Exploratory Data Analysis IPL 2022

Importing important libraries

In [1]:
```
#loding the required libraries for analysis
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Loading IPL dataset

In [2]:
```
# settings to display all columns
pd.set_option("display.max_columns", None)
pd.set_option("display.max_rows", None)
```

In [3]:
```
#loding the ipl matches dataset for analysis
ipl = pd.read_csv("C:/Users/navna/Downloads/tata_ipl_2022.csv")
```

In [4]:
```
#having the glance at the all records of the dataset
ipl
```

59	22-05-2022 16:07	60	2022	Mumbai	13-05-2022	Punjab Kings	Royal Challengers Bangalore	Royal Challengers Bangalore	bowl	normal	no	Punjab Kings	5
60	22-05-2022 16:10	61	2022	Pune	14-05-2022	Kolkata Knight Riders	Sunrisers Hyderabad	Kolkata Knight Riders	bat	normal	no	Kolkata Knight Riders	5
61	22-05-2022 16:10	62	2022	Mumbai	15-05-2022	Chennai Super Kings	Gujarat Titans	Chennai Super Kings	bat	normal	no	Gujarat Titans	0
62	22-05-2022 16:14	63	2022	Mumbai	15-05-2022	Rajasthan Royals	Lucknow Super Giants	Rajasthan Royals	bat	normal	no	Rajasthan Royals	2
63	22-05-2022 16:14	64	2022	Navi Mumbai	16-05-2022	Delhi Capitals	Punjab Kings	Punjab Kings	bowl	normal	no	Delhi Capitals	1
64	22-05-2022 16:17	65	2022	Mumbai	17-05-2022	Sunrisers Hyderabad	Mumbai Indians	Mumbai Indians	bowl	normal	no	Sunrisers Hyderabad	3
65	22-05-2022 16:17	66	2022	Navi Mumbai	18-05-2022	Lucknow Super Giants	Kolkata Knight Riders	Lucknow Super Giants	bat	normal	no	Lucknow Super Giants	2
66	22-05-2022 16:20	67	2022	Mumbai	19-05-	Gujarat Titans	Royal Challengers	Gujarat Titans	bat	normal	no	Royal Challengers	0

In [5]:
```
ipl.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 74 entries, 0 to 73
Data columns (total 19 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   Timestamp            74 non-null     object
 1   Match Number         74 non-null     int64 
 2   Season               74 non-null     int64 
 3   City                 74 non-null     object
 4   Date                 74 non-null     object
 5   Team 1               74 non-null     object
 6   Team 2               74 non-null     object
 7   Toss Winner          74 non-null     object
 8   Toss Decision        74 non-null     object
 9   Match Result         74 non-null     object
 10  DL Applied           74 non-null     object
 11  Match Winner         74 non-null     object
 12  Win by runs          74 non-null     int64 
 13  Win by wickets       74 non-null     int64 
 14  Player of the Match  74 non-null     object
 15  Venue                74 non-null     object
 16  Umpire 1             74 non-null     object
 17  Umpire 2             74 non-null     object
 18  Umpire 3             74 non-null     object
dtypes: int64(4), object(15)
memory usage: 11.1+ KB
```

In [6]:
```
#Rename specific columns
ipl.rename(columns = {'Timestamp':'timestamp','Match Number':'match_number','Season':'season',
                      'City':'city','Date':'date','Team 1':'team_1','Team 2':'team_2','Toss Winner':'toss_w
                      'Toss Decision':'toss_decision','Match Result':'match_result','DL Applied':'dl_applie
                      'Match Winner':'match_winner','Win by runs':'win_by_runs','Win by wickets':'win_by_wi
                      'Player of the Match':'player_of_the_match','Venue':'venue','Umpire 1':'umpire_1',
                      'Umpire 2':'umpire_2','Umpire 3':'umpire_3'},inplace = True)
```

In [7]:
```
#having a glance at the first five records of the dataset
ipl.head()
```

	timestamp	match_number	season	city	date	team_1	team_2	toss_winner	toss_decision	match_result	dl_appli
0	26-03-2022 23:43	1	2022	Mumbai	26-03-2022	Chennai Super Kings	Kolkata Knight Riders	Kolkata Knight Riders	bowl	normal	no
1	27-03-2022 19:37	2	2022	Mumbai	27-03-2022	Mumbai Indians	Delhi Capitals	Delhi Capitals	bowl	normal	no
2	27-03-2022 23:44	3	2022	Navi Mumbai	27-03-2022	Royal Challengers Bangalore	Punjab Kings	Punjab Kings	bowl	normal	no
3	29-03-2022 00:21	4	2022	Mumbai	29-03-2022	Lucknow Super Giants	Gujarat Titans	Gujarat Titans	bowl	normal	no
4	29-03-2022 23:41	5	2022	Pune	29-03-2022	Rajasthan Royals	Sunrisers Hyderabad	Sunrisers Hyderabad	bowl	normal	no

How big is the dataset? (Rows and columns)

In [8]:
```
# Looking at the number of rows and columns in the dataset
ipl.shape
```

```
(74, 19)
```

In [9]:
```
# get the info of dataset
ipl.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 74 entries, 0 to 73
Data columns (total 19 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   timestamp            74 non-null     object
 1   match_number         74 non-null     int64 
 2   season               74 non-null     int64 
 3   city                 74 non-null     object
 4   date                 74 non-null     object
 5   team_1               74 non-null     object
 6   team_2               74 non-null     object
 7   toss_winner          74 non-null     object
 8   toss_decision        74 non-null     object
 9   match_result         74 non-null     object
 10  dl_applied           74 non-null     object
 11  match_winner         74 non-null     object
 12  win_by_runs          74 non-null     int64 
 13  win_by_wickets       74 non-null     int64 
 14  player_of_the_match  74 non-null     object
 15  venue                74 non-null     object
 16  umpire_1             74 non-null     object
 17  umpire_2             74 non-null     object
 18  umpire_3             74 non-null     object
dtypes: int64(4), object(15)
memory usage: 11.1+ KB
```

Data Pre-processing: Finding out NaN values

In [10]:
```
ipl.isna().any()
```

```
timestamp             False
match_number          False
season                False
city                  False
date                  False
team_1                False
team_2                False
toss_winner           False
toss_decision         False
match_result          False
dl_applied            False
match_winner          False
win_by_runs           False
win_by_wickets        False
player_of_the_match   False
venue                 False
umpire_1              False
umpire_2              False
umpire_3              False
dtype: bool
```

Statistical Description of dataset

In [11]:
```
ipl.describe()
```

	match_number	season	win_by_runs	win_by_wickets
count	74.000000	74.0	74.000000	74.000000
mean	37.500000	2022.0	13.972973	3.000000
std	21.505813	0.0	21.464831	3.226602
min	1.000000	2022.0	0.000000	0.000000
25%	19.250000	2022.0	0.000000	0.000000
50%	37.500000	2022.0	1.000000	1.500000
75%	55.750000	2022.0	18.000000	6.000000
max	74.000000	2022.0	91.000000	9.000000

How many matches (in total) were played according to the dataset?

In [12]:
```
ipl['match_number'].count()
```

```
74
```

How many IPL seasons are we using to analyse?

In [13]:
```
ipl['season'].unique()
```

```
array([2022], dtype=int64)
```

In [14]:
```
len(ipl['season'].unique())
```

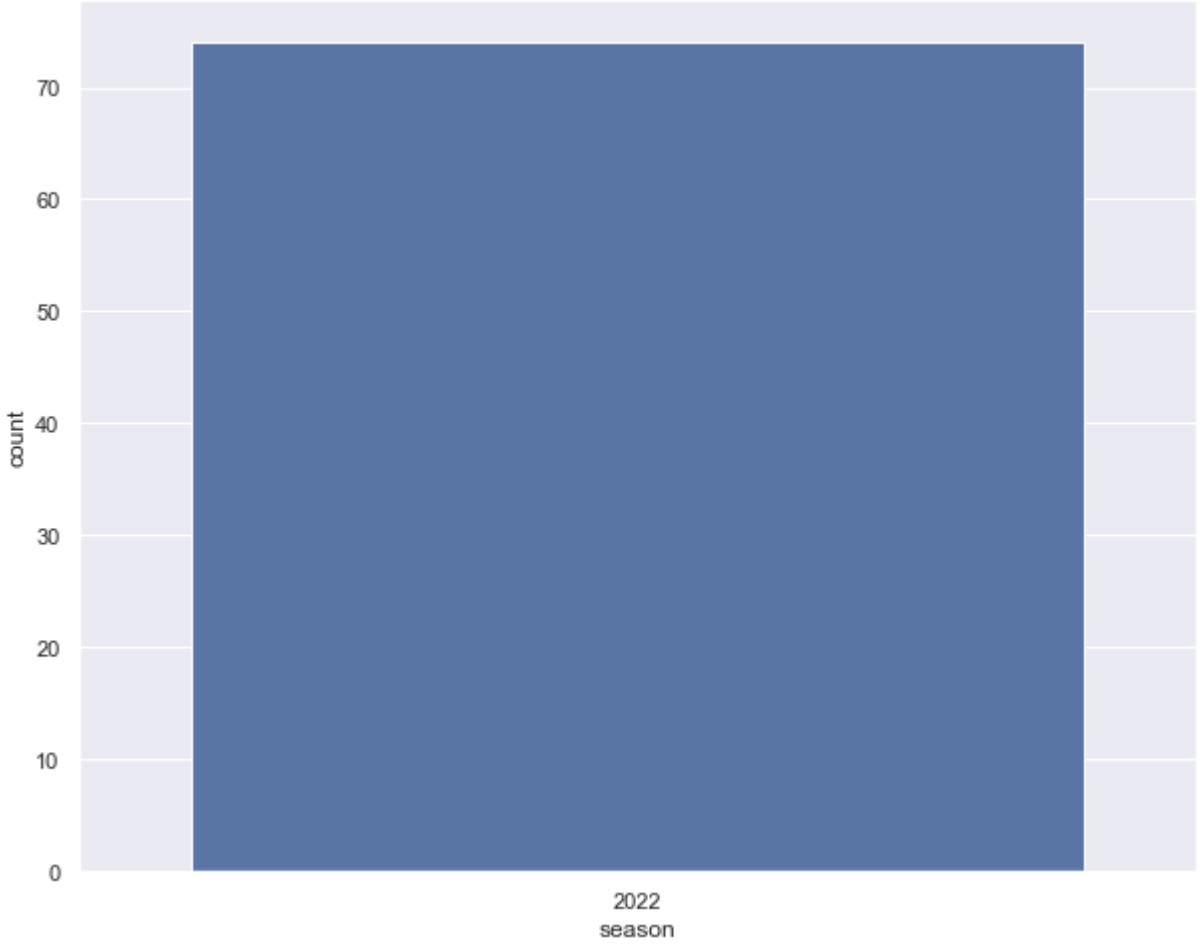
```
1
```

In which season how many the number of matches played?

In [15]:
```
#Looking at the number of matches played in each season
ipl['season'].value_counts()
```

```
2022    74
Name: season, dtype: int64
```

In [16]:
```
from matplotlib import rcParams
rcParams['figure.figsize']=10,8
sns.set_theme(style="darkgrid")
sns.countplot(x='season', data=ipl)
plt.show()
```



Man of the match - Highest to lowest (in won matches)

In [17]:
```
#Getting the frequency of most man of the match awards
ipl['player_of_the_match'].value_counts()
```

```
Jos Buttler           3
Kuldeep Yadav         3
Umesh Yadav           2
Dinesh Karthik        2
Jasprit Bumrah        2
Yuzvendra Chahal      2
KL Rahul              2
Umran Malik           2
David Miller          2
Shubman Gill          2
Quinton de Kock       2
Avesh Khan            2
Hardik Pandya         2
Wanindu Hasaranga     2
Rahul Tripathi        2
Tim David             1
David Warner          1
Harshal Patel         1
Kagiso Rabada         1
Rinku Singh           1
Ruturaj Gaikwad       1
Mitchel Marsh         1
Yashasvi Jaiswal      1
Devon Conway          1
Suryakumar Yadav      1
Daniel Sams           1
Jonny Bairstow        1
Andre Russell         1
Wriddhiman Saha       1
Trent Boult           1
Shardul Thakur        1
Virat Kohli           1
Ravichandran Ashwin   1
Harpreet Brar         1
Mohsin Khan           1
Mukesh Choudhary      1
Rahul Tewatia         1
```

