4.Who has been awarded with Player Of the Max maximum Number Of Times 5.Who Has Won the Ipl Trophy Most Number of Times 6.Which Season had Most Number of Matches Played

## Q1. What was the most preferred Decision On winning Toss i.e. Bat / Field

In [63]:

```
ipl_df
```

Out[63]:

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

756 rows × 15 columns

In [64]:

```
# We can see toss decision is either bat/field
ipl_df.toss_decision.unique()
```

Out[64]:

```
array(['field', 'bat'], dtype=object)
```

In [65]:

```
decision_df = ipl_df.groupby('toss_decision')[['id']].count()
decision_df = decision_df.sort_values('id').reset_index()
decision_df.rename(columns={'id':'Total','toss_decision':'Decision'},inplace=True)
```

In [66]:

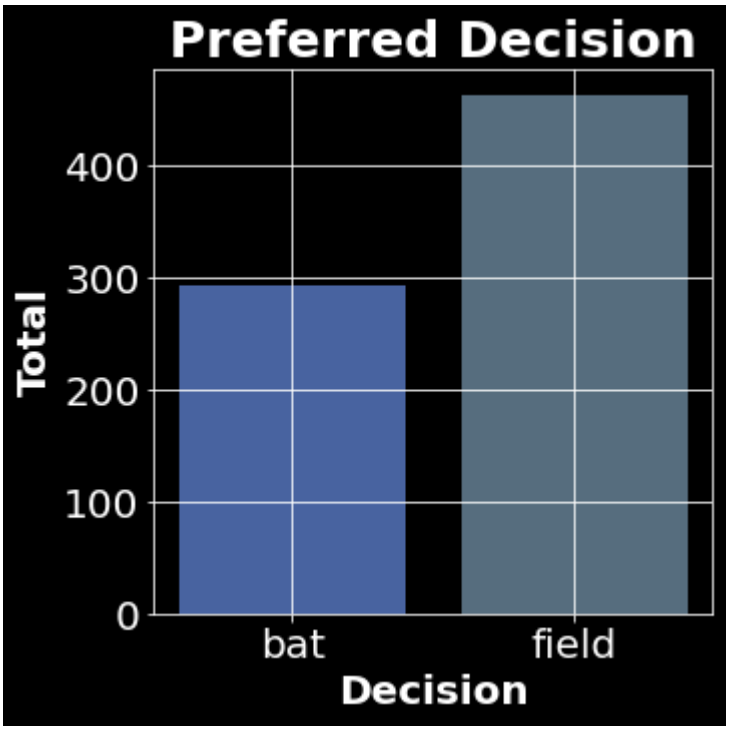
```
decision_df
```

Out[66]:

| | Decision | Total |
|---|---|---|
| **0** | bat | 293 |
| **1** | field | 463 |

In [68]:

```
#Lets plot the Result
plt.figure(figsize=(5,5))
plt.title("Preferred Decision",fontweight='bold',fontsize=25)
plt.xlabel('Decision',fontweight='bold',fontsize=20)
plt.ylabel('Total',fontweight='bold',fontsize=20)
plt.tick_params(labelsize=20)
plt.grid()
plt.bar(decision_df.Decision, decision_df.Total, color=['#4863A0','#566D7E']);
```




In [69]:

```
print('The Most Preferred Decision After Winning Toss in the IPL Until 2019 has been "Choos
```

```
The Most Preferred Decision After Winning Toss in the IPL Until 2019 has bee
n "Choose to Field First"
```

In [70]:

```
jovian.commit(project=project_name)
```

<IPython.core.display.Javascript object>

[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)

Out[70]:

'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'

## Q2. Which Decision has proved most beneficial i.e Field / Bat

In [71]:

```
field_df = ipl_df.loc[(ipl_df['toss_winner'] == ipl_df['winner']) &
                      (ipl_df['toss_decision'] == 'field'),
                      ['id', 'winner','toss_decision']]
```

In [73]:

```
field_df.winner.count()
```

Out[73]:

259

In [76]:

```
bat_df = ipl_df.loc[(ipl_df['toss_winner'] == ipl_df['winner']) &
                    (ipl_df['toss_decision'] == 'bat'),
                     ['id', 'winner','toss_decision']]
```

In [77]:

```
bat_df.winner.count()
```

Out[77]:

134

In [78]:

```
frames = [bat_df, field_df]
result_df = pd.concat(frames)
result_df = result_df.groupby('toss_decision')[['id']].count()
result_df
```

Out[78]:

| toss_decision | id |
|---|---|
| bat | 134 |
| field | 259 |

In [79]:

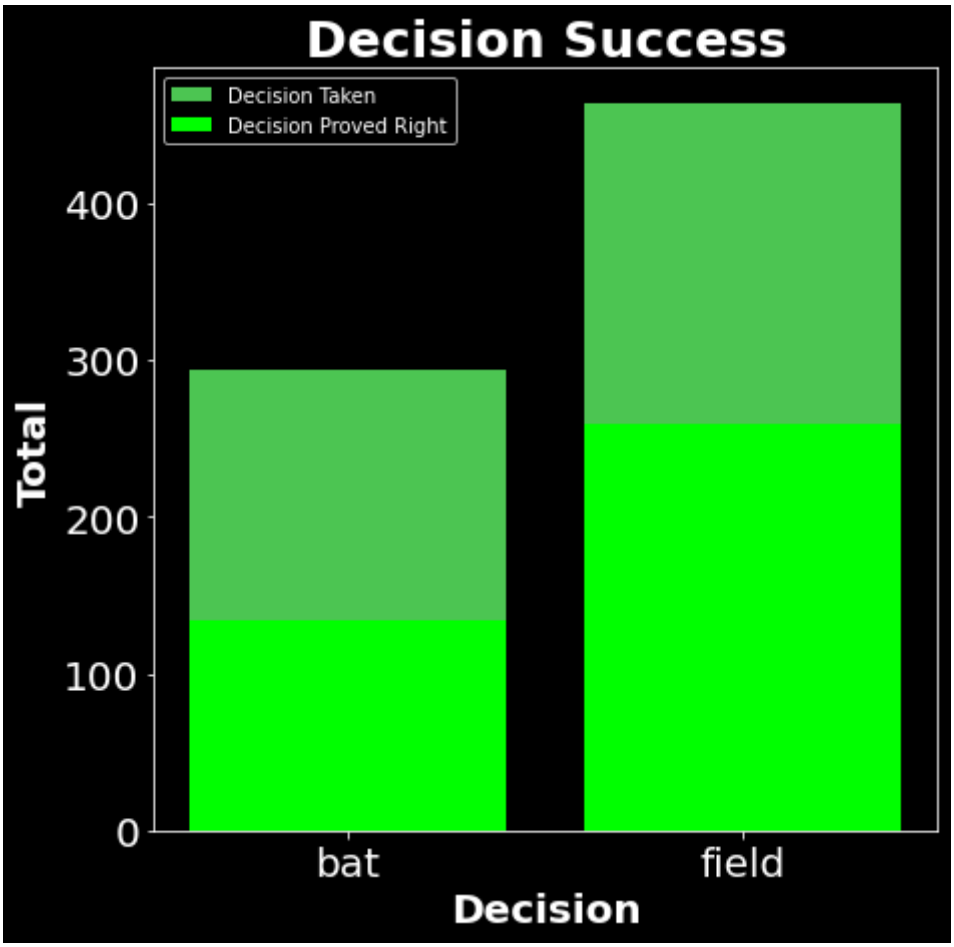
```
#As from Earlier Analysis we know out of 756 Toss that were tossed (2008 - 2019) "463 times
# Now Lets Plot the New Understanding Regarding the Success of these decisions
result_df = result_df.sort_values('id').reset_index()
result_df.rename(columns={'id':'Total','toss_decision':'Decision'},inplace=True)
result_df
```

Out[79]:

| | Decision | Total |
|---|---|---|
| 0 | bat | 134 |
| 1 | field | 259 |

In [80]:

```
plt.figure(figsize=(7,7))
plt.title("Decision Success",fontweight='bold',fontsize=25)
plt.xlabel('Decision',fontweight='bold',fontsize=20)
plt.ylabel('Total',fontweight='bold',fontsize=20)
plt.tick_params(labelsize=20)
plt.bar(decision_df.Decision, decision_df.Total, color=['#4CC552','#4CC552']);
plt.bar(result_df.Decision, result_df.Total, color=['#00FF00','#00FF00']);
plt.legend(['Decision Taken','Decision Proved Right']);
```




We can See the Fielding decision on winning toss has not only been most Preferred one But it has also proven to be a good Decision as almost 60% of the Time it is Proved Right

## Q3. Which Venue has hosted the Most Number Of Matches

In [81]:

```
# Lets see how many venues have hosted the Ipl Matches
ipl_df.venue.unique()
```

Out[81]:

```
array(['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'], dtype=object)
```

In [82]:

```
total_venue = list(ipl_df.venue.unique())
len(total_venue)
```

Out[82]:

```
41
```

We Can See ipl has hosted the Matches across 41 Different venues

Lets See Which Venue Hosted the Most Number Of Matches

In [83]:

```
venue_df = ipl_df.groupby('venue')[['id']].count()
venue_df = venue_df.sort_values('id',ascending=False).reset_index()
venue_df.rename(columns={'id':'Total','venue':'Stadium'},inplace=True)
```

In [84]:

```
labels = list(venue_df.Stadium)
venue_df
```

Out[84]:

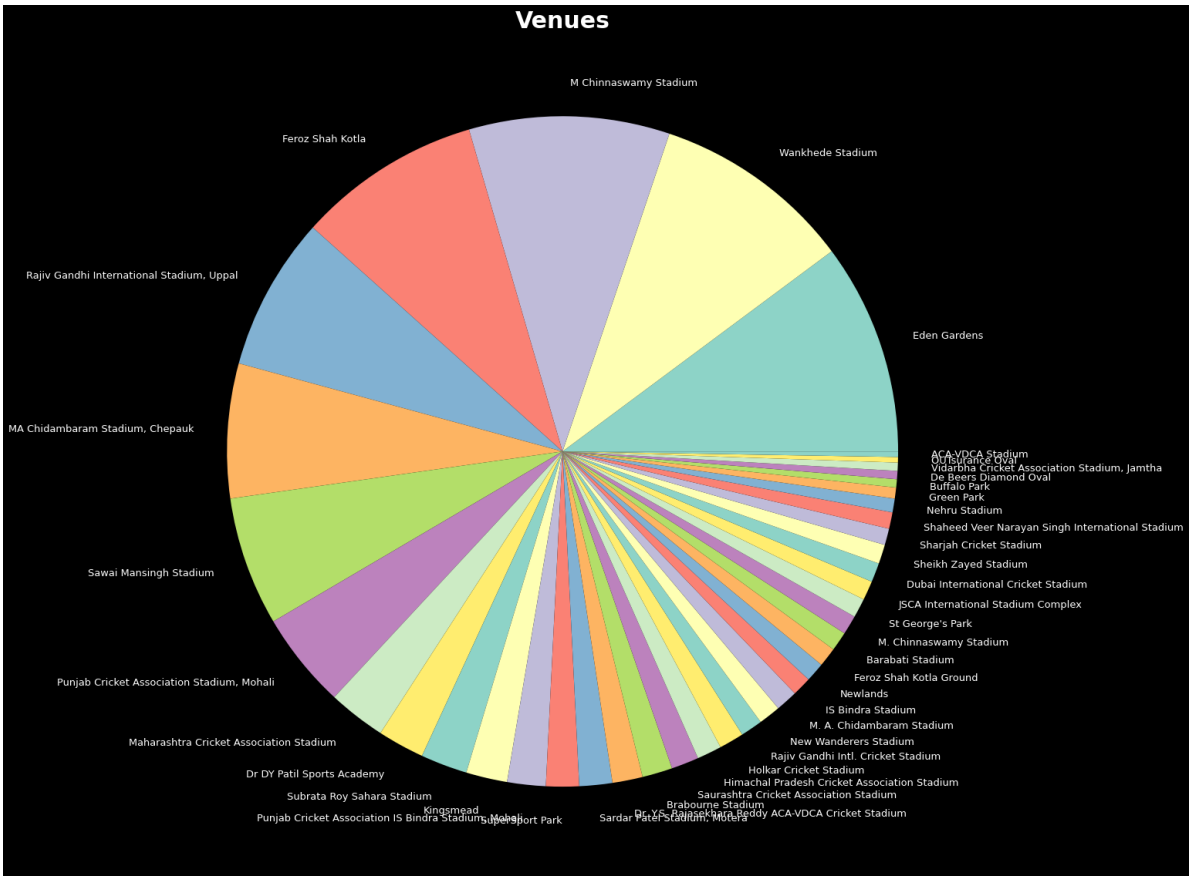
| | Stadium | Total |
|---|---|---|
| 0 | Eden Gardens | 77 |
| 1 | Wankhede Stadium | 73 |
| 2 | M Chinnaswamy Stadium | 73 |
| 3 | Feroz Shah Kotla | 67 |
| 4 | Rajiv Gandhi International Stadium, Uppal | 56 |
| 5 | MA Chidambaram Stadium, Chepauk | 49 |
| 6 | Sawai Mansingh Stadium | 47 |
| 7 | Punjab Cricket Association Stadium, Mohali | 35 |
| 8 | Maharashtra Cricket Association Stadium | 21 |
| 9 | Dr DY Patil Sports Academy | 17 |
| 10 | Subrata Roy Sahara Stadium | 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 | Himachal Pradesh Cricket Association Stadium | 9 |
| 19 | Holkar Cricket Stadium | 9 |
| 20 | Rajiv Gandhi Intl. Cricket Stadium | 8 |
| 21 | New Wanderers Stadium | 8 |
| 22 | M. A. Chidambaram Stadium | 8 |
| 23 | IS Bindra Stadium | 7 |
| 24 | Newlands | 7 |
| 25 | Feroz Shah Kotla Ground | 7 |
| 26 | Barabati Stadium | 7 |
| 27 | M. Chinnaswamy Stadium | 7 |
| 28 | St George's Park | 7 |
| 29 | JSCA International Stadium Complex | 7 |
| 30 | Dubai International Cricket Stadium | 7 |
| 31 | Sheikh Zayed Stadium | 7 |
| 32 | Sharjah Cricket Stadium | 6 |
| 33 | Shaheed Veer Narayan Singh International Stadium | 6 |