```
Anuj Rawat            1
Odean Smith           1
Mohammed Shami        1
Sanju Samson          1
Evin Lewis            1
Lockie Ferguson       1
Liam Livingstone      1
Pat Cummins           1
Abhishek Sharma       1
 Kuldeep Yadav        1
Krunal Pandya         1
Kane Williamson       1
Shivam Dube           1
Mayank Agarwal        1
Faf du Plessis        1
Rashid Khan           1
Marco Jensen          1
Shikhar Dhawan        1
Riyan Parag           1
Rajat Patidar         1
Name: player_of_the_match, dtype: int64
```

Top player of the match Winners

In [18]:

```
#Getting the top 10 players with most man of the match awards
ipl['player_of_the_match'].value_counts()[0:10]
```

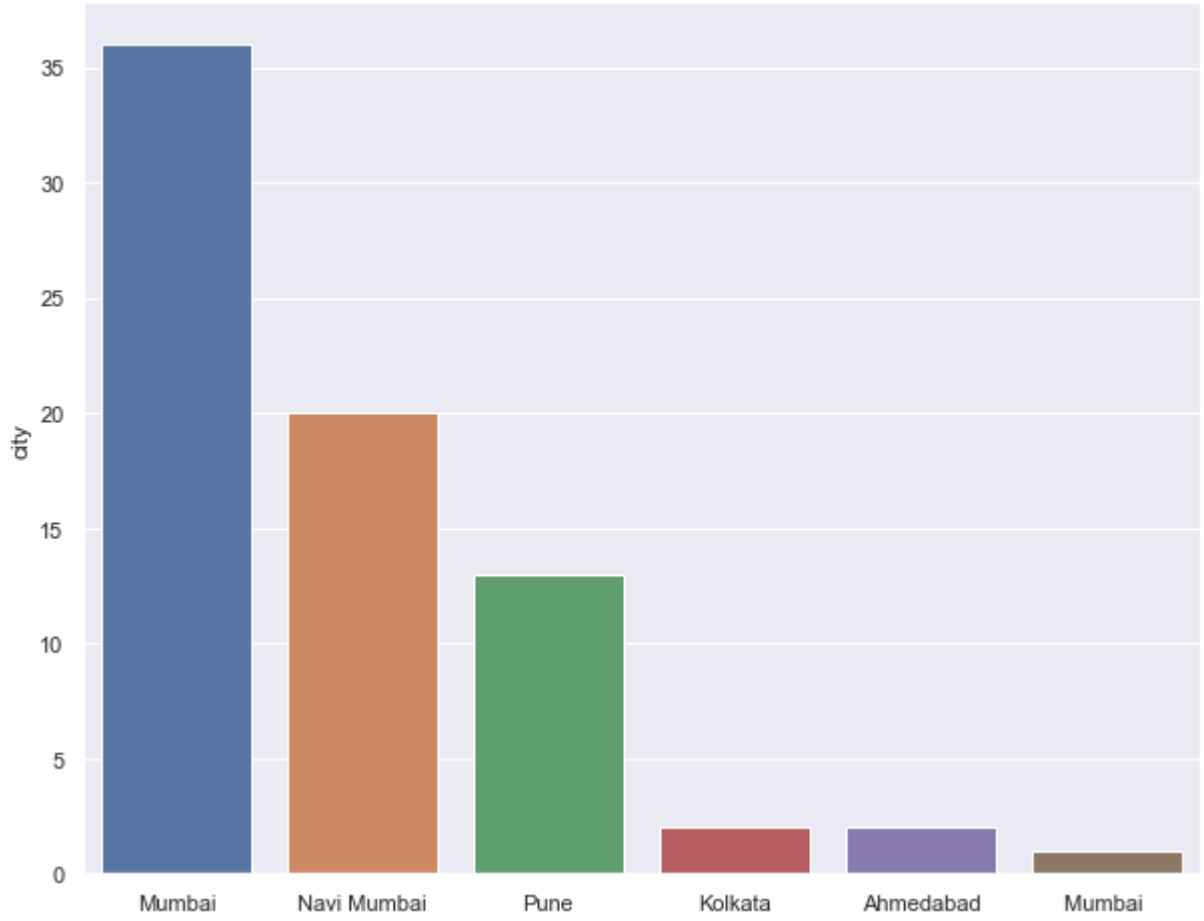
```
Jos Buttler         3
Kuldeep Yadav       3
Umesh Yadav         2
Dinesh Karthik      2
Jasprit Bumrah      2
Yuzvendra Chahal    2
KL Rahul            2
Umran Malik         2
David Miller        2
Shubman Gill        2
Name: player_of_the_match, dtype: int64
```

In which city were the number of matches played?

In [19]:
```
#Looking at the number of matches played in each city
ipl['city'].value_counts()
```

```
Mumbai         36
Navi Mumbai    20
Pune           13
Kolkata         2
Ahmedabad       2
Mumbai          1
Name: city, dtype: int64
```

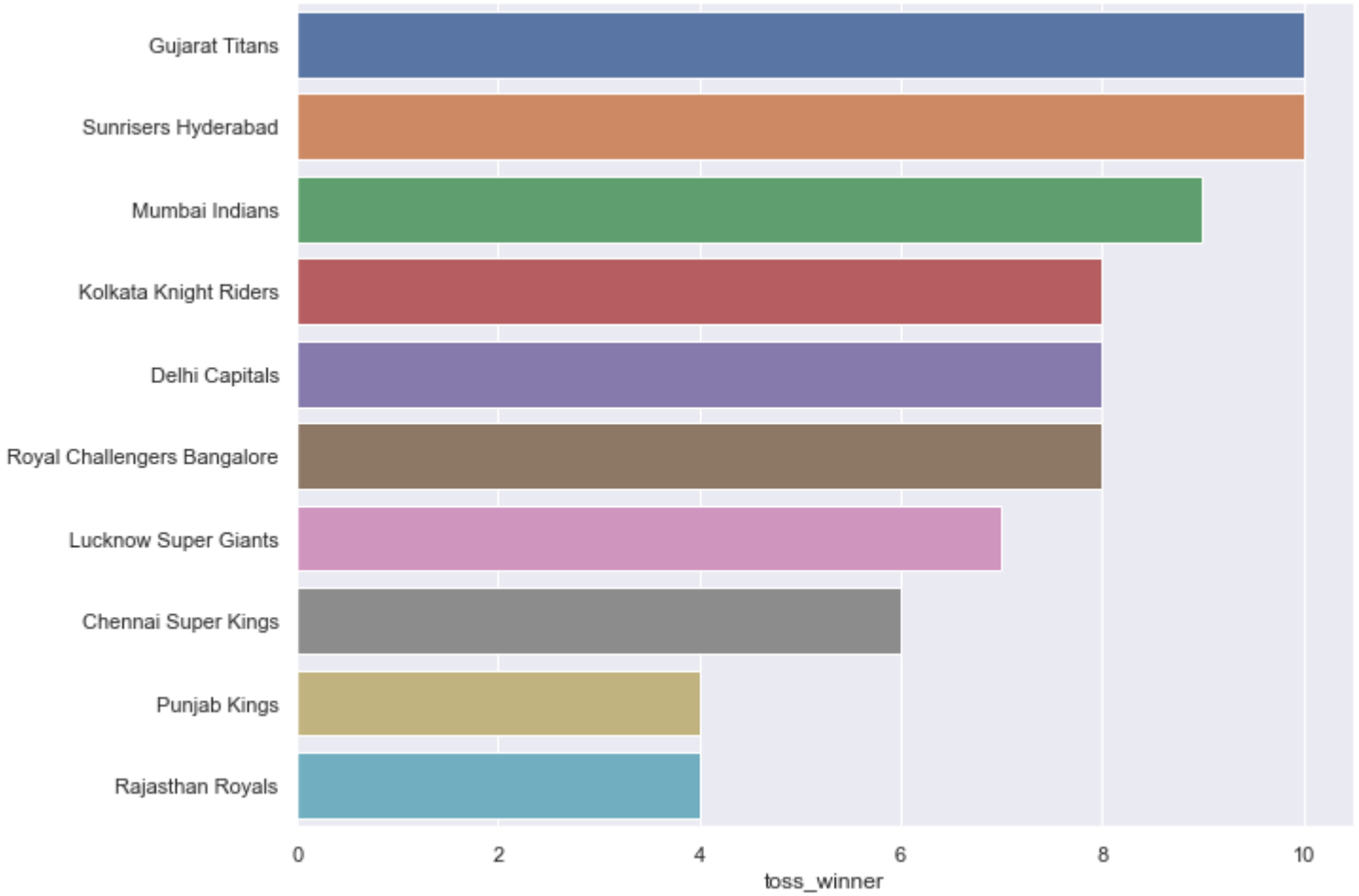
In [20]:
```
data = ipl.city.value_counts()
sns.barplot(y = data, x = data.index, orient='v');
```



In [21]:
```
#Getting the frequency of toss_winner column
ipl['toss_winner'].value_counts()
```

```
Gujarat Titans                10
Sunrisers Hyderabad           10
Mumbai Indians                 9
Kolkata Knight Riders          8
Delhi Capitals                 8
Royal Challengers Bangalore    8
Lucknow Super Giants           7
Chennai Super Kings            6
Punjab Kings                   4
Rajasthan Royals               4
Name: toss_winner, dtype: int64
```

In [22]:
```
data = ipl.toss_winner.value_counts()
sns.barplot(y = data.index, x = data, orient='h');
```



In [23]:
```
#Getting the frequency of toss_decision column
ipl['toss_decision'].value_counts()
```

```
bowl    59
bat     15
Name: toss_decision, dtype: int64
```

In [24]:
```
#Getting the frequency of result column
ipl['match_result'].value_counts()
```

```
normal    74
Name: match_result, dtype: int64
```

In [25]:
```
#Getting the frequency of dl_applied column
ipl['dl_applied'].value_counts()
```

```
no    74
Name: dl_applied, dtype: int64
```

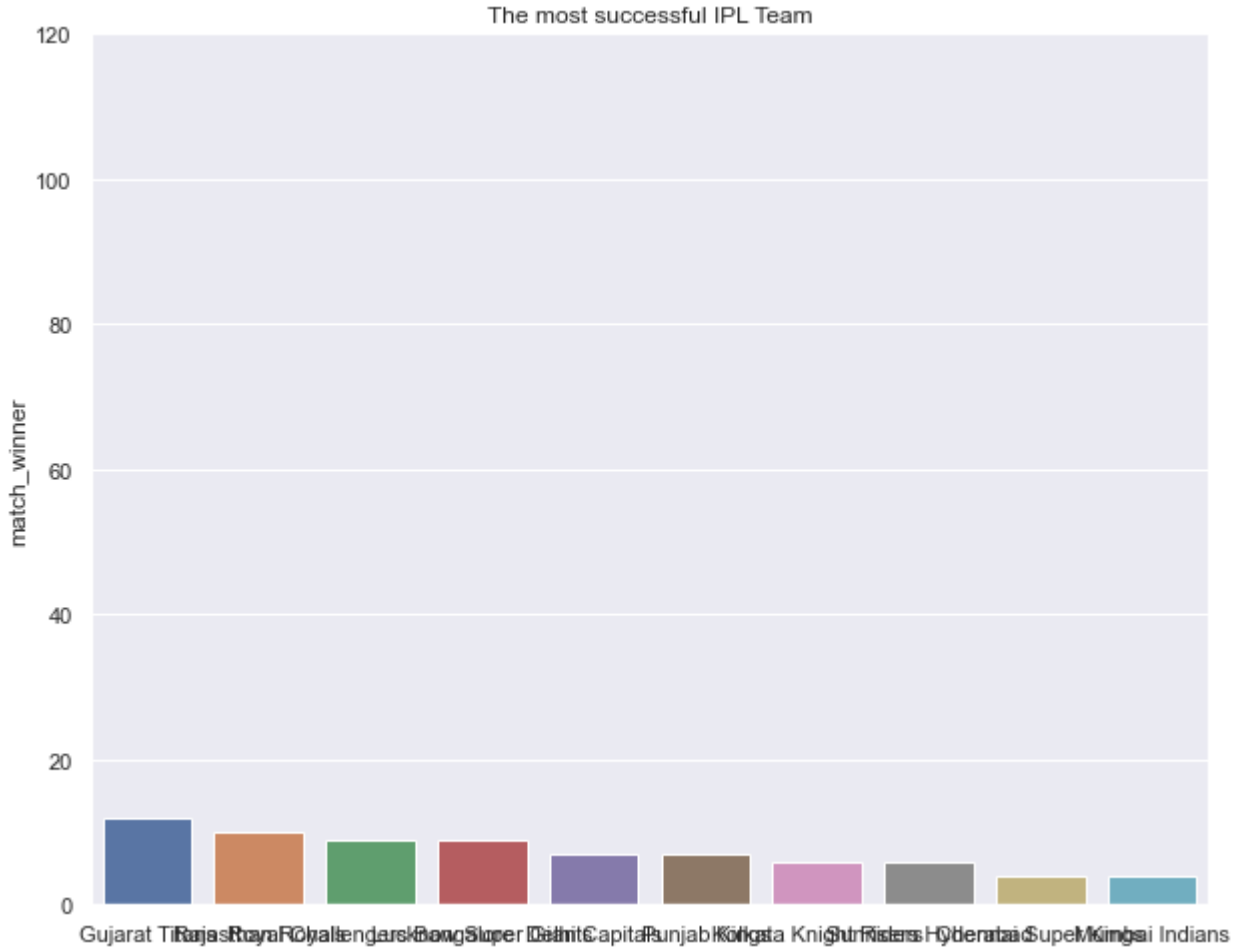
The most successful IPL Team

In [26]:
```
#Getting the frequency of winner column
ipl['match_winner'].value_counts()
```

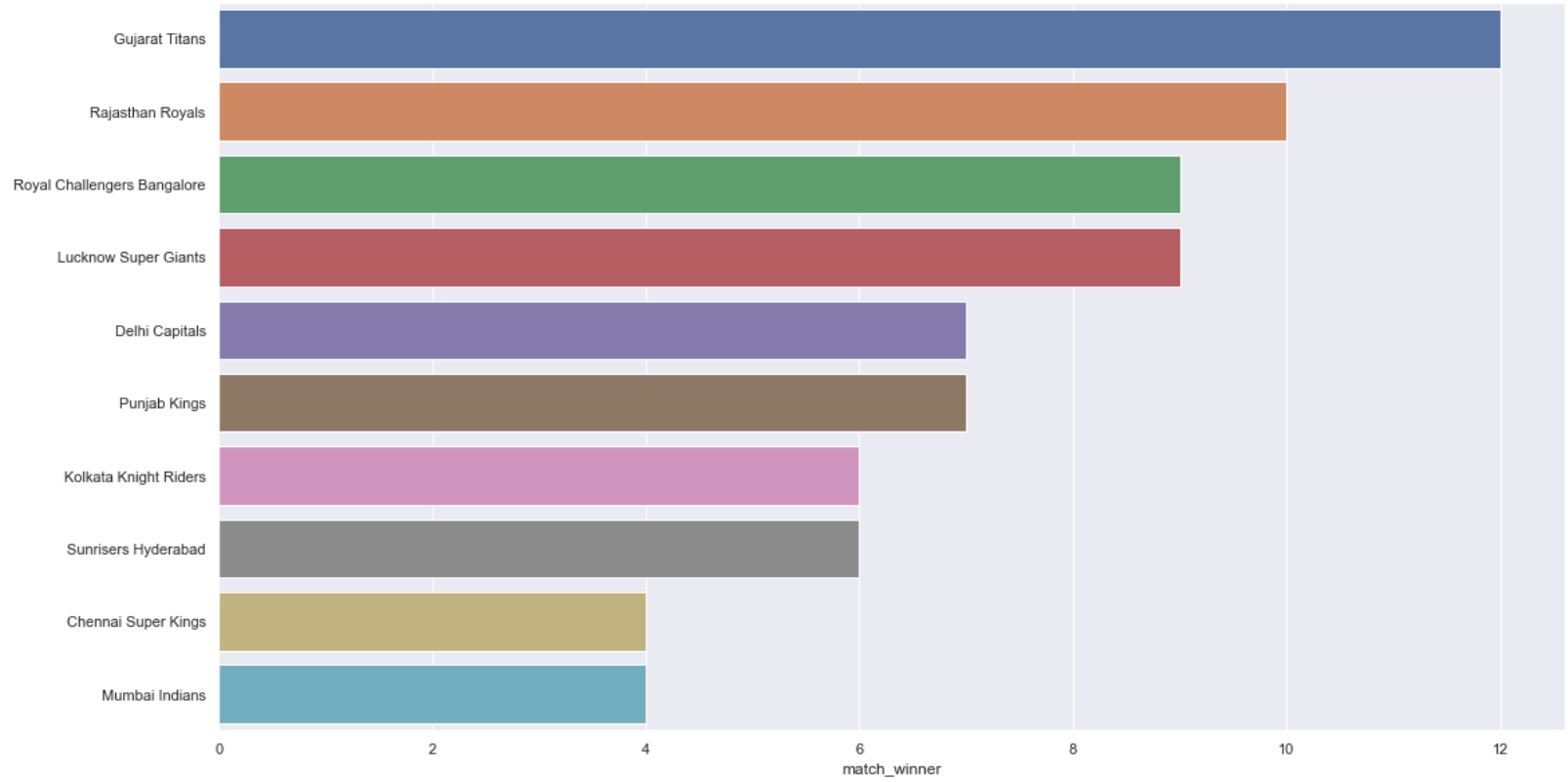
```
Gujarat Titans                12
Rajasthan Royals              10
Royal Challengers Bangalore    9
Lucknow Super Giants           9
Delhi Capitals                 7
Punjab Kings                   7
Kolkata Knight Riders          6
Sunrisers Hyderabad            6
Chennai Super Kings            4
Mumbai Indians                 4
Name: match_winner, dtype: int64
```

In [27]:

```
most_successful_team = ipl.match_winner.value_counts()#[:8]
fig, ax = plt.subplots()
ax.set_ylim([0,120])
from matplotlib import rcParams
rcParams['figure.figsize']=18,10
ax.set_ylabel("Count")
ax.set_title("The most successful IPL Team")
sns.barplot(x = most_successful_team.index, y = most_successful_team, orient='v');
plt.show()
```


The most successful IPL Team
match_winner
120
100
80
60
40
20
0

In [28]:
```
#sns.countplot(y='winner', data = matches)
#plt.show
from matplotlib import rcParams
rcParams['figure.figsize']=18,10
data = ipl.match_winner.value_counts()
sns.barplot(y = data.index, x = data, orient='h');
```



Highest wins by teams per season

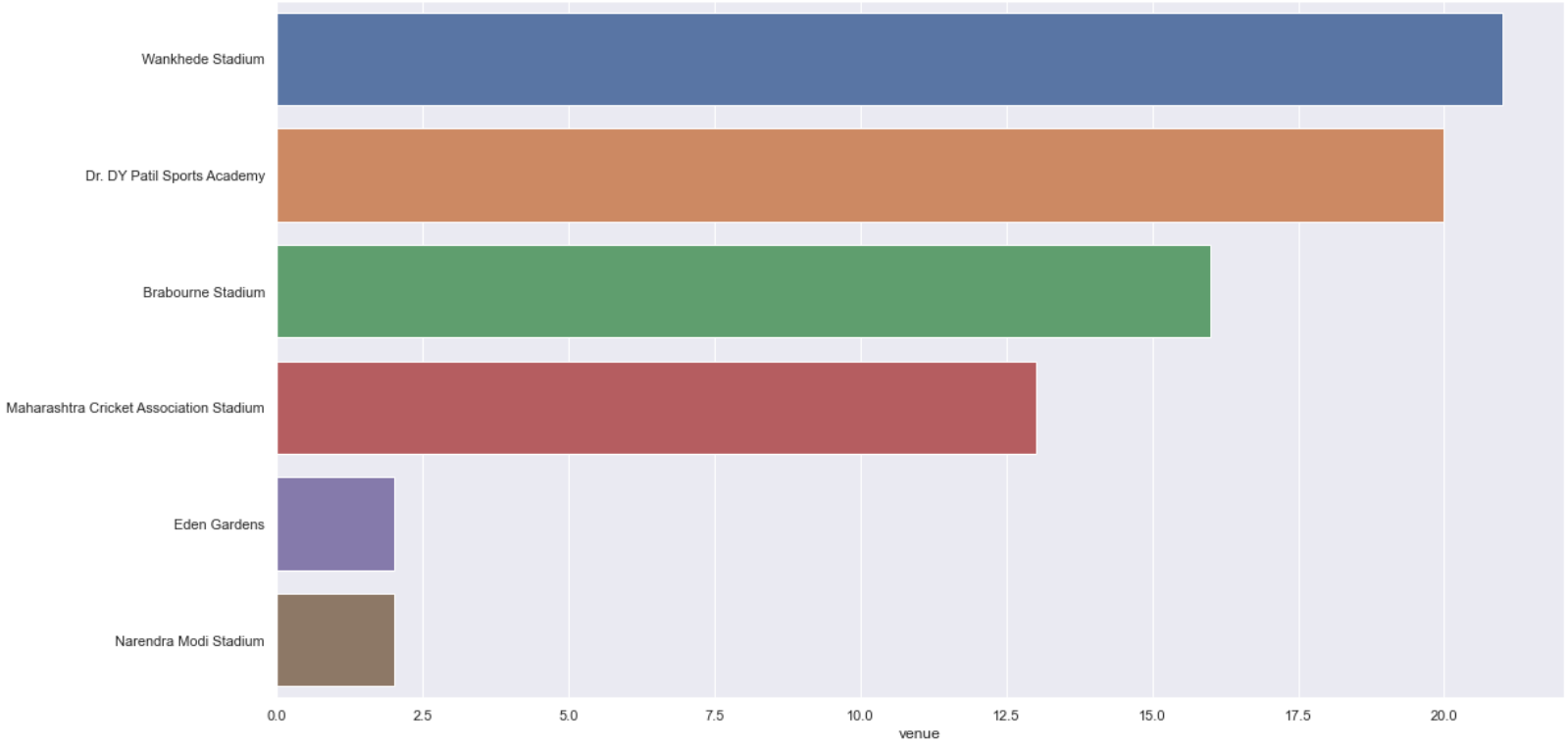
In [29]:
```
ipl.groupby('season')['match_winner'].value_counts()
```

```
season  match_winner
2022    Gujarat Titans                12
        Rajasthan Royals              10
        Lucknow Super Giants           9
        Royal Challengers Bangalore    9
        Delhi Capitals                 7
        Punjab Kings                   7
        Kolkata Knight Riders          6
        Sunrisers Hyderabad            6
        Chennai Super Kings            4
        Mumbai Indians                 4
Name: match_winner, dtype: int64
```

In [30]:
```
#Getting the frequency of venue column
ipl['venue'].value_counts()
```