| | Stadium | Total |
|---|---|---|
| **34** | Nehru Stadium | 5 |
| **35** | Green Park | 4 |
| **36** | Buffalo Park | 3 |
| **37** | De Beers Diamond Oval | 3 |
| **38** | Vidarbha Cricket Association Stadium, Jamtha | 3 |
| **39** | OUTsurance Oval | 2 |
| **40** | ACA-VDCA Stadium | 2 |

As we have a long list We will only Take Top 10 Venues for our Graphical Representation

In [85]:

```
plt.figure(figsize=(20,20))
plt.title("Venues",fontweight='bold',fontsize=30)
plt.tick_params(labelsize=40)
plt.pie(venue_df.Total,labels=labels,textprops={'fontsize': 13});
```




So We can See the most Number of matches were played at Eden Gardens(77) Followed By Wankhede Stadium (73)

In [86]:

```
jovian.commit(project=project_name)
```

```
<IPython.core.display.Javascript object>

[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[86]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

## Q4. Who has been awarded with Player Of the Match maximum Number Of Times

In [87]:

```
#Lets Check how many players have been awarded with player of the match award
len(ipl_df.player_of_match.unique())
```

Out[87]:

```
227
```

This is Huge Number, we can see 227 Players have been awarded with player of the match title

Now Among these players lets see who have Got the maximum Player of The Match Awards

In [91]:

```
player_df = ipl_df.groupby('player_of_match')[['id']].count()
player_df = player_df.sort_values('id',ascending=False).reset_index()
player_df
```

Out[91]:

| | player_of_match | id |
|---|---|---|
| **0** | CH Gayle | 21 |
| **1** | AB de Villiers | 20 |
| **2** | MS Dhoni | 17 |
| **3** | DA Warner | 17 |
| **4** | RG Sharma | 17 |
| **...** | ... | ... |
| **221** | KMDN Kulasekara | 1 |
| **222** | KK Cooper | 1 |
| **223** | K Rabada | 1 |
| **224** | K Paul | 1 |
| **225** | Z Khan | 1 |

226 rows × 2 columns

In [92]:

```
#Now From these Players Lets Extract Top 10 Players
players_df = player_df.head(10).copy()
players_df.rename(columns={'id':'Total_Awards','player_of_match':'Man_Of_The_Match'},inplac
players_df
```

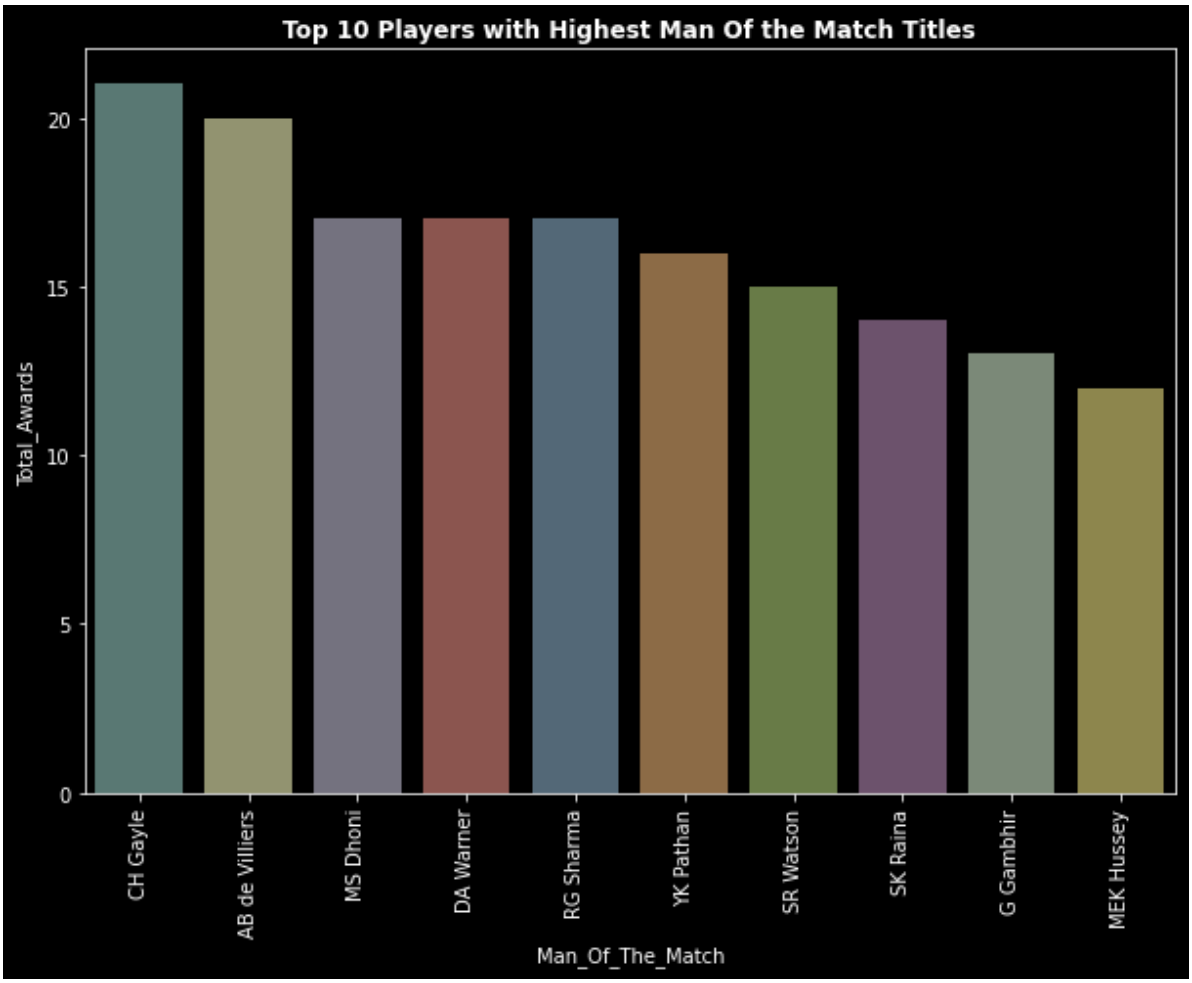
Out[92]:

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

From the above result it is clear that Chris Gayle has received "21 Man of The Match Titles" and is followed by AB de Villiers having "20"

In [95]:

```
plt.figure(figsize=(10,7))
plt.title("Top 10 Players with Highest Man Of the Match Titles",fontweight='bold' )
plt.xticks(rotation=90)
plt.yticks(ticks=np.arange(0,25,5))
plt.ylabel('No. of Awards')
plt.xlabel('Players')
sns.barplot(x=players_df.Man_Of_The_Match,y=players_df.Total_Awards, alpha=0.6);
```




In [96]:

```
jovian.commit(project=project_name)
```

```
<IPython.core.display.Javascript object>

[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[96]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

## Q5. Who Has Won the Ipl Trophy Most Number of Times

Now lets search For Team with Most Season Wins

We will have to extract the Final matches from the Entire Data

To do that we can sort the matches season wise and then select the last match of the season

In [97]:

```
final_df = ipl_df.groupby('Season').tail(1).copy()
```

In [98]:

```
final_df
```

Out[98]:

| | id | Season | city | date | team1 | team2 | toss_winner | toss_decision | ı |
|---|---|---|---|---|---|---|---|---|---|
| **58** | 59 | IPL-2017 | Hyderabad | 21-05-2017 | Mumbai Indians | Rising Pune Supergiant | Mumbai Indians | bat | n |
| **116** | 117 | IPL-2008 | Mumbai | 01-06-2008 | Chennai Super Kings | Rajasthan Royals | Rajasthan Royals | field | n |
| **173** | 174 | IPL-2009 | Johannesburg | 24-05-2009 | Deccan Chargers | Royal Challengers Bangalore | Royal Challengers Bangalore | field | n |
| **233** | 234 | IPL-2010 | Mumbai | 25-04-2010 | Chennai Super Kings | Mumbai Indians | Chennai Super Kings | bat | n |
| **306** | 307 | IPL-2011 | Chennai | 28-05-2011 | Chennai Super Kings | Royal Challengers Bangalore | Chennai Super Kings | bat | n |
| **380** | 381 | IPL-2012 | Chennai | 27-05-2012 | Chennai Super Kings | Kolkata Knight Riders | Chennai Super Kings | bat | n |
| **456** | 457 | IPL-2013 | Kolkata | 26-05-2013 | Mumbai Indians | Chennai Super Kings | Mumbai Indians | bat | n |
| **516** | 517 | IPL-2014 | Bengaluru | 01-06-2014 | Kings XI Punjab | Kolkata Knight Riders | Kolkata Knight Riders | field | n |
| **575** | 576 | IPL-2015 | Kolkata | 24-05-2015 | Mumbai Indians | Chennai Super Kings | Chennai Super Kings | field | n |
| **635** | 636 | IPL-2016 | Bengaluru | 29-05-2016 | Sunrisers Hyderabad | Royal Challengers Bangalore | Sunrisers Hyderabad | bat | n |
| **695** | 7953 | IPL-2018 | Mumbai | 27-05-2018 | Sunrisers Hyderabad | Chennai Super Kings | Chennai Super Kings | field | n |
| **755** | 11415 | IPL-2019 | Hyderabad | 12-05-2019 | Mumbai Indians | Chennai Super Kings | Mumbai Indians | bat | n |

In [99]:

```
#Now Lets sort The Data According to Seasons
final_df = final_df.sort_values('Season')
final_df
```

Out[99]:

| | id | Season | city | date | team1 | team2 | toss_winner | toss_decision | r |
|---|---|---|---|---|---|---|---|---|---|
| **116** | 117 | IPL-2008 | Mumbai | 01-06-2008 | Chennai Super Kings | Rajasthan Royals | Rajasthan Royals | field | n |
| **173** | 174 | IPL-2009 | Johannesburg | 24-05-2009 | Deccan Chargers | Royal Challengers Bangalore | Royal Challengers Bangalore | field | n |
| **233** | 234 | IPL-2010 | Mumbai | 25-04-2010 | Chennai Super Kings | Mumbai Indians | Chennai Super Kings | bat | n |
| **306** | 307 | IPL-2011 | Chennai | 28-05-2011 | Chennai Super Kings | Royal Challengers Bangalore | Chennai Super Kings | bat | n |
| **380** | 381 | IPL-2012 | Chennai | 27-05-2012 | Chennai Super Kings | Kolkata Knight Riders | Chennai Super Kings | bat | n |
| **456** | 457 | IPL-2013 | Kolkata | 26-05-2013 | Mumbai Indians | Chennai Super Kings | Mumbai Indians | bat | n |
| **516** | 517 | IPL-2014 | Bengaluru | 01-06-2014 | Kings XI Punjab | Kolkata Knight Riders | Kolkata Knight Riders | field | n |
| **575** | 576 | IPL-2015 | Kolkata | 24-05-2015 | Mumbai Indians | Chennai Super Kings | Chennai Super Kings | field | n |
| **635** | 636 | IPL-2016 | Bengaluru | 29-05-2016 | Sunrisers Hyderabad | Royal Challengers Bangalore | Sunrisers Hyderabad | bat | n |
| **58** | 59 | IPL-2017 | Hyderabad | 21-05-2017 | Mumbai Indians | Rising Pune Supergiant | Mumbai Indians | bat | n |
| **695** | 7953 | IPL-2018 | Mumbai | 27-05-2018 | Sunrisers Hyderabad | Chennai Super Kings | Chennai Super Kings | field | n |
| **755** | 11415 | IPL-2019 | Hyderabad | 12-05-2019 | Mumbai Indians | Chennai Super Kings | Mumbai Indians | bat | n |

In [100]:

```
final_df.winner.unique()
```

Out[100]:

```
array(['Rajasthan Royals', 'Deccan Chargers', 'Chennai Super Kings',
       'Kolkata Knight Riders', 'Mumbai Indians', 'Sunrisers Hyderabad'],
      dtype=object)
```

In [101]:

```
final_df['winner'].value_counts()
```

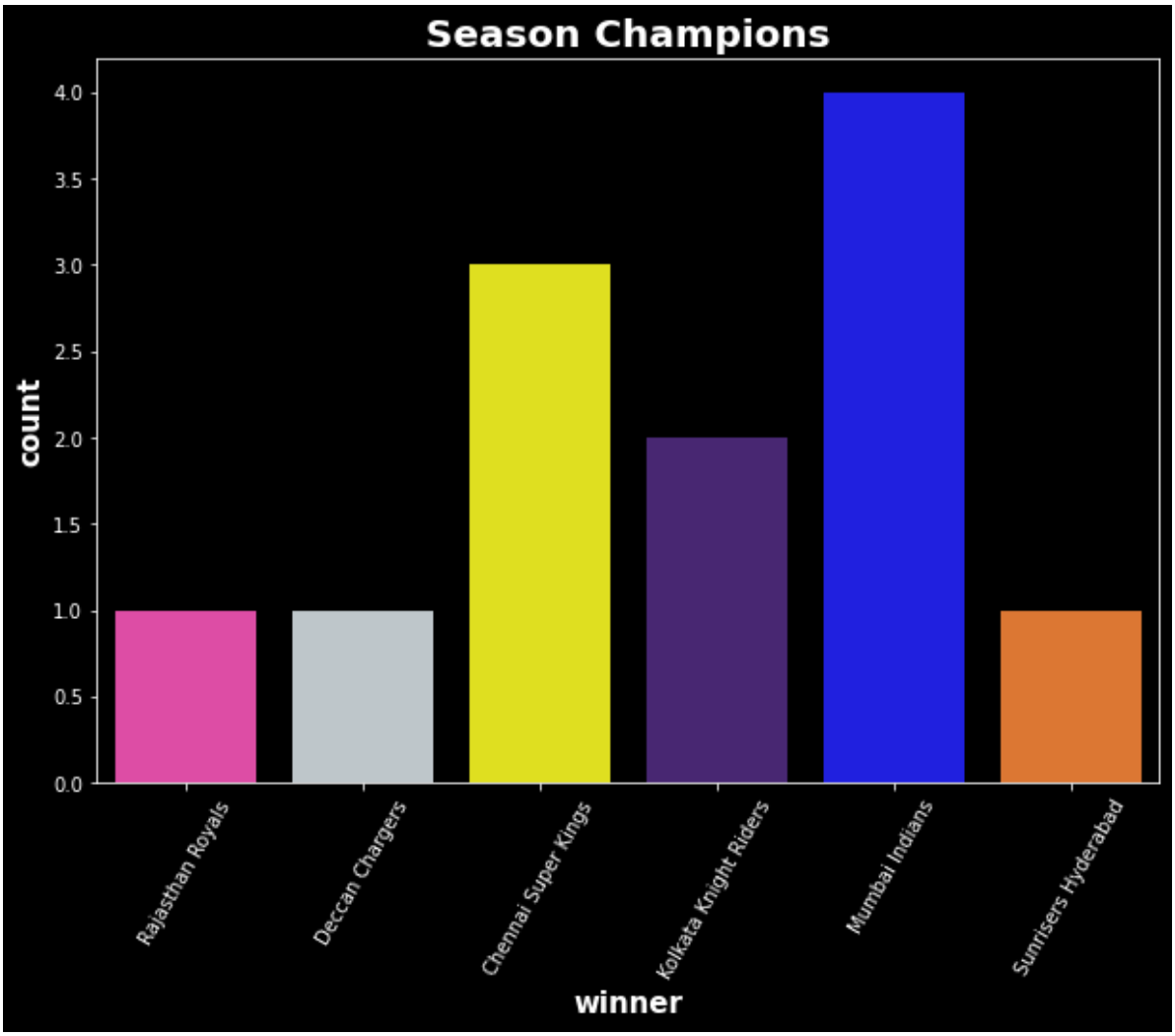
Out[101]:

```
Mumbai Indians           4
Chennai Super Kings      3
Kolkata Knight Riders    2
Rajasthan Royals         1
Deccan Chargers          1
Sunrisers Hyderabad      1
Name: winner, dtype: int64
```

We can See Mumbai Indians have Won the Most Season Titles till 2019

In [105]:

```
plt.figure(figsize=(10,7))
plt.title("Season Champions",fontweight='bold',fontsize=20)
plt.xlabel('Teams',fontweight='bold',fontsize=15)
plt.ylabel('Total Seasons',fontweight='bold',fontsize=15)
plt.xticks(rotation='60')
plt.tick_params(labelsize=10)
sns.countplot(x=final_df['winner'],palette=['#F535AA','#BCC6CC','yellow','#461B7E','blue','
```




In [106]:

```
jovian.commit(project=project_name)
```

```
<IPython.core.display.Javascript object>

[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[106]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

## Q6. Which Season Had Most Number of Matches

In [107]:

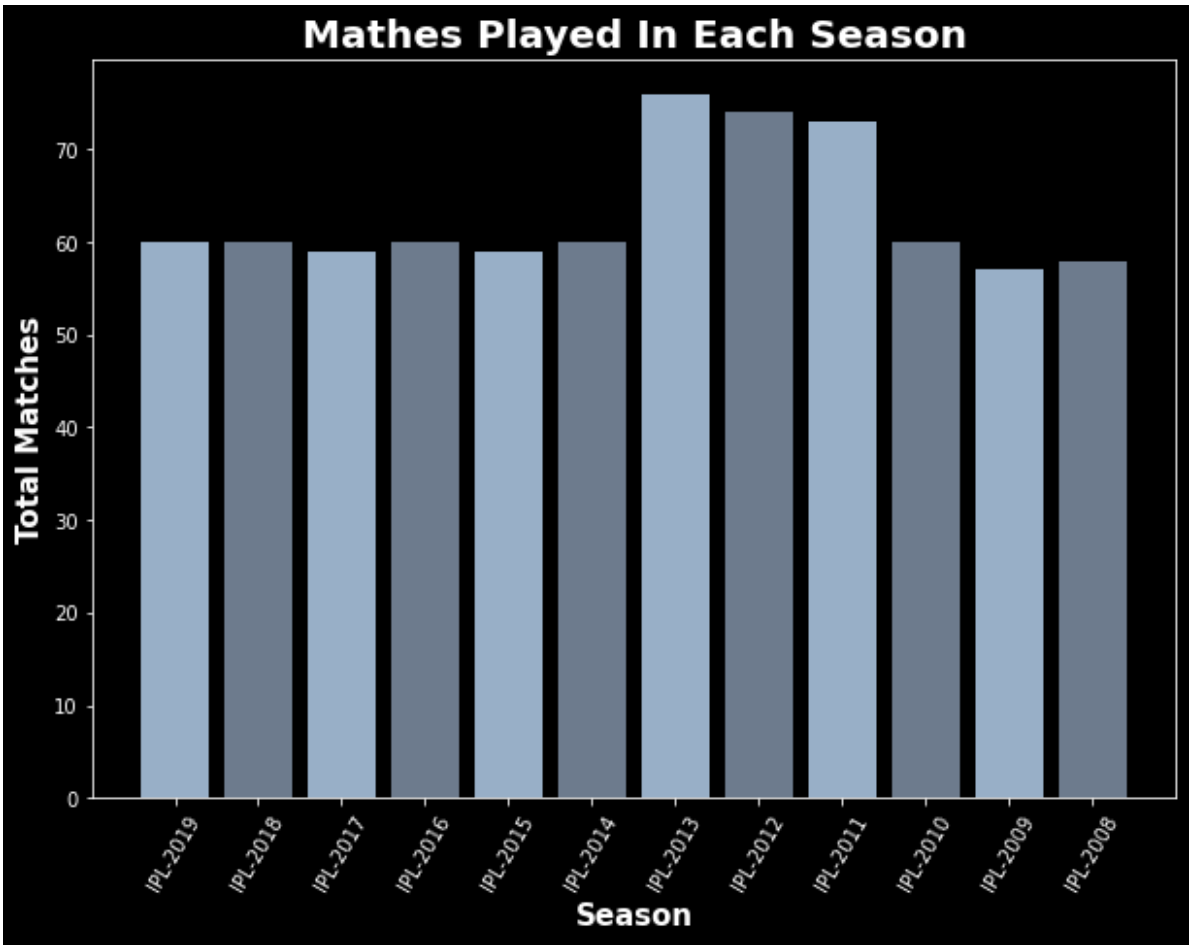
```
#This we had Explored Earlier
season_df
```

Out[107]:

| | Year | Matches |
|---|---|---|
| **0** | IPL-2019 | 60 |
| **1** | IPL-2018 | 60 |
| **2** | IPL-2017 | 59 |
| **3** | IPL-2016 | 60 |
| **4** | IPL-2015 | 59 |
| **5** | IPL-2014 | 60 |
| **6** | IPL-2013 | 76 |
| **7** | IPL-2012 | 74 |
| **8** | IPL-2011 | 73 |
| **9** | IPL-2010 | 60 |
| **10** | IPL-2009 | 57 |
| **11** | IPL-2008 | 58 |

In [110]:

```
#To make it look more neat we will rotate the x-axis name with an angle of 60 using .xticks
# Also will make the font bold and increase its size for readability
plt.figure(figsize=(10,7))
plt.title("Mathes Played In Each Season",fontweight='bold',fontsize=20)
plt.xlabel('Season',fontweight='bold',fontsize=15)
plt.ylabel('Total Matches',fontweight='bold',fontsize=15)
plt.xticks(rotation='60')
plt.tick_params(labelsize=10)
plt.bar(season_df.Year,season_df.Matches,color=['#98AFC7','#6D7B8D']);
```




## Section 4: Inferences and Conclusion

In this analysis I used the matches.csv file from the kaggle Datasets. Following are my conclusions about it

1.A total of 756 matches have been played from 2008 - 2019 2.Out of these 756 matches 743 matches were played normally and had a normal result 3.Most number of Matches were played in Mumbai [101] 4.Mumbai Indian's Have Won the Most Number of Matches (109) followed by Chennai Super Kings with 100 Matches 5.IPL-2013 Season Hosted most Number of Matches (76) 6.Eden Gardens (Stadium) Hosted the Most Number of Matches (77) followed by wankhede Stadium (73) 7.Chris Gayle has been the Man Of The Match Most Number of Times with "21" Awards followed by AB di Villiers (20) and MS Dhoni (17) 8.Mumbai Indians Have been the IPL Champions Most number of times (4) followed by Chennai Super Kings (3) 9.Mumbai Indians and Chennai Super Kings have been the dominant Teams

In [112]:

```
jovian.commit(project=project_name)
```

```
<IPython.core.display.Javascript object>
```

```
[jovian] Updating notebook "melrick-pais98/ipl-data-analysis-and-visualizati
on" on https://jovian.ai (https://jovian.ai)
[jovian] Committed successfully! https://jovian.ai/melrick-pais98/ipl-data-a
nalysis-and-visualization (https://jovian.ai/melrick-pais98/ipl-data-analysi
s-and-visualization)
```

Out[112]:

```
'https://jovian.ai/melrick-pais98/ipl-data-analysis-and-visualization'
```

## References

In [ ]:

```

```