```
Wankhede Stadium                        21
Dr. DY Patil Sports Academy             20
Brabourne Stadium                       16
Maharashtra Cricket Association Stadium 13
Eden Gardens                             2
Narendra Modi Stadium                    2
Name: venue, dtype: int64
```

In [31]:
```
from matplotlib import rcParams
rcParams['figure.figsize']=18,10
venue = ipl.venue.value_counts()
sns.barplot(y = venue.index, x = venue, orient='h');
```



In [32]:
```
# Task 1
'''
1. New column to record how win was achieved in a match
2. New column to record quantity of win in a match
'''
#adding columns with null values
ipl['win_by_type'] = 0
ipl['win_by_quantity'] = 0
```

In [33]:
```
ipl.head()
```

	timestamp	match_number	season	city	date	team_1	team_2	toss_winner	toss_decision	match_result	dl_appli
0	26-03-2022 23:43	1	2022	Mumbai	26-03-2022	Chennai Super Kings	Kolkata Knight Riders	Kolkata Knight Riders	bowl	normal	no
1	27-03-2022 19:37	2	2022	Mumbai	27-03-2022	Mumbai Indians	Delhi Capitals	Delhi Capitals	bowl	normal	no
2	27-03-2022 23:44	3	2022	Navi Mumbai	27-03-2022	Royal Challengers Bangalore	Punjab Kings	Punjab Kings	bowl	normal	no
3	29-03-2022 00:21	4	2022	Mumbai	29-03-2022	Lucknow Super Giants	Gujarat Titans	Gujarat Titans	bowl	normal	no
4	29-03-2022 23:41	5	2022	Pune	29-03-2022	Rajasthan Royals	Sunrisers Hyderabad	Sunrisers Hyderabad	bowl	normal	no

In [34]:
```
# Looking at the number of rows and columns adding columns with null values in the dataset
ipl.shape
```

```
(74, 21)
```

In [35]:
```
# filling above added columns with null values in the dataset columns with winning type based on existing c
for i in ipl.index:

    if (ipl.win_by_runs[i] != 0):
        ipl.win_by_type[i] = 'Runs'

    elif ((ipl.win_by_runs[i] == 0) & (ipl.win_by_wickets[i] == 0)):
        ipl.win_by_type[i] = 'Super Over'

    else:
        ipl.win_by_type[i] = 'Wickets'
```

```
C:\Users\navna\AppData\Local\Temp/ipykernel_3580/982310820.py:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  ipl.win_by_type[i] = 'Wickets'
C:\Users\navna\anaconda3\Anaconda\Anacondadbda\lib\site-packages\pandas\core\indexing.py:1732: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  self._setitem_single_block(indexer, value, name)
```

In [36]:
```
for i in ipl.index:
        if (ipl.win_by_runs[i] != 0):
            ipl.win_by_quantity[i] = ipl.win_by_runs[i]

        if (ipl.win_by_wickets[i] != 0):
            ipl.win_by_quantity[i] = ipl.win_by_wickets[i]

        if ((ipl.win_by_runs[i] == 0) & (ipl.win_by_wickets[i] == 0)):
            ipl.win_by_quantity[i] = 0
```

```
C:\Users\navna\AppData\Local\Temp/ipykernel_3580/1204894242.py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  ipl.win_by_quantity[i] = ipl.win_by_wickets[i]
C:\Users\navna\AppData\Local\Temp/ipykernel_3580/1204894242.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  ipl.win_by_quantity[i] = ipl.win_by_runs[i]
```

In [37]:
```
ipl.head()
```

	timestamp	match_number	season	city	date	team_1	team_2	toss_winner	toss_decision	match_result	dl_appli
0	26-03-2022 23:43	1	2022	Mumbai	26-03-2022	Chennai Super Kings	Kolkata Knight Riders	Kolkata Knight Riders	bowl	normal	no
1	27-03-2022 19:37	2	2022	Mumbai	27-03-2022	Mumbai Indians	Delhi Capitals	Delhi Capitals	bowl	normal	no
2	27-03-2022 23:44	3	2022	Navi Mumbai	27-03-2022	Royal Challengers Bangalore	Punjab Kings	Punjab Kings	bowl	normal	no
3	29-03-2022 00:21	4	2022	Mumbai	29-03-2022	Lucknow Super Giants	Gujarat Titans	Gujarat Titans	bowl	normal	no
4	29-03-2022 23:41	5	2022	Pune	29-03-2022	Rajasthan Royals	Sunrisers Hyderabad	Sunrisers Hyderabad	bowl	normal	no

In [38]:
```
#Getting the frequency of toss_decision column
ipl['toss_decision'].value_counts()
```

```
bowl    59
bat     15
Name: toss_decision, dtype: int64
```

In [39]:
```
#Getting the frequency of win_by_type column
ipl['win_by_type'].value_counts()
```

```
Wickets    37
Runs       37
Name: win_by_type, dtype: int64
```

In [40]:
```
#Getting the frequency of result column
ipl['match_result'].value_counts()
```

```
normal    74
Name: match_result, dtype: int64
```

In [41]:
```
#Getting the frequency of dl_applied column
ipl['dl_applied'].value_counts()
```

```
no    74
Name: dl_applied, dtype: int64
```

First Inning Insights

In [42]:
```
#Extracting the records where a team won batting first
batting_first=ipl[ipl['win_by_runs']!=0]
```

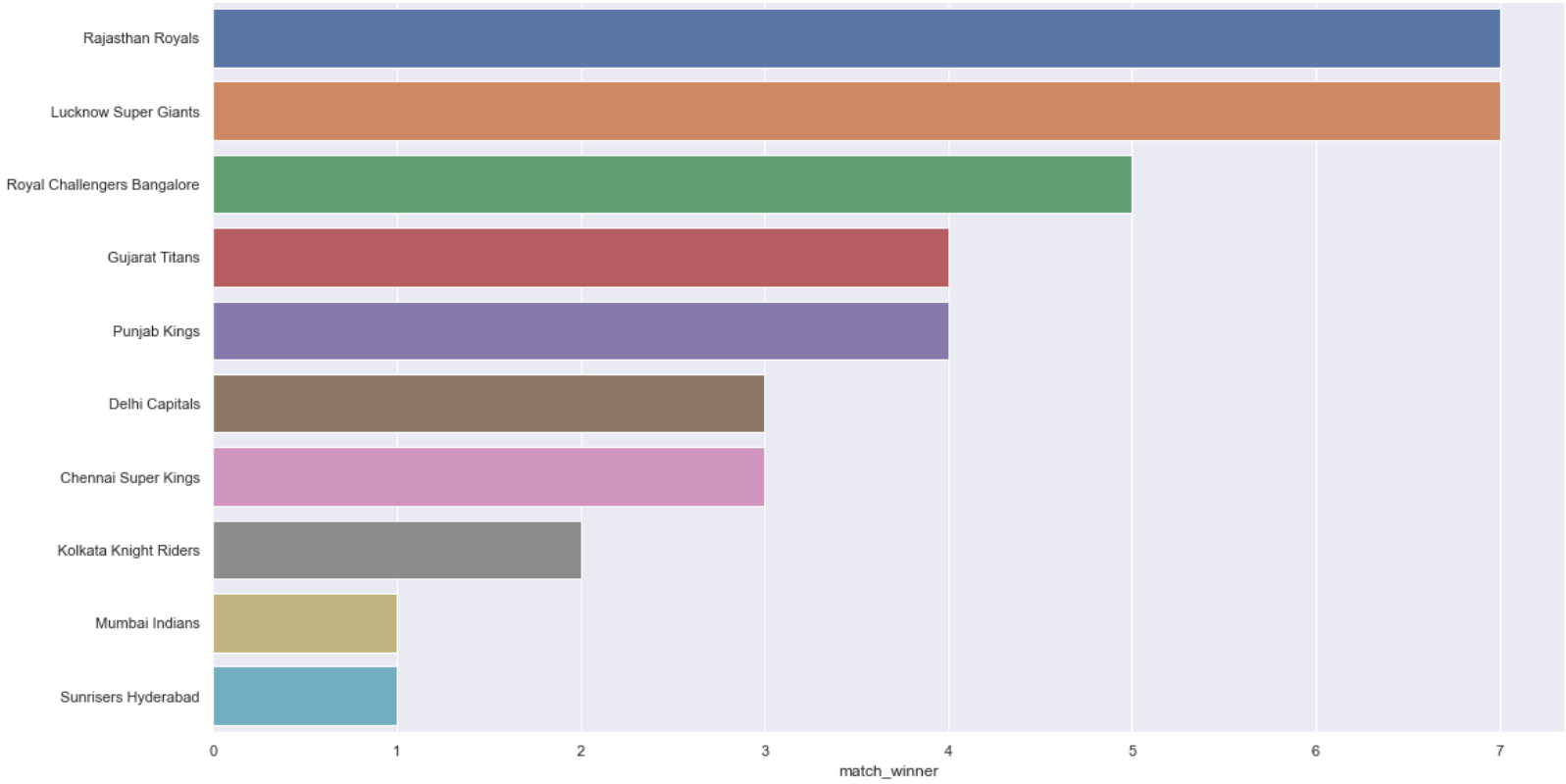
In [43]:
```
batting_first.head()
```

	timestamp	match_number	season	city	date	team_1	team_2	toss_winner	toss_decision	match_result	dl_applie
4	29-03-2022 23:41	5	2022	Pune	29-03-2022	Rajasthan Royals	Sunrisers Hyderabad	Sunrisers Hyderabad	bowl	normal	no
8	02-04-2022 20:49	9	2022	Navi Mumbai	02-04-2022	Rajasthan Royals	Mumbai Indians	Mumbai Indians	bowl	normal	no
9	02-04-2022 23:51	10	2022	Pune	02-04-2022	Gujarat Titans	Delhi Capitals	Delhi Capitals	bowl	normal	no
10	03-04-2022 23:44	11	2022	Mumbai	03-04-2022	Punjab Kings	Chennai Super Kings	Chennai Super Kings	bowl	normal	no
11	08-04-2022 11:20	12	2022	Navi Mumbai	04-04-2022	Lucknow Super Giants	Sunrisers Hyderabad	Sunrisers Hyderabad	bowl	normal	no

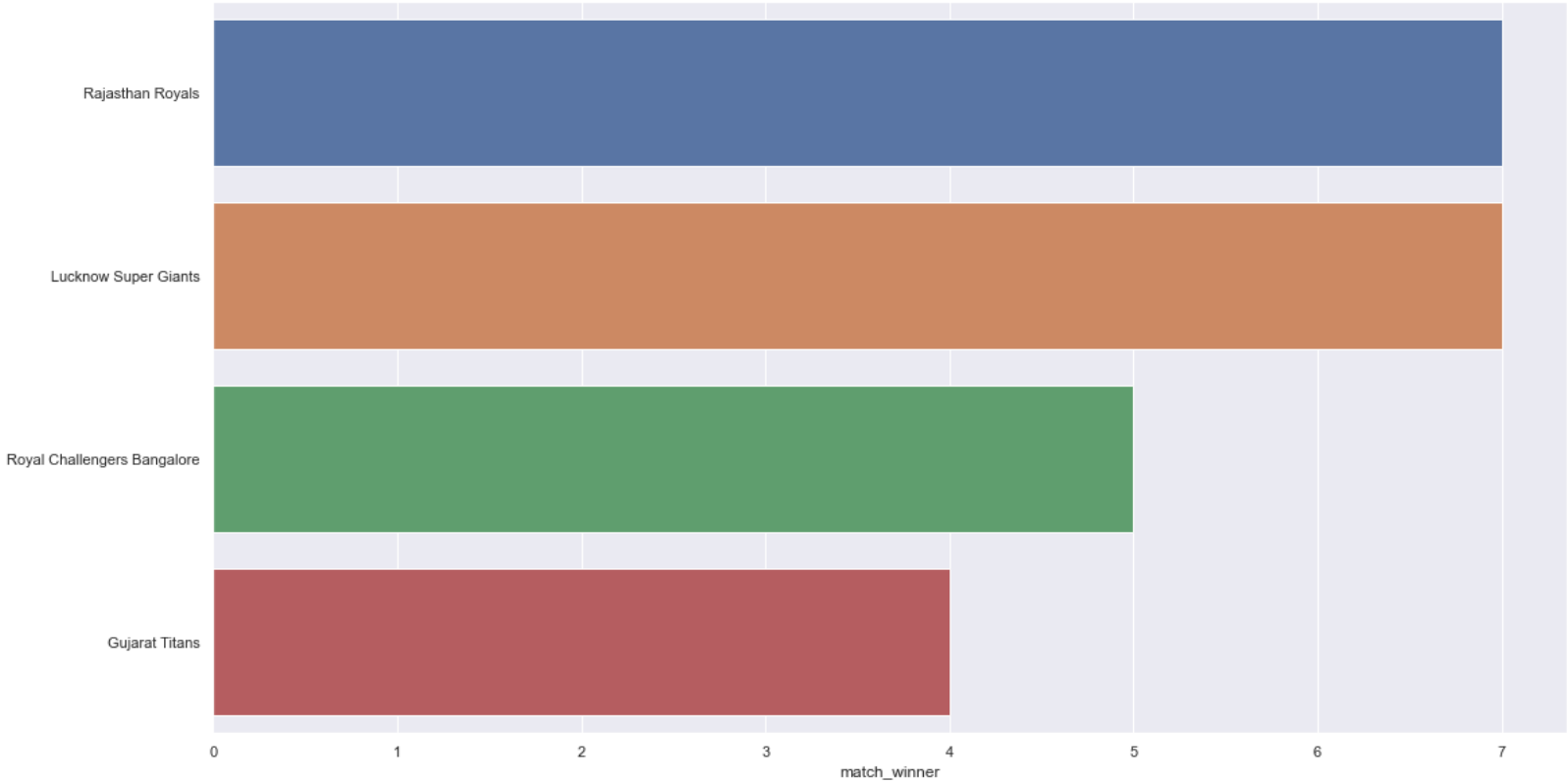
In [44]:
```
#Finding out the number of wins w.r.t each team after batting first
batting_first['match_winner'].value_counts()
```

```
Rajasthan Royals              7
Lucknow Super Giants          7
Royal Challengers Bangalore   5
Gujarat Titans                4
Punjab Kings                  4
Delhi Capitals                3
Chennai Super Kings           3
Kolkata Knight Riders         2
Mumbai Indians                1
Sunrisers Hyderabad           1
Name: match_winner, dtype: int64
```

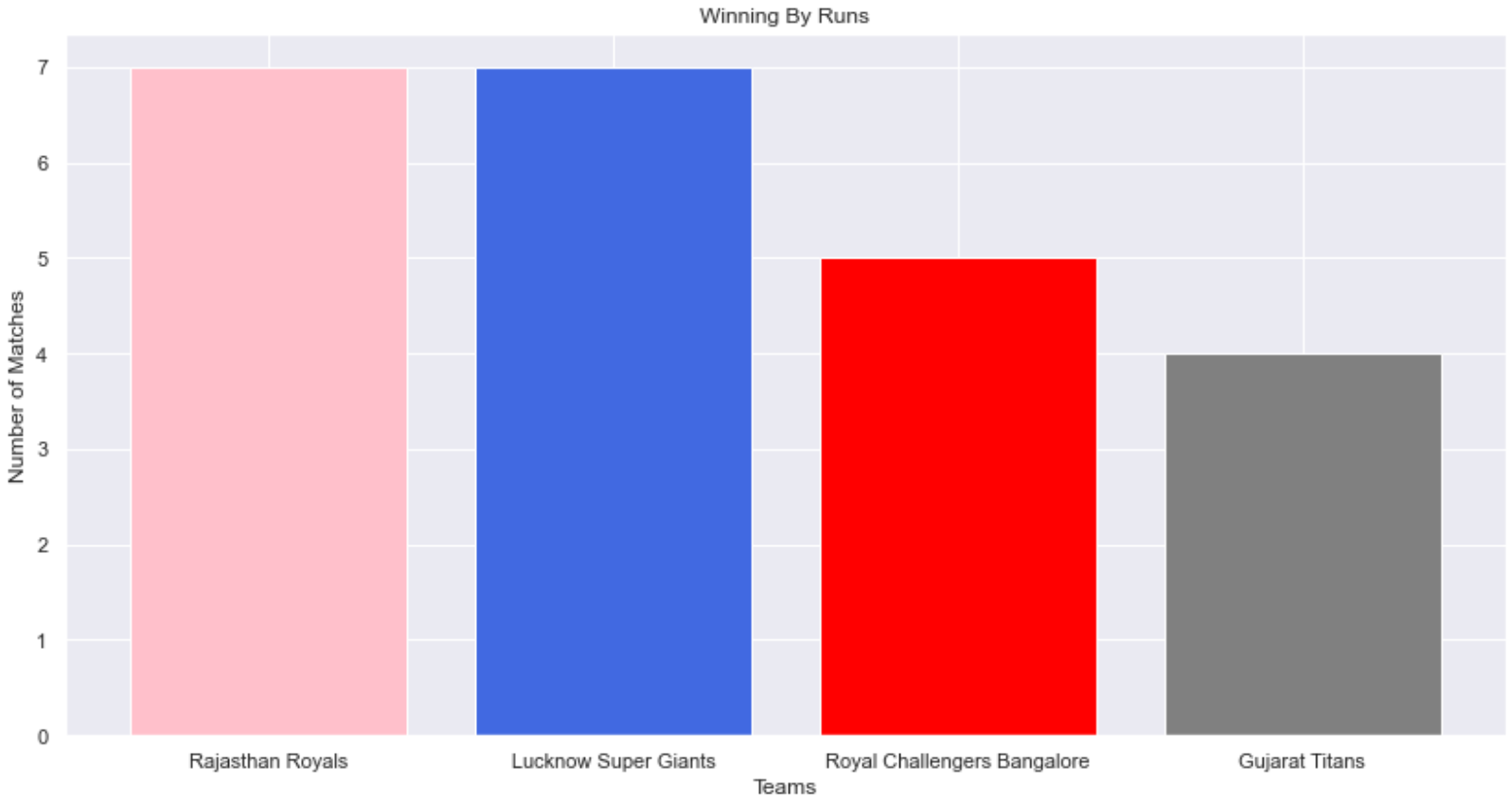
In [45]:
```
batfirst = batting_first['match_winner'].value_counts()
sns.barplot(y = batfirst.index, x = batfirst, orient='h');
```



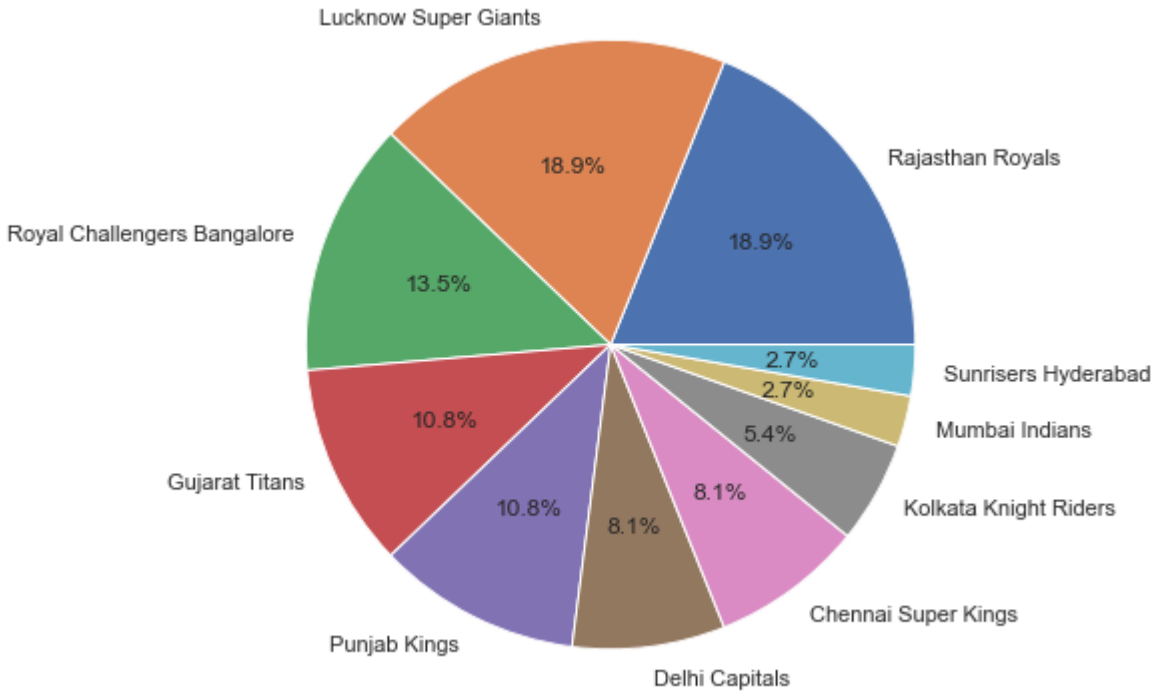
In [46]:
```
batfirst = batting_first['match_winner'].value_counts()[:4]
sns.barplot(y = batfirst.index, x = batfirst, orient='h');
```



In [47]:
```
tting first

0:4].keys()),list(batting_first['match_winner'].value_counts()[0:4]),color=["pink","royalblue","red","grey"])
```

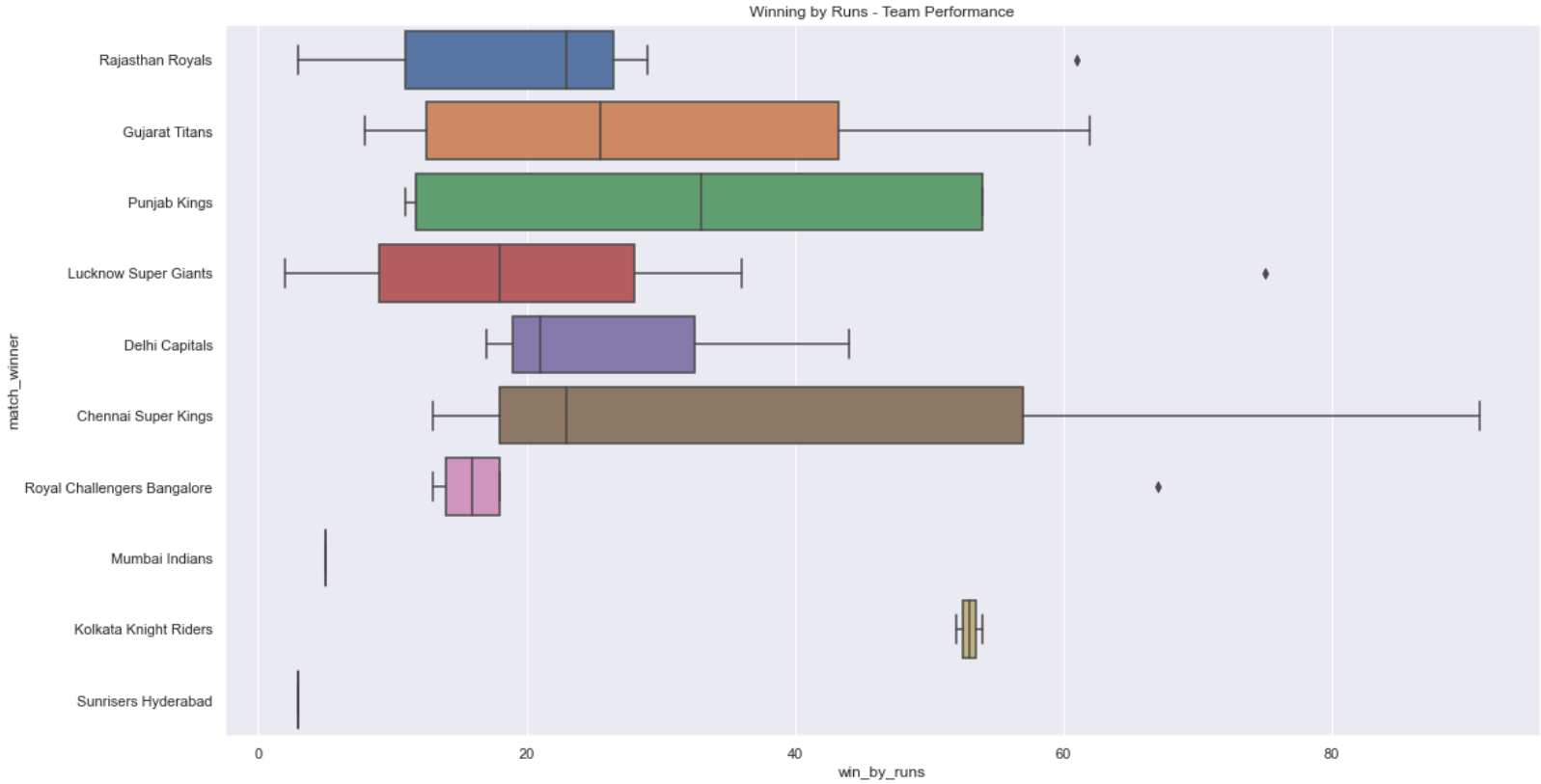


In [48]:
```
#Making a pie  chart
plt.figure(figsize=(7,7))
plt.pie(list(batting_first['match_winner'].value_counts()),labels=list(batting_first['match_winner'].value_
plt.show()
```

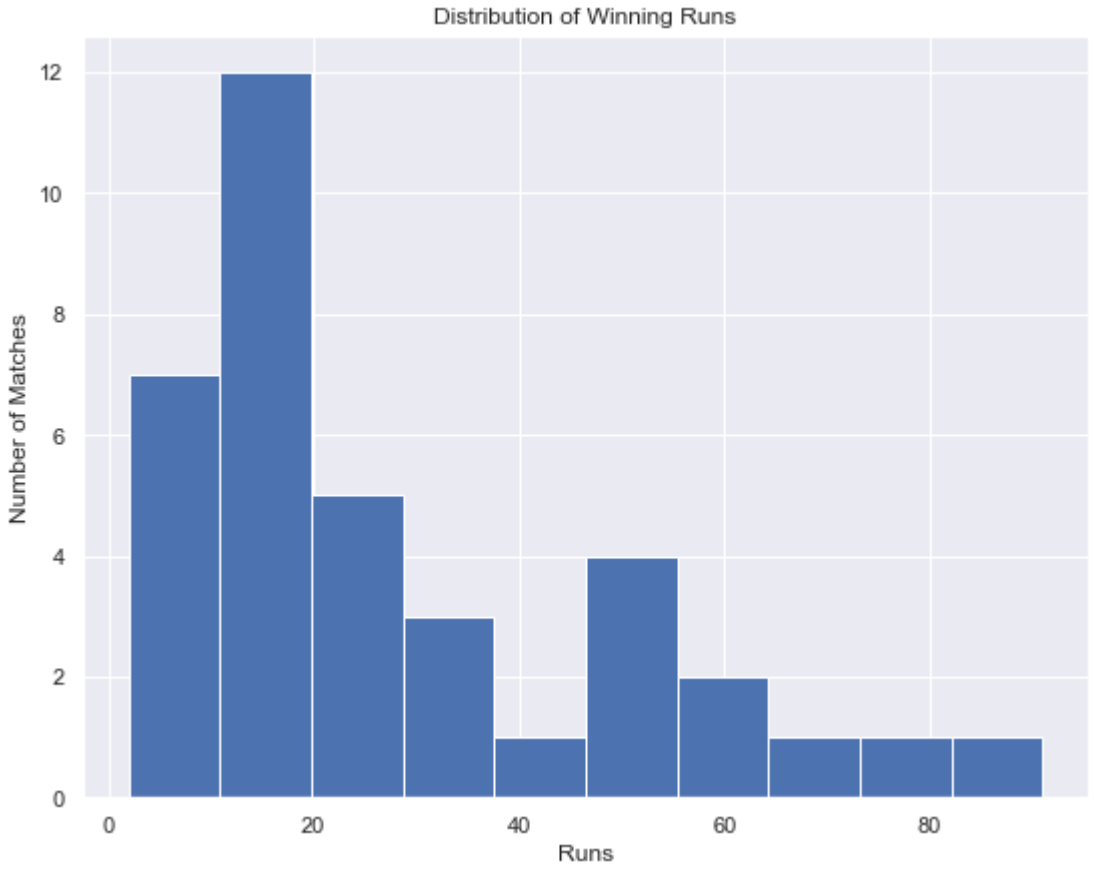


In [49]:

```
#sns.barplot(x="day", y="total_bill", data=tips)
fig, ax = plt.subplots()
#fig.figsize = [16,10]
#ax.set_ylim([0,20])
ax.set_title("Winning by Runs - Team Performance")
#top_players.plot.bar()
sns.boxplot(y = 'match_winner', x = 'win_by_runs', data=ipl[ipl['win_by_runs']>0], orient = 'h'); #palette='
plt.show()
```



In [50]:
```
#Making a histogram
plt.figure(figsize=(9,7))
plt.hist(batting_first['win_by_runs'])
plt.title("Distribution of Winning Runs")
plt.xlabel('Runs')
plt.ylabel('Number of Matches')
plt.show()
```



Which IPL team won by scoring the maximum runs?

In [51]:
```
ipl.iloc[ipl['win_by_runs'].idxmax()]
```

```
timestamp                          22-05-2022 15:56
match_number                                    55
season                                        2022
city                                     Navi Mumbai
date                                      08-05-2022
team_1                          Chennai Super Kings
team_2                               Delhi Capitals
toss_winner                          Delhi Capitals
toss_decision                                  bowl
match_result                                normal
dl_applied                                      no
match_winner                    Chennai Super Kings
win_by_runs                                     91
win_by_wickets                                   0
player_of_the_match                     Devon Conway
venue                  Dr. DY Patil Sports Academy
umpire_1                                Nitin Menon
umpire_2                                Rohan Pandit
umpire_3                               Saiyed Khalid
win_by_type                                     Runs
win_by_quantity                                 91
Name: 54, dtype: object
```

Second Inning Insights

In [52]:
```
#Extracting those records where a team has won after batting second
batting_second=ipl[ipl['win_by_wickets']!=0]
```

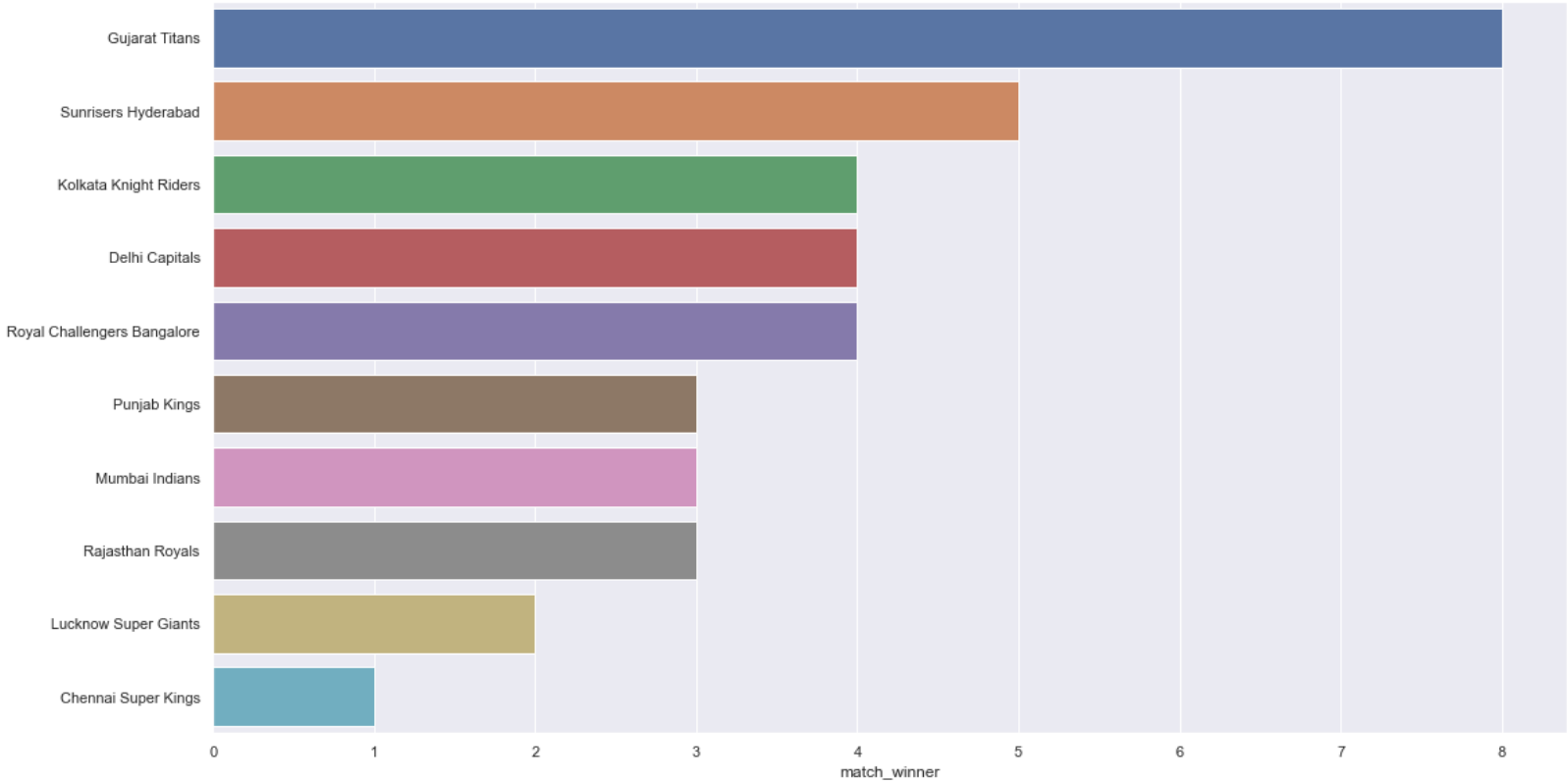
In [53]:
```
#Looking top performance at whose batting second
batting_second.head()
```

	timestamp	match_number	season	city	date	team_1	team_2	toss_winner	toss_decision	match_result	dl_appl
0	26-03-2022 23:43	1	2022	Mumbai	26-03-2022	Chennai Super Kings	Kolkata Knight Riders	Kolkata Knight Riders	bowl	normal	no
1	27-03-2022 19:37	2	2022	Mumbai	27-03-2022	Mumbai Indians	Delhi Capitals	Delhi Capitals	bowl	normal	no
2	27-03-2022 23:44	3	2022	Navi Mumbai	27-03-2022	Royal Challengers Bangalore	Punjab Kings	Punjab Kings	bowl	normal	no
3	29-03-2022 00:21	4	2022	Mumbai	29-03-2022	Lucknow Super Giants	Gujarat Titans	Gujarat Titans	bowl	normal	no
5	31-03-2022 00:30	6	2022	Navi Mumbai	30-03-2022	Kolkata Knight Riders	Royal Challengers Bangalore	Royal Challengers Bangalore	bowl	normal	no

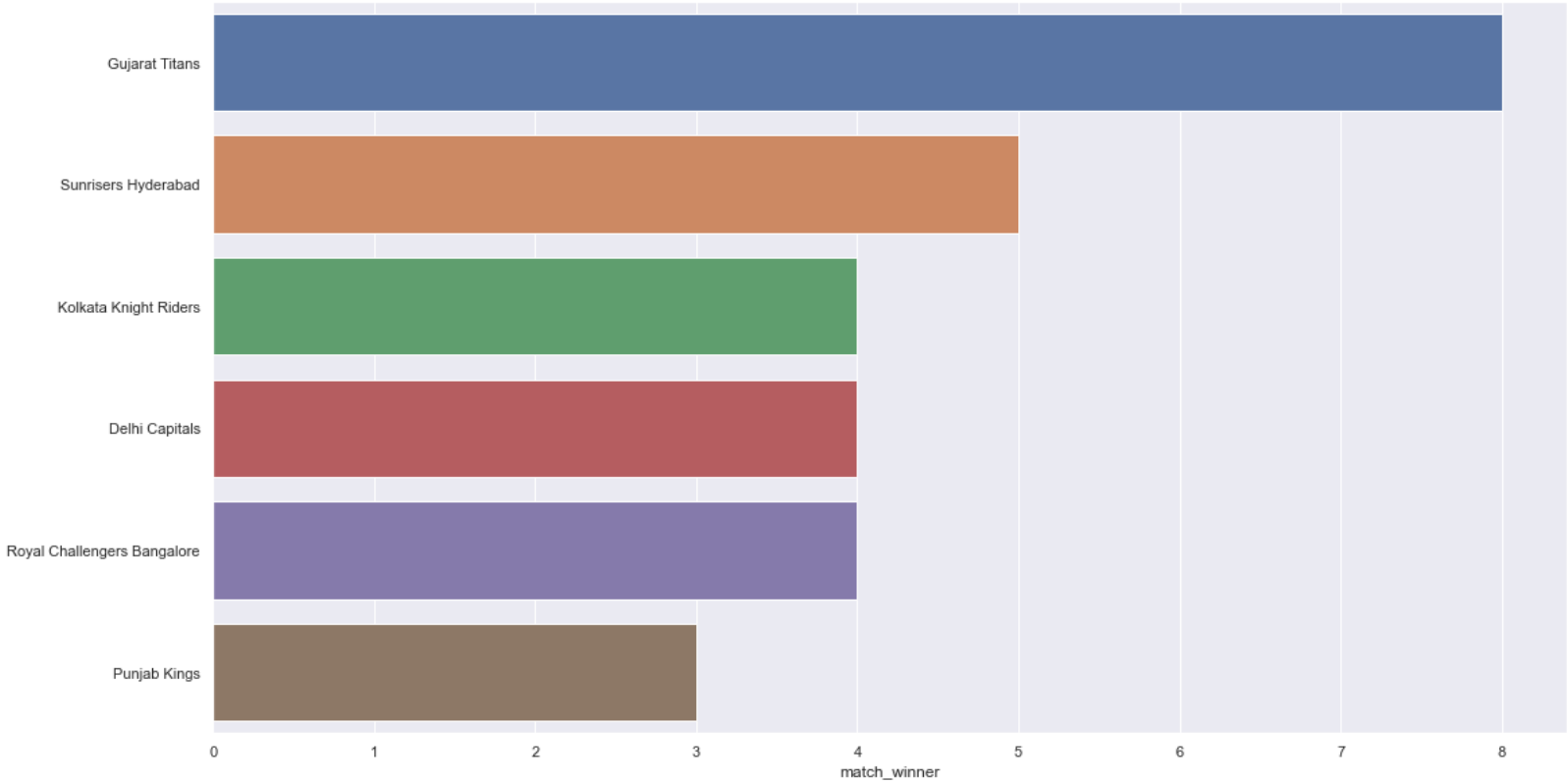
In [54]:
```
#Finding out the number of wins w.r.t each team after batting_second
batting_second['match_winner'].value_counts()
```

```
Gujarat Titans                8
Sunrisers Hyderabad           5
Kolkata Knight Riders         4
Delhi Capitals                4
Royal Challengers Bangalore   4
Punjab Kings                  3
Mumbai Indians                3
Rajasthan Royals              3
Lucknow Super Giants          2
Chennai Super Kings           1
Name: match_winner, dtype: int64
```

In [55]:
```
batsecond = batting_second['match_winner'].value_counts()
sns.barplot(y = batsecond.index, x = batsecond, orient='h');
```

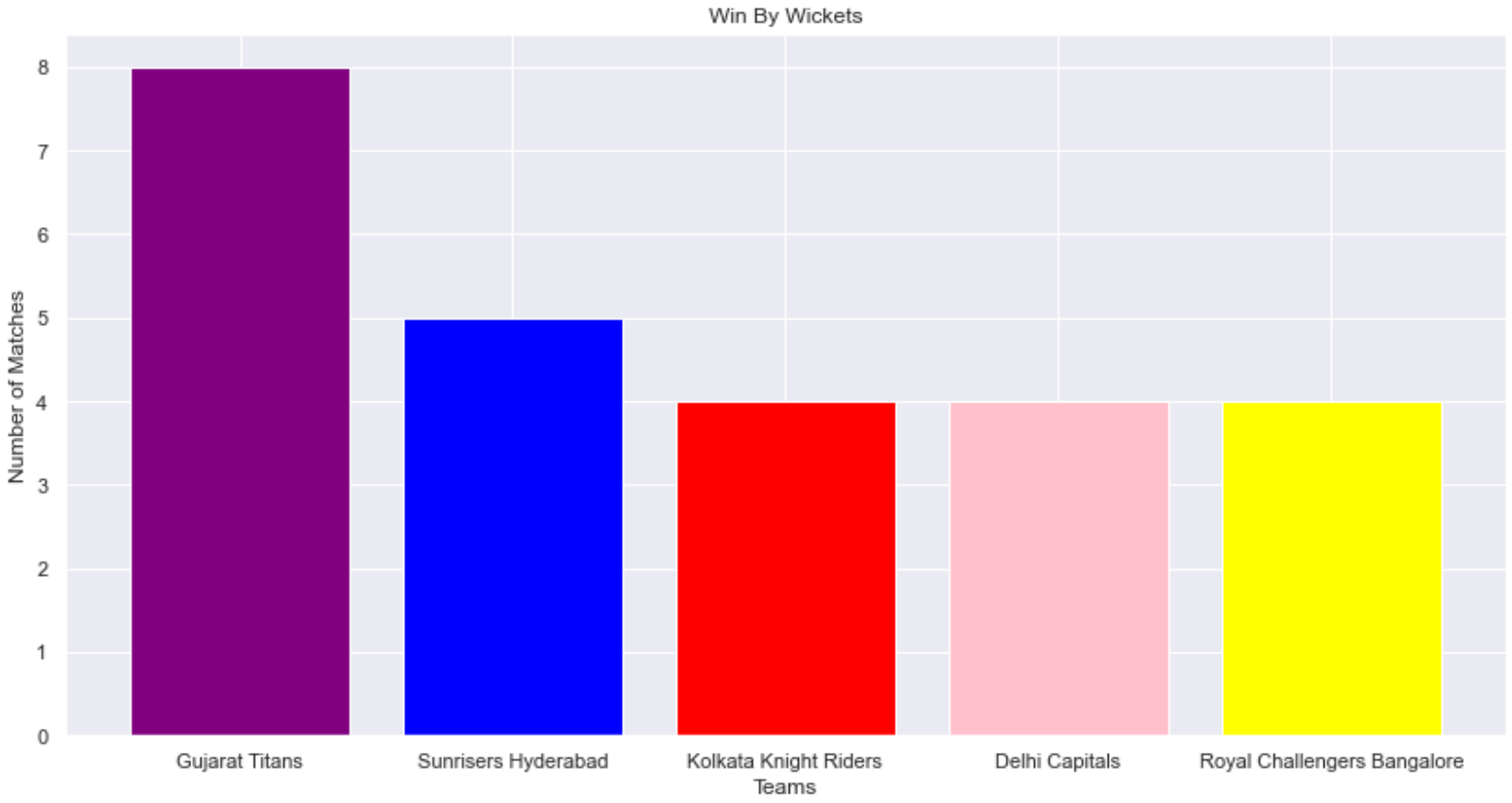


In [56]:
```
batsecond = batting_second['match_winner'].value_counts()[:6]
sns.barplot(y = batsecond.index, x = batsecond, orient='h');
```

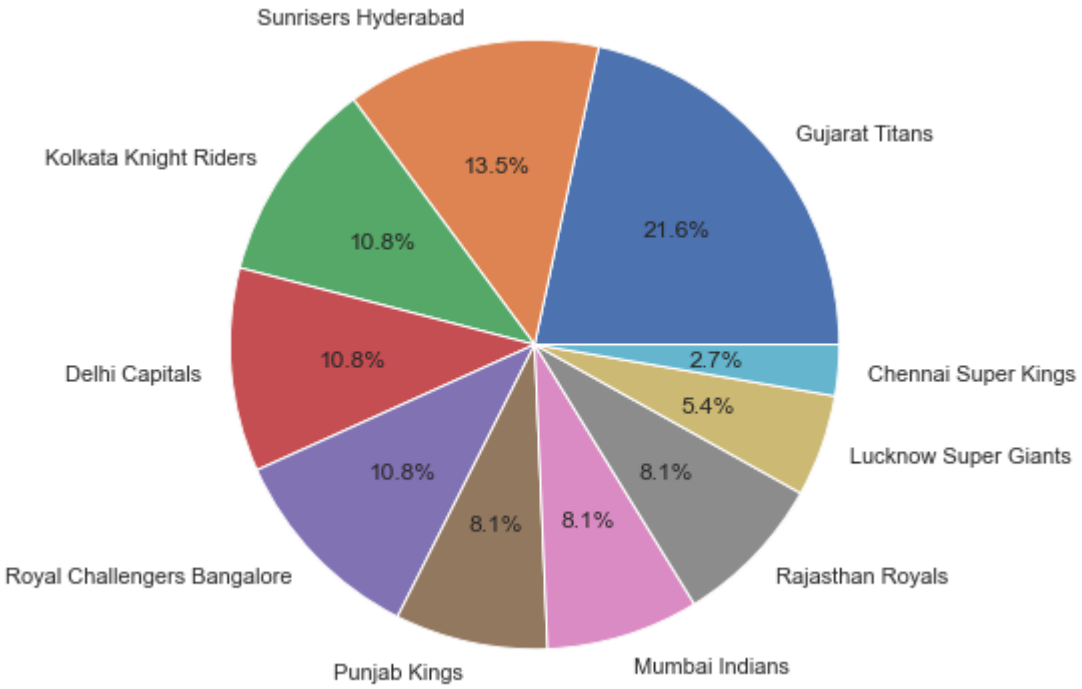


In [57]:

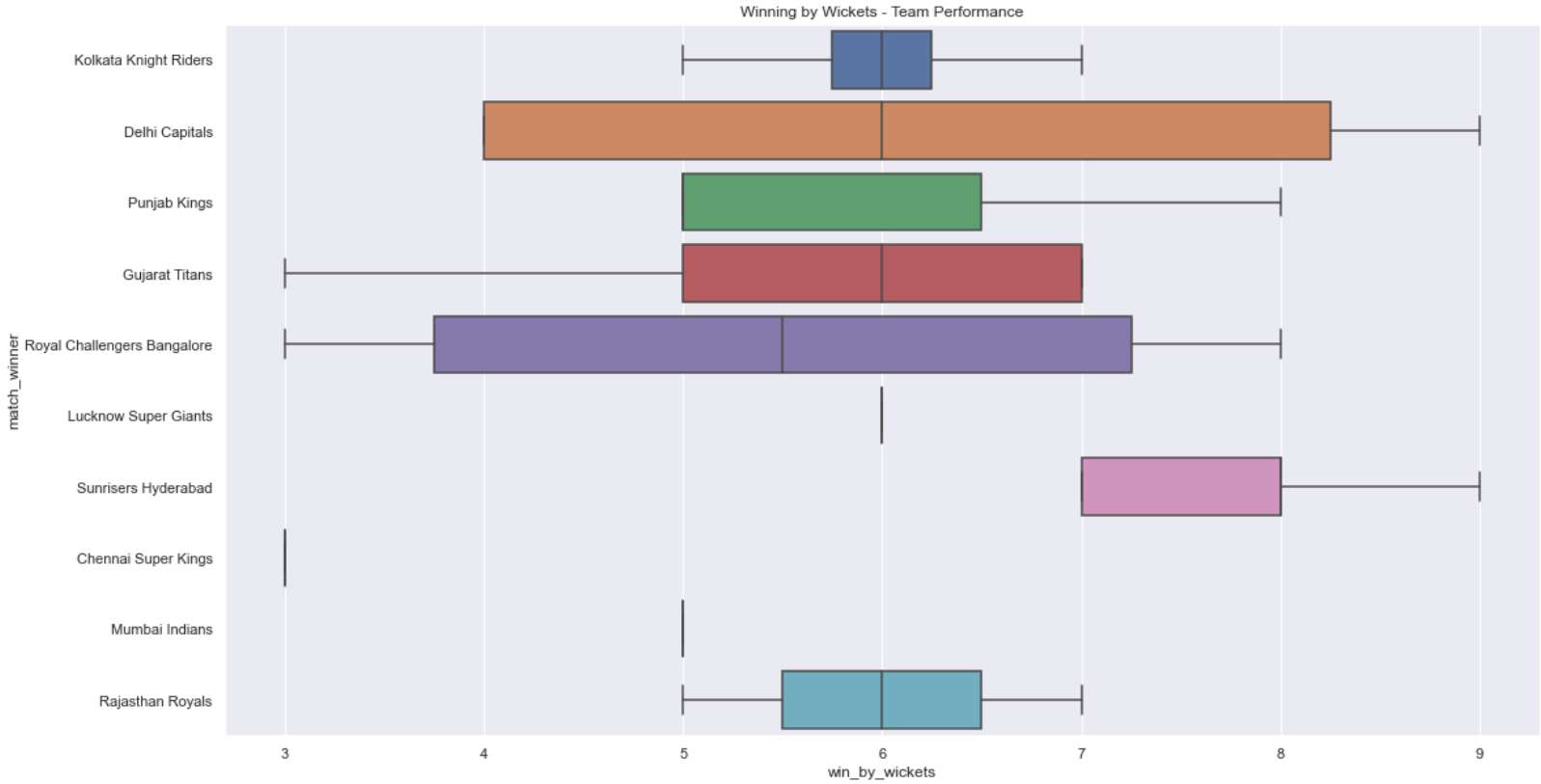
```
#Making a bar-plot for top 5 teams with most wins after batting_second
plt.figure(figsize=(14,7))
plt.bar(list(batting_second['match_winner'].value_counts()[0:5].keys()),list(batting_second['match_winner']
plt.title("Win By Wickets")
plt.xlabel('Teams')
plt.ylabel('Number of Matches')
plt.show()
```



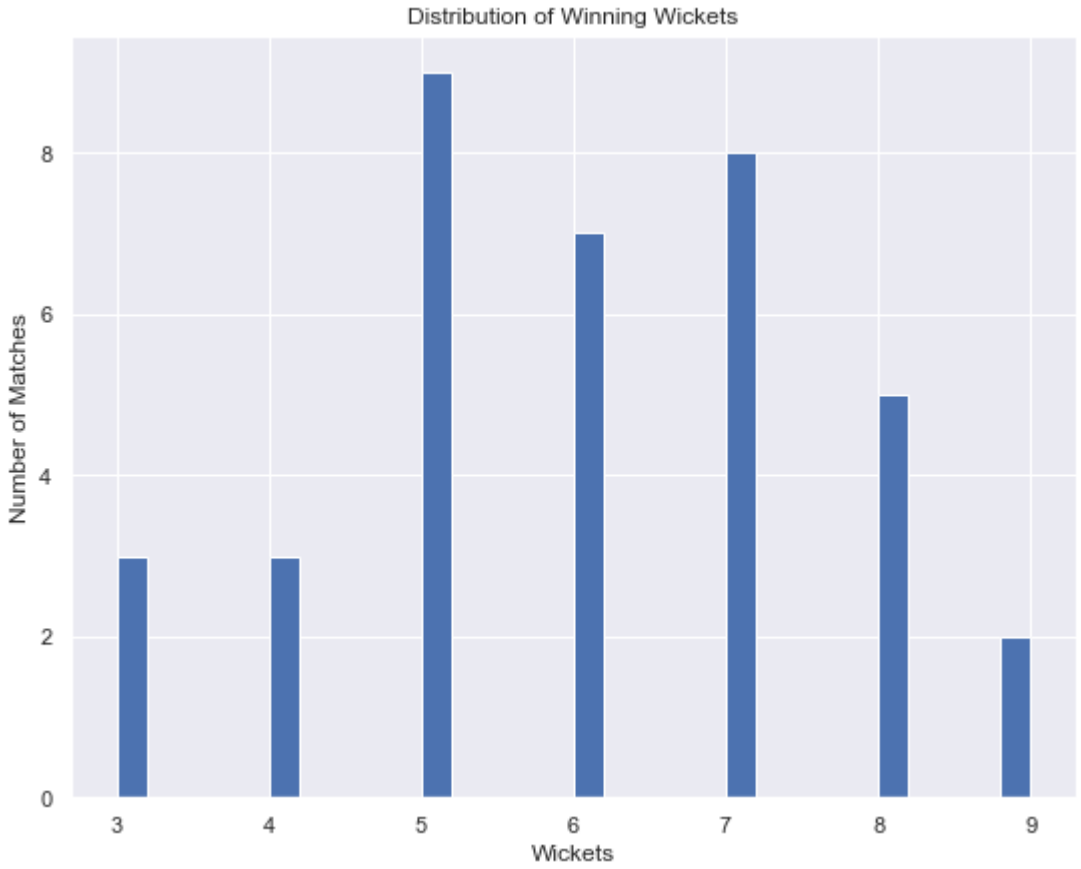
In [58]:
```
#Making a pie  chart
plt.figure(figsize=(7,7))
plt.pie(list(batting_second['match_winner'].value_counts()),labels=list(batting_second['match_winner'].valu
plt.show()
```



In [59]:
```
#sns.barplot(x="day", y="total_bill", data=tips)
fig, ax = plt.subplots()
#fig.figsize = [16,10]
#ax.set_ylim([0,20])
ax.set_title("Winning by Wickets - Team Performance")
#top_players.plot.bar()
sns.boxplot(y = 'match_winner', x = 'win_by_wickets', data=ipl[ipl['win_by_wickets']>0], orient = 'h'); #pa
plt.show()
```



In [60]:
```
#Making a histogram
plt.figure(figsize=(9,7))
plt.hist(batting_second['win_by_wickets'],bins=30)
plt.title("Distribution of Winning Wickets")
plt.xlabel('Wickets')
plt.ylabel('Number of Matches')
plt.show()
```



Which IPL team won by consuming maximum wickets?

In [61]:
```
ipl.iloc[ipl['win_by_wickets'].idxmax()]
```

```
timestamp                 22-05-2022 15:10
match_number                            32
season                                2022
city                                Mumbai
date                            20-04-2022
team_1                        Punjab Kings
team_2                      Delhi Capitals
toss_winner                 Delhi Capitals
toss_decision                         bowl
match_result                        normal
dl_applied                              no
match_winner                Delhi Capitals
win_by_runs                              0
win_by_wickets                           9
player_of_the_match          Kuldeep Yadav
venue                      Brabourne Stadium
umpire_1                       Tapan Sharma
umpire_2                         Rod Tucker
umpire_3               Anil Kumar Chaudhary
win_by_type                         Wickets
win_by_quantity                          9
Name: 31, dtype: object
```

Which Team had won by keeping minimum wickets in hand?

In [62]:
```
ipl.iloc[ipl[ipl['win_by_wickets'].ge(1)].win_by_wickets.idxmin()]
```

```
timestamp                         31-03-2022 00:30
match_number                                     6
season                                        2022
city                                     Navi Mumbai
date                                      30-03-2022
team_1                         Kolkata Knight Riders
team_2                  Royal Challengers Bangalore
toss_winner             Royal Challengers Bangalore
toss_decision                                  bowl
match_result                                normal
dl_applied                                      no
match_winner            Royal Challengers Bangalore
win_by_runs                                      0
win_by_wickets                                   3
player_of_the_match                Wanindu Hasaranga
venue                   Dr. DY Patil Sports Academy
umpire_1                      Jayaraman Madanagopal
umpire_2                              Navdeep Singh
umpire_3                                Nitin Menon
win_by_type                                 Wickets
win_by_quantity                                  3
Name: 5, dtype: object
```

What is the probability of winning a match if the toss was won?

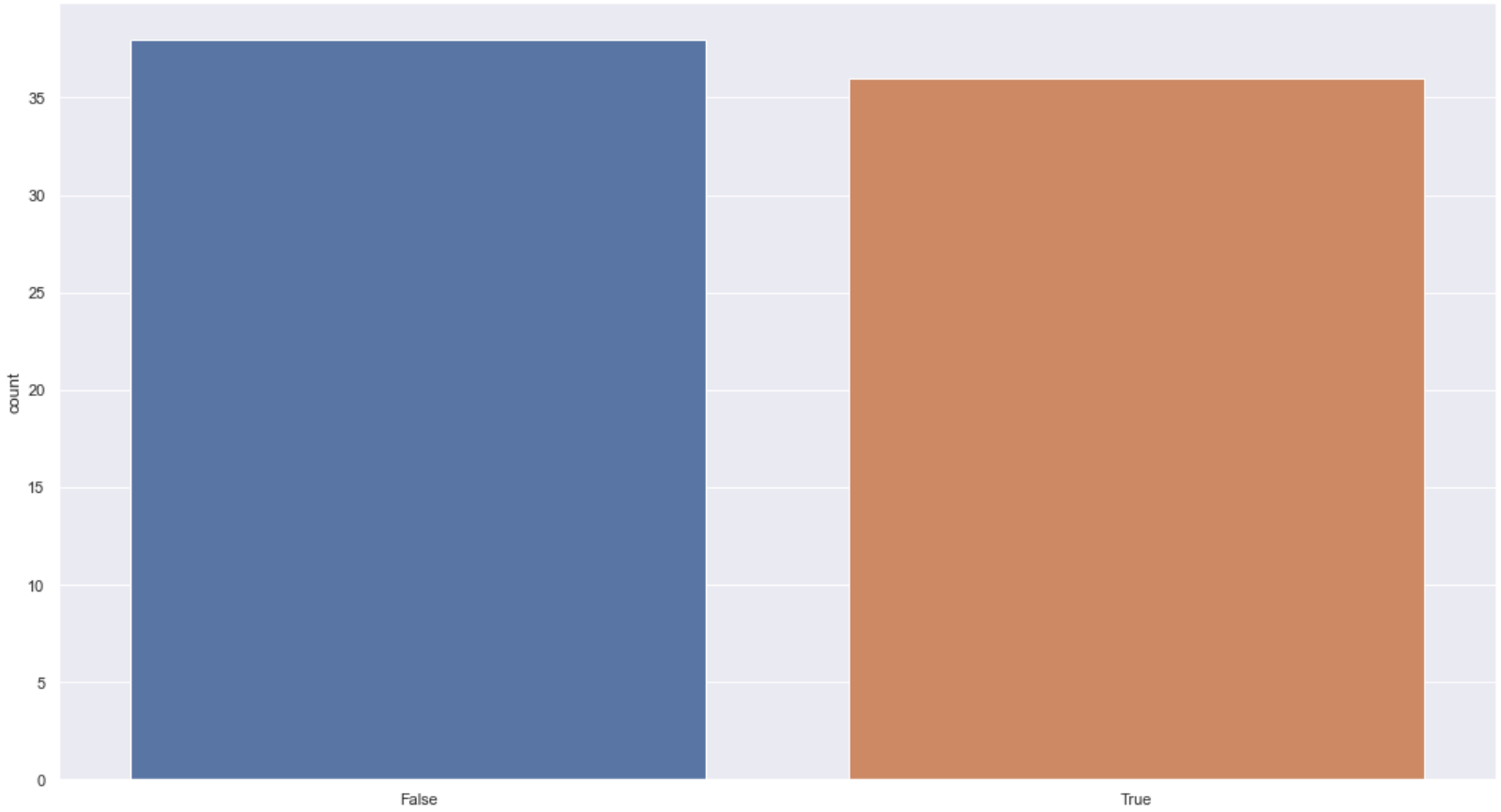
In [63]:
```
probability_of_win = ipl['toss_winner'] == ipl['match_winner']

probability_of_win.groupby(probability_of_win).size()
```

```
False    38
True     36
dtype: int64
```

In [64]:
```
sns.countplot(probability_of_win)
```

```
C:\Users\navna\anaconda3\Anaconda\Anacondadbda\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as
a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:ylabel='count'>
```



In [65]:
```
#Findind out how many times a team has won the match after winning the toss
import numpy as np
np.sum(ipl['toss_winner']==ipl['match_winner'])
```

```
36
```

In [68]:
```
percent=(36/74)*100
percent
```

```
48.64864864864865
```

In [69]:
```
print("The probability of winning a match if the toss was won is 48.65 %")
```

```
The probability of winning a match if the toss was won is 48.65 %
```

In [ ]: