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

In [13]:
```
matches_df = pd.read_csv("matches.csv")
score_df = pd.read_csv("deliveries.csv.zip")
```

In [14]:
```
matches_df.head()
```

Out[14]:

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

In [15]:
```
score_df.head()
```

Out[15]:

	match_id	inning	batting_team	bowling_team	over	ball	batsman	non_striker	bowler	is_supe
0	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	1	DA Warner	S Dhawan	TS Mills	
1	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	2	DA Warner	S Dhawan	TS Mills	
2	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	3	DA Warner	S Dhawan	TS Mills	
3	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	4	DA Warner	S Dhawan	TS Mills	
4	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	5	DA Warner	S Dhawan	TS Mills	

5 rows × 21 columns

DATA INFORMATION

In [16]:
```
print(matches_df.info())
print(score_df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 636 entries, 0 to 635
Data columns (total 18 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   id               636 non-null    int64  
 1   season           636 non-null    int64  
 2   city             629 non-null    object 
 3   date             636 non-null    object 
 4   team1            636 non-null    object 
 5   team2            636 non-null    object 
 6   toss_winner      636 non-null    object 
 7   toss_decision    636 non-null    object 
 8   result           636 non-null    object 
 9   dl_applied       636 non-null    int64  
 10  winner           633 non-null    object 
 11  win_by_runs      636 non-null    int64  
 12  win_by_wickets   636 non-null    int64  
 13  player_of_match  633 non-null    object 
 14  venue            636 non-null    object 
 15  umpire1          635 non-null    object 
 16  umpire2          635 non-null    object 
 17  umpire3          0 non-null      float64
dtypes: float64(1), int64(5), object(12)
memory usage: 89.6+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150460 entries, 0 to 150459
Data columns (total 21 columns):
 #   Column            Non-Null Count   Dtype 
---  ------            --------------   ----- 
 0   match_id          150460 non-null  int64 
 1   inning            150460 non-null  int64 
 2   batting_team      150460 non-null  object
 3   bowling_team      150460 non-null  object
 4   over              150460 non-null  int64 
 5   ball              150460 non-null  int64 
 6   batsman           150460 non-null  object
 7   non_striker       150460 non-null  object
 8   bowler            150460 non-null  object
 9   is_super_over     150460 non-null  int64 
 10  wide_runs         150460 non-null  int64 
 11  bye_runs          150460 non-null  int64 
 12  legbye_runs       150460 non-null  int64 
 13  noball_runs       150460 non-null  int64 
 14  penalty_runs      150460 non-null  int64 
 15  batsman_runs      150460 non-null  int64 
 16  extra_runs        150460 non-null  int64 
 17  total_runs        150460 non-null  int64 
 18  player_dismissed  7438 non-null    object
 19  dismissal_kind    7438 non-null    object
 20  fielder           5369 non-null    object
dtypes: int64(13), object(8)
memory usage: 24.1+ MB
None
```

In [18]:
```
matches_df["umpire3"].isnull().sum()
```

Out[18]:
```
636
```

In [19]:
```
matches_df["umpire3"].tail(10)
```

Out[19]:
```
626   NaN
627   NaN
628   NaN
629   NaN
630   NaN
631   NaN
632   NaN
633   NaN
634   NaN
635   NaN
Name: umpire3, dtype: float64
```

In [20]:
```
matches_df.describe()
```

Out[20]:

	id	season	dl_applied	win_by_runs	win_by_wickets	umpire3
count	636.000000	636.000000	636.000000	636.000000	636.000000	0.0
mean	318.500000	2012.490566	0.025157	13.682390	3.372642	NaN
std	183.741666	2.773026	0.156726	23.908877	3.420338	NaN
min	1.000000	2008.000000	0.000000	0.000000	0.000000	NaN
25%	159.750000	2010.000000	0.000000	0.000000	0.000000	NaN
50%	318.500000	2012.000000	0.000000	0.000000	4.000000	NaN
75%	477.250000	2015.000000	0.000000	20.000000	7.000000	NaN
max	636.000000	2017.000000	1.000000	146.000000	10.000000	NaN

In [21]:
```
# Matches we have got in the dataset
matches_df['id'].max()
```

Out[21]:
```
636
```

In [22]:
```
# Seasons we have got in the dataset
matches_df['season'].unique()
```

Out[22]:
```
array([2017, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016],
      dtype=int64)
```

Team won by Maximum Runs

In [23]:
```
matches_df.iloc[matches_df['win_by_runs'].idxmax()]
```

Out[23]:
```
id                                44
season                          2017
city                           Delhi
date                      2017-05-06
team1                 Mumbai Indians
team2               Delhi Daredevils
toss_winner         Delhi Daredevils
toss_decision                  field
result                        normal
dl_applied                         0
winner                Mumbai Indians
win_by_runs                      146
win_by_wickets                     0
player_of_match          LMP Simmons
venue               Feroz Shah Kotla
umpire1                  Nitin Menon
umpire2                    CK Nandan
umpire3                          NaN
Name: 43, dtype: object
```

In [24]:
```
matches_df.iloc[matches_df['win_by_runs'].idxmax()]['winner']
```

Out[24]:
```
'Mumbai Indians'
```

Team won by Maximum Wickets

In [25]:
```
matches_df.iloc[matches_df['win_by_wickets'].idxmax()]['winner']
```

Out[25]:
```
'Kolkata Knight Riders'
```

Team won by minimum runs

In [26]:
```
matches_df.iloc[matches_df[matches_df['win_by_runs'].ge(1)].win_by_runs.idxmin()
```

Out[26]:
```
'Mumbai Indians'
```

In [27]:
```
matches_df.iloc[matches_df[matches_df['win_by_wickets'].ge(1)].win_by_wickets.id
```

Out[27]:
```
id                                560
season                           2015
city                          Kolkata
date                       2015-05-09
team1                 Kings XI Punjab
team2           Kolkata Knight Riders
toss_winner           Kings XI Punjab
toss_decision                     bat
result                         normal
dl_applied                          0
winner          Kolkata Knight Riders
win_by_runs                         0
win_by_wickets                      1
player_of_match            AD Russell
venue                    Eden Gardens
umpire1                  AK Chaudhary
umpire2               HDPK Dharmasena
umpire3                           NaN
Name: 559, dtype: object
```

In [28]:
```
matches_df.iloc[matches_df[matches_df['win_by_wickets'].ge(1)].win_by_wickets.id
```

Out[28]:
```
'Kolkata Knight Riders'
```

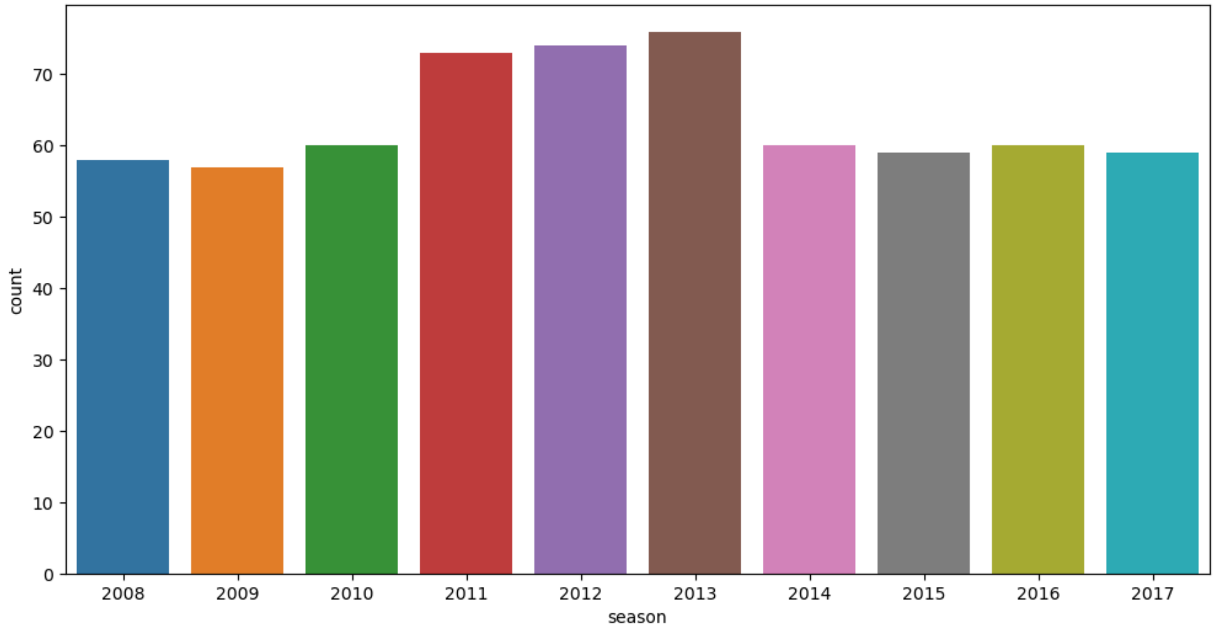
Observation :

1)Mumbai Indians is the team which won by maximum and minimum runs

2)Kolkata Knight Riders is the team which won by maximum and minimum wickets

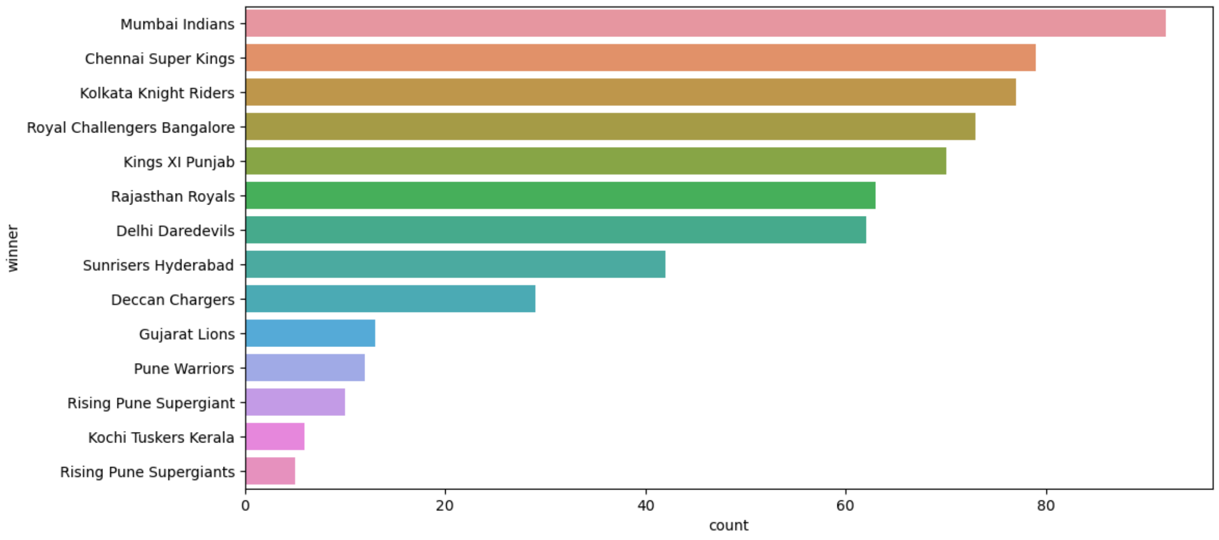
Season Which had most number of matches

In [29]:
```
plt.figure(figsize=(12,6))
sns.countplot(x='season', data=matches_df)
plt.show()
```



In 2013, we have the most number of matches

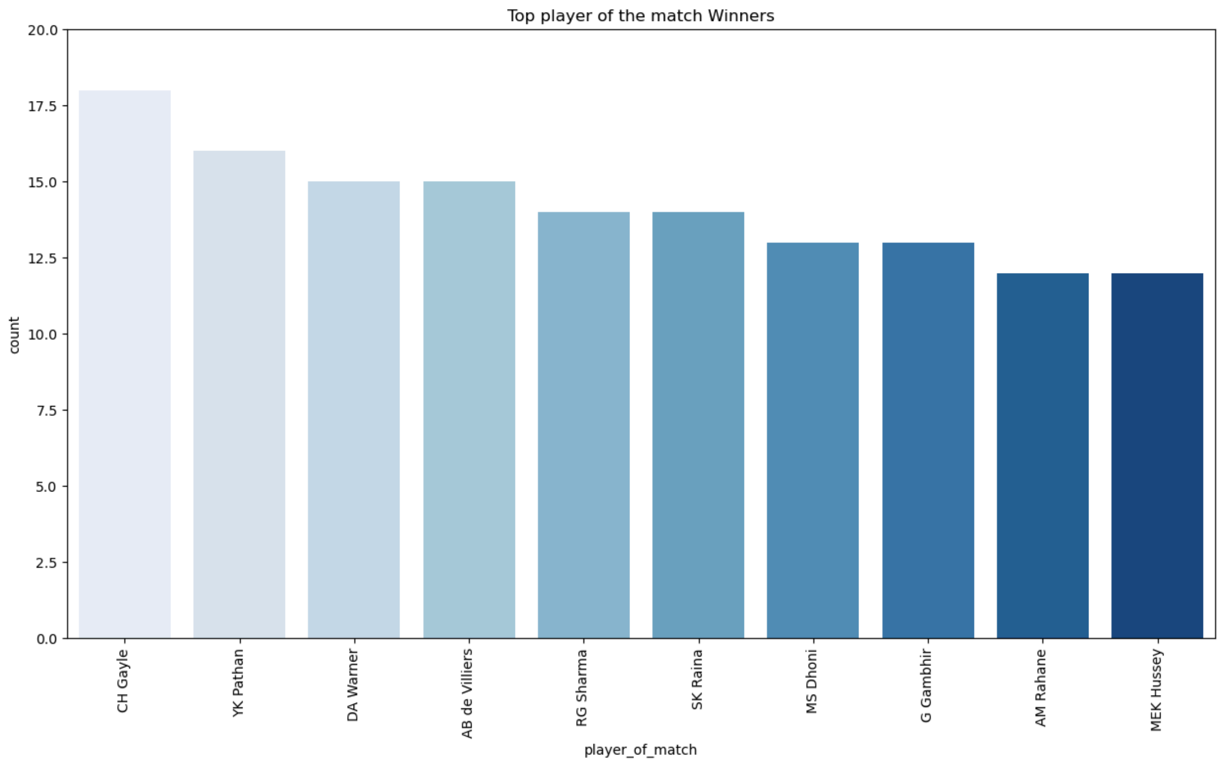
In [30]:
```
plt.figure(figsize=(12,6))
data = matches_df.winner.value_counts()
sns.barplot(y = data.index, x = data, orient='h')
plt.show()
```



Mumbai Indians are the winners in most of the matches

Top Player of the match winners¶

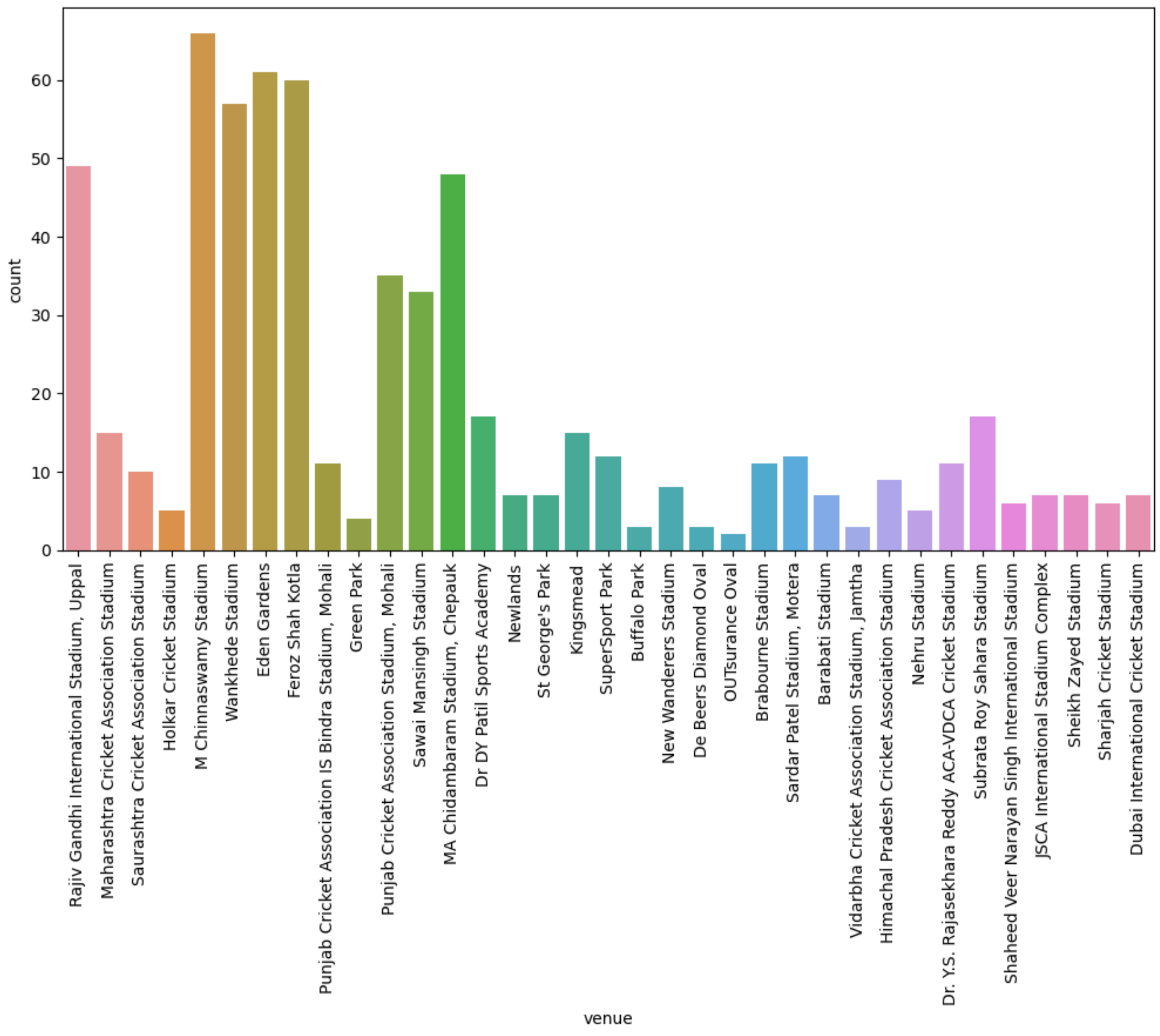
In [31]:
```
top_players = matches_df.player_of_match.value_counts()[:10]
#sns.barplot(x="day", y="total_bill", data=df)
fig, ax = plt.subplots(figsize=(15,8))
ax.set_ylim([0,20])
ax.set_ylabel("Count")
ax.set_title("Top player of the match Winners")
top_players.plot.bar()
sns.barplot(x = top_players.index, y = top_players, orient='v', palette="Blues")
plt.show()
```



CH Gayle is the most Successful player in all match winners

Number of matches in each venue:

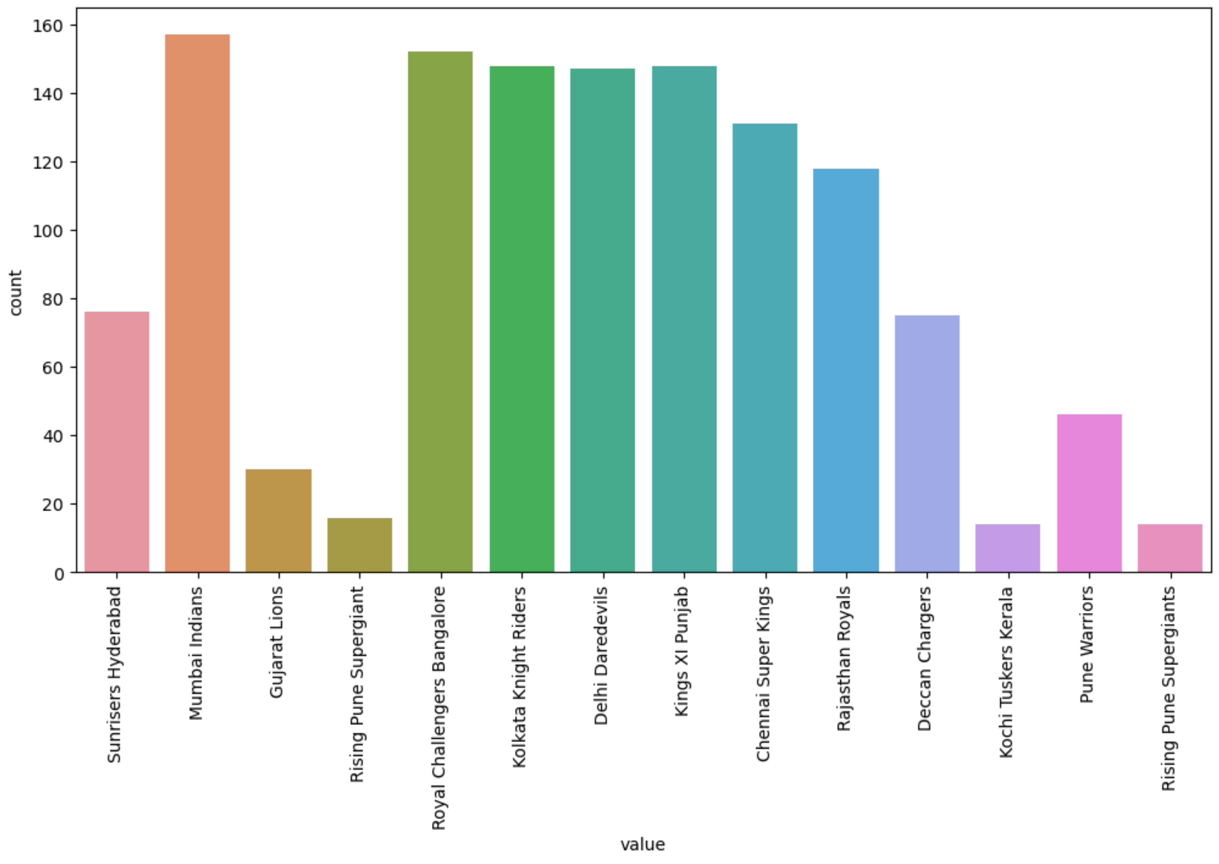
In [32]:
```
plt.figure(figsize=(12,6))
sns.countplot(x='venue', data=matches_df)
plt.xticks(rotation='vertical')
plt.show()
```



There are quite a few venues present in the data with "M Chinnaswamy Stadium" being the one with most number of matches followed by "Eden Gardens"

Number of matches played by each team:

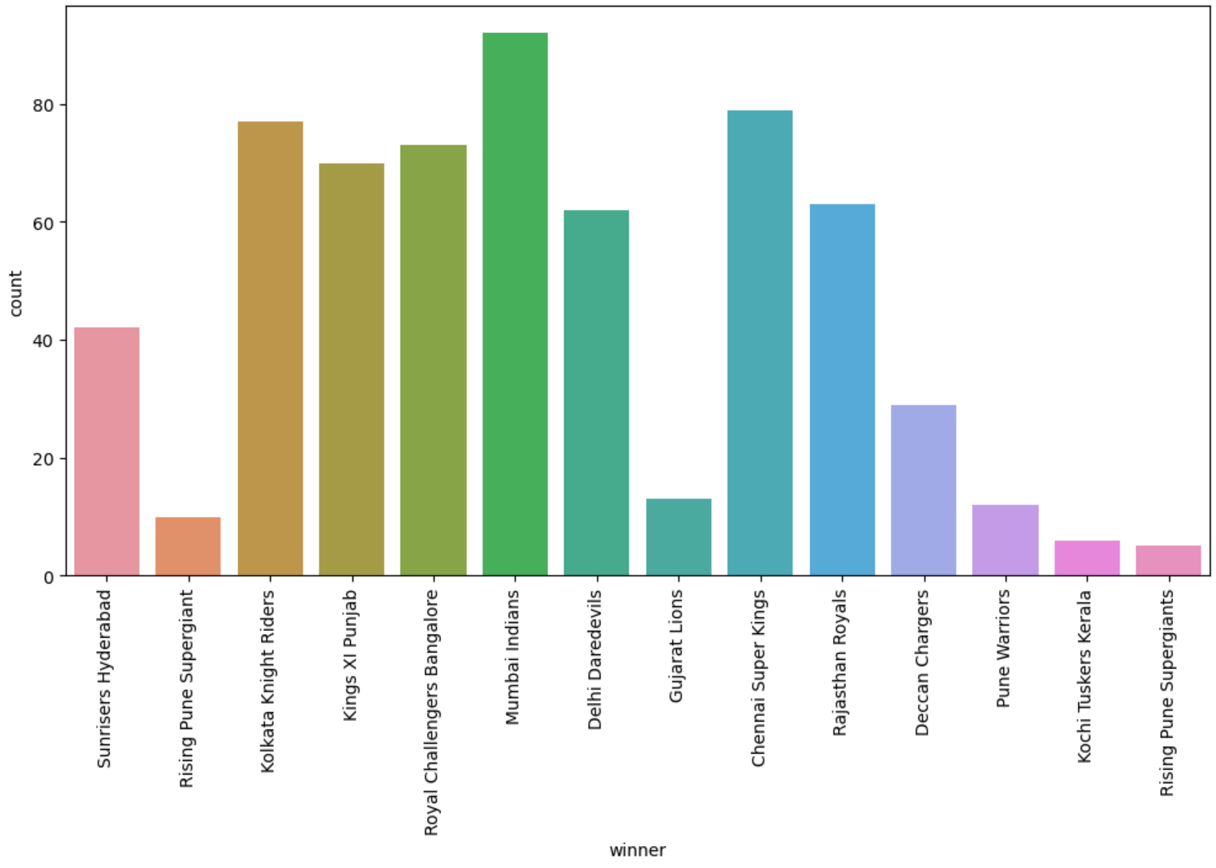
In [33]:
```
temp_df = pd.melt(matches_df, id_vars=['id','season'], value_vars=['team1', 'tear

plt.figure(figsize=(12,6))
sns.countplot(x='value', data=temp_df)
plt.xticks(rotation='vertical')
plt.show()
```



"Mumbai Indians" lead the pack with most number of matches played followed by "Royal Challengers Bangalore". There are also teams with very few matches like 'Rising Pune Supergiants', 'Gujarat Lions' as they are new teams that came in only last season.

Number of wins per team:

In [34]:
```python
plt.figure(figsize=(12,6))
sns.countplot(x='winner', data=matches_df)
plt.xticks(rotation=90)
plt.show()
```



MI again leads the pack followed by CSK.

Champions each season:

Now let us see the champions in each season.

In [35]:
```
temp_df = matches_df.drop_duplicates(subset=['season'], keep='last')[['season',
temp_df
```

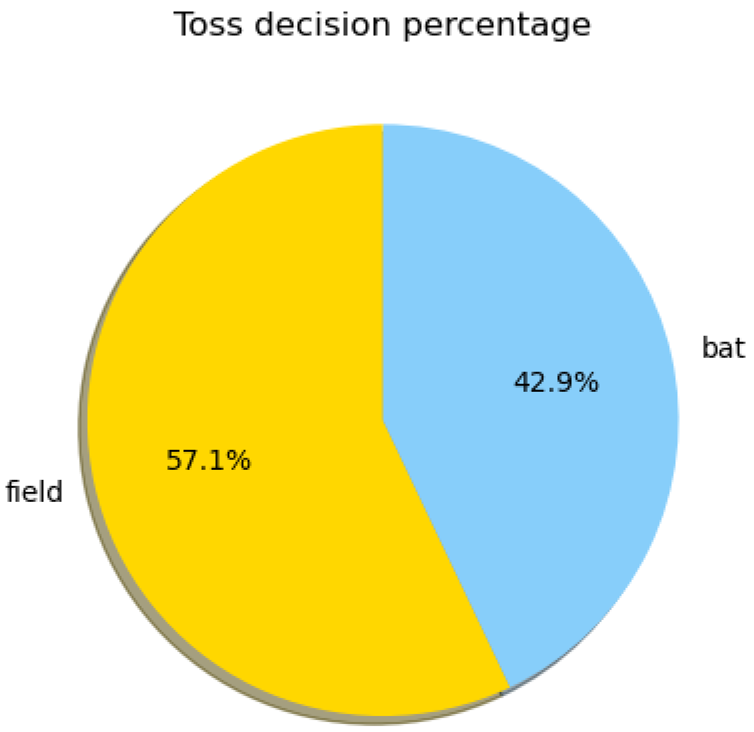
Out[35]:

	season	winner
0	2017	Mumbai Indians
1	2008	Rajasthan Royals
2	2009	Deccan Chargers
3	2010	Chennai Super Kings
4	2011	Chennai Super Kings
5	2012	Kolkata Knight Riders
6	2013	Mumbai Indians
7	2014	Kolkata Knight Riders
8	2015	Mumbai Indians
9	2016	Sunrisers Hyderabad

Toss decision:

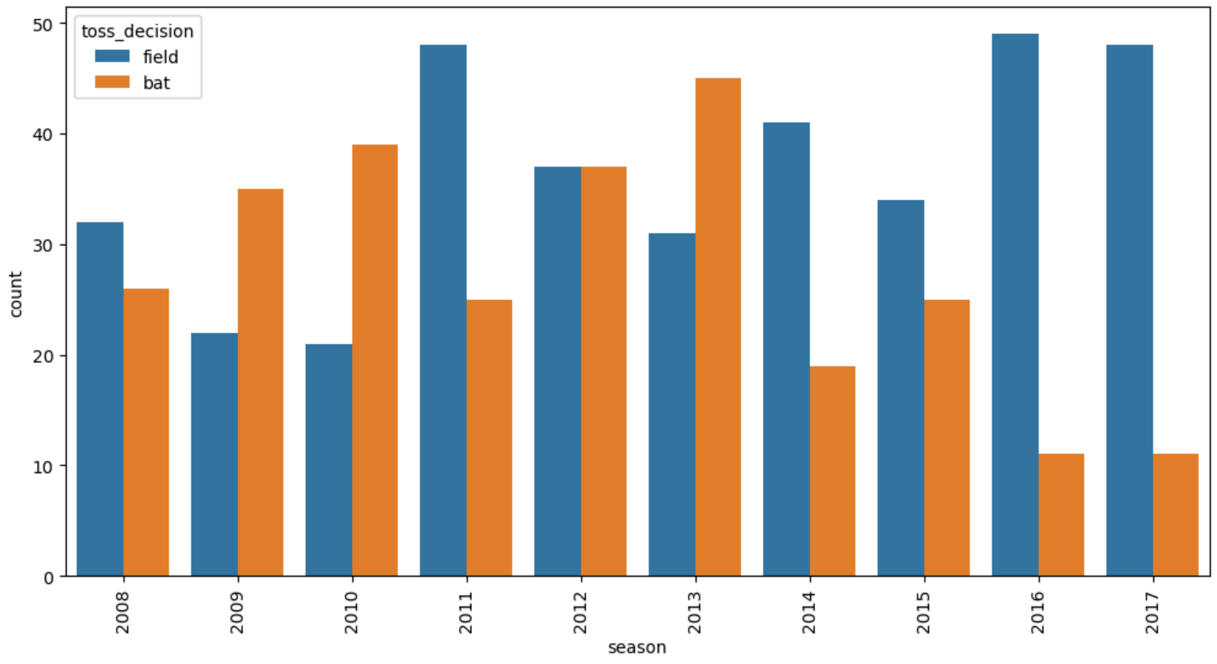
Let us see the toss decisions taken so far.

In [36]:
```
temp_series = matches_df.toss_decision.value_counts()
labels = (np.array(temp_series.index))
sizes = (np.array((temp_series / temp_series.sum())*100))
colors = ['gold', 'lightskyblue']
plt.pie(sizes, labels=labels, colors=colors,
        autopct='%1.1f%%', shadow=True, startangle=90)
plt.title("Toss decision percentage")
plt.show()
```



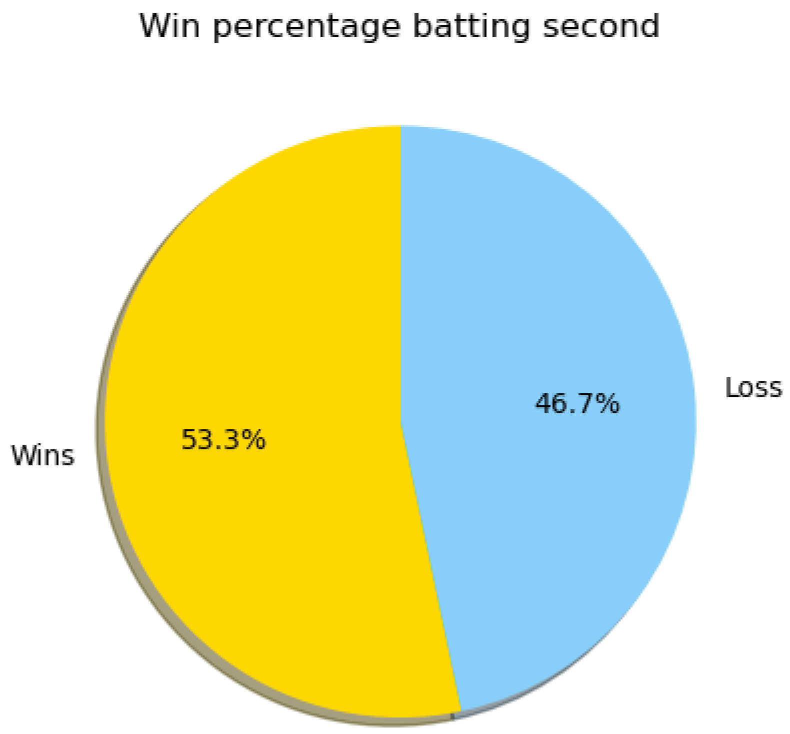
Almost 55% of the toss decisions are made to field first. Now let us see how this decision varied over time.

In [37]:
```
plt.figure(figsize=(12,6))
sns.countplot(x='season', hue='toss_decision', data=matches_df)
plt.xticks(rotation='vertical')
plt.show()
```



It seems during the initial years, teams wanted to bat first. Voila.! Look at the 2016 season, most of the toss decisions are to field first.

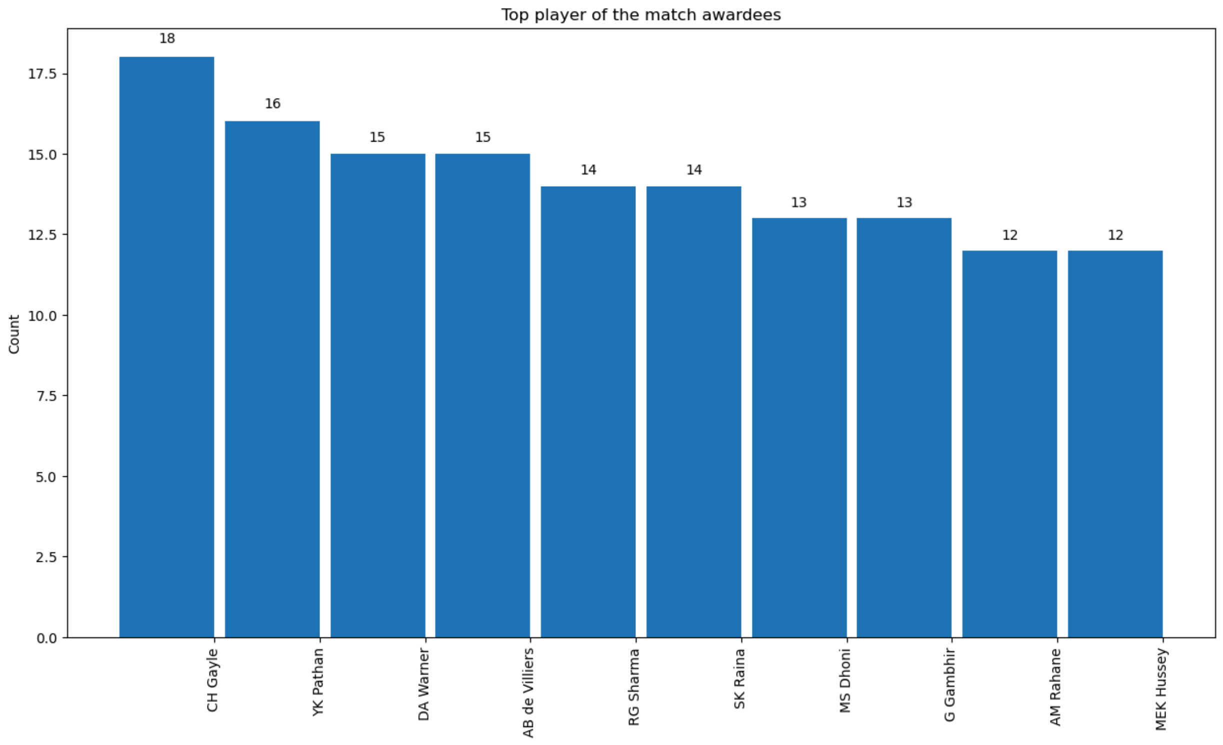
In [38]:
```
# Since there is a very strong trend towards batting second let us see the win pe
num_of_wins = (matches_df.win_by_wickets>0).sum()
num_of_loss = (matches_df.win_by_wickets==0).sum()
labels = ["Wins", "Loss"]
total = float(num_of_wins + num_of_loss)
sizes = [(num_of_wins/total)*100, (num_of_loss/total)*100]
colors = ['gold', 'lightskyblue']
plt.pie(sizes, labels=labels, colors=colors,
        autopct='%1.1f%%', shadow=True, startangle=90)
plt.title("Win percentage batting second")
plt.show()
```



So percentage of times teams batting second has won is 53.2. Now let us split this by year and see the distribution.

In [39]:
```
# create a function for labeling #
def autolabel(rects):
    for rect in rects:
        height = rect.get_height()
        ax.text(rect.get_x() + rect.get_width()/2., 1.02*height,
                '%d' % int(height),
                ha='center', va='bottom')
```

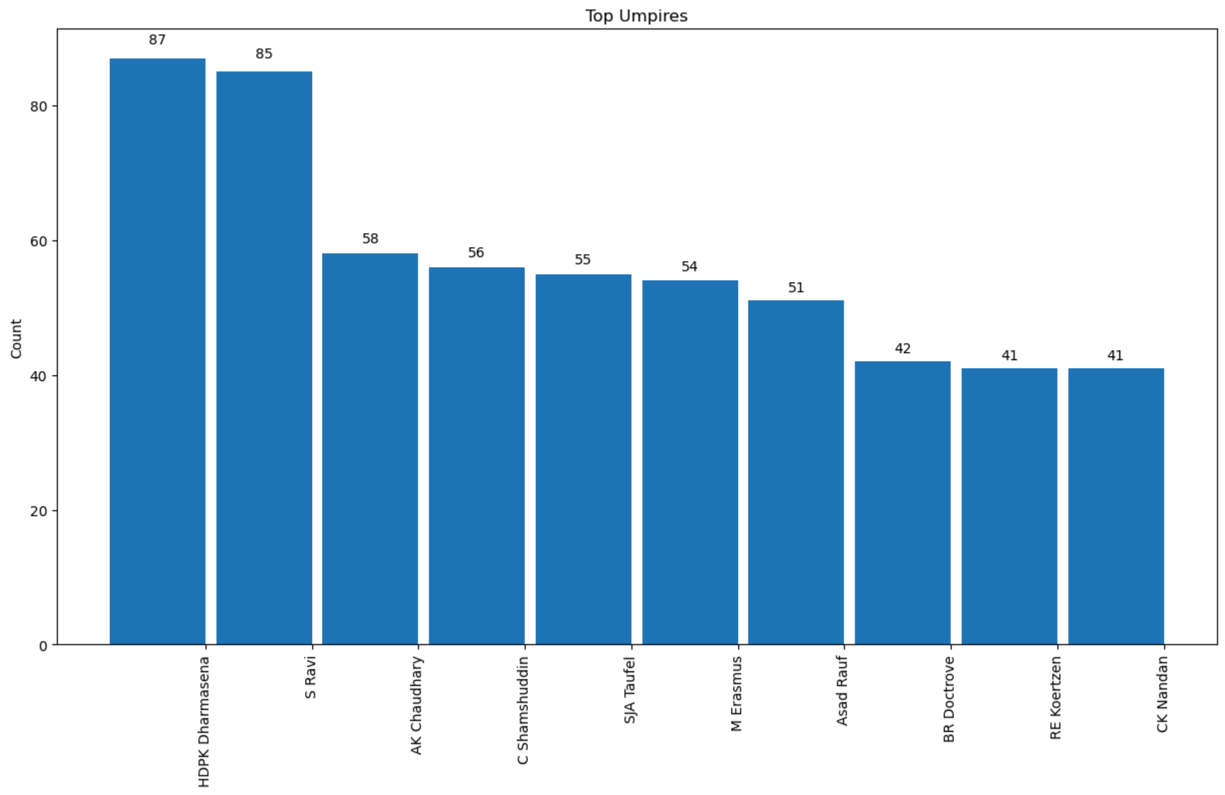
In [40]:
```
temp_series = matches_df.player_of_match.value_counts()[:10]
labels = np.array(temp_series.index)
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_series), width=width)
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation='vertical')
ax.set_ylabel("Count")
ax.set_title("Top player of the match awardees")
autolabel(rects)
plt.show()
```



CH Gayle is the top player of the match awardee in all the seasons of IPL.

Top Umpires:

In [41]:
```
temp_df = pd.melt(matches_df, id_vars=['id'], value_vars=['umpire1', 'umpire2'])

temp_series = temp_df.value.value_counts()[:10]
labels = np.array(temp_series.index)
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_series), width=width,)
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation='vertical')
ax.set_ylabel("Count")
ax.set_title("Top Umpires")
autolabel(rects)
plt.show()
```



Dharmasena seems to be the most sought after umpire for IPL matches followed by Ravi. Others are fairly close to each other.

Score Data Set

In [42]:
```
score_df.head()
```

Out[42]:

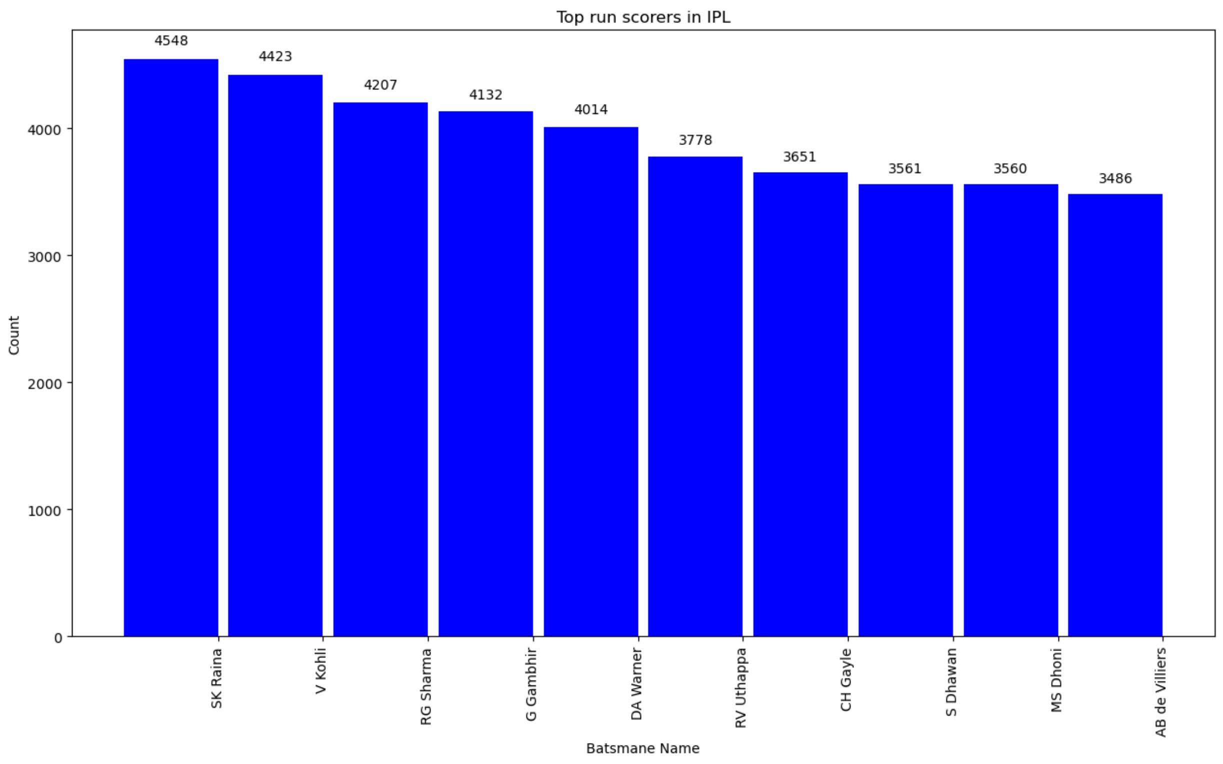
	match_id	inning	batting_team	bowling_team	over	ball	batsman	non_striker	bowler	is_supe
0	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	1	DA Warner	S Dhawan	TS Mills	
1	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	2	DA Warner	S Dhawan	TS Mills	
2	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	3	DA Warner	S Dhawan	TS Mills	
3	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	4	DA Warner	S Dhawan	TS Mills	
4	1	1	Sunrisers Hyderabad	Royal Challengers Bangalore	1	5	DA Warner	S Dhawan	TS Mills	

5 rows × 21 columns

Batsman analysis:

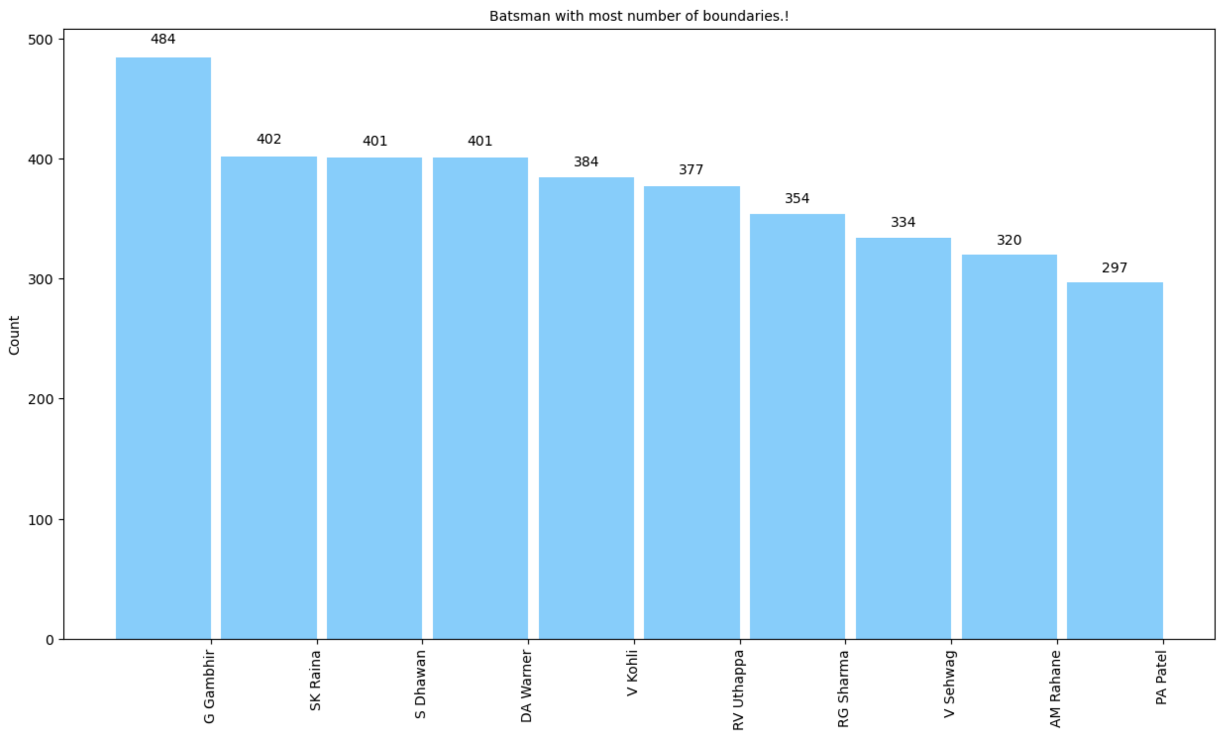
Let us start our analysis with batsman. Let us first see the ones with most number of IPL runs under their belt

In [43]:
```
temp_df = score_df.groupby('batsman')['batsman_runs'].agg('sum').reset_index().so
temp_df = temp_df.iloc[:10,:]

labels = np.array(temp_df['batsman'])
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_df['batsman_runs']), width=width, color='blue')
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation='vertical')
ax.set_ylabel("Count")
ax.set_title("Top run scorers in IPL")
ax.set_xlabel('Batsmane Name')
autolabel(rects)
plt.show()
```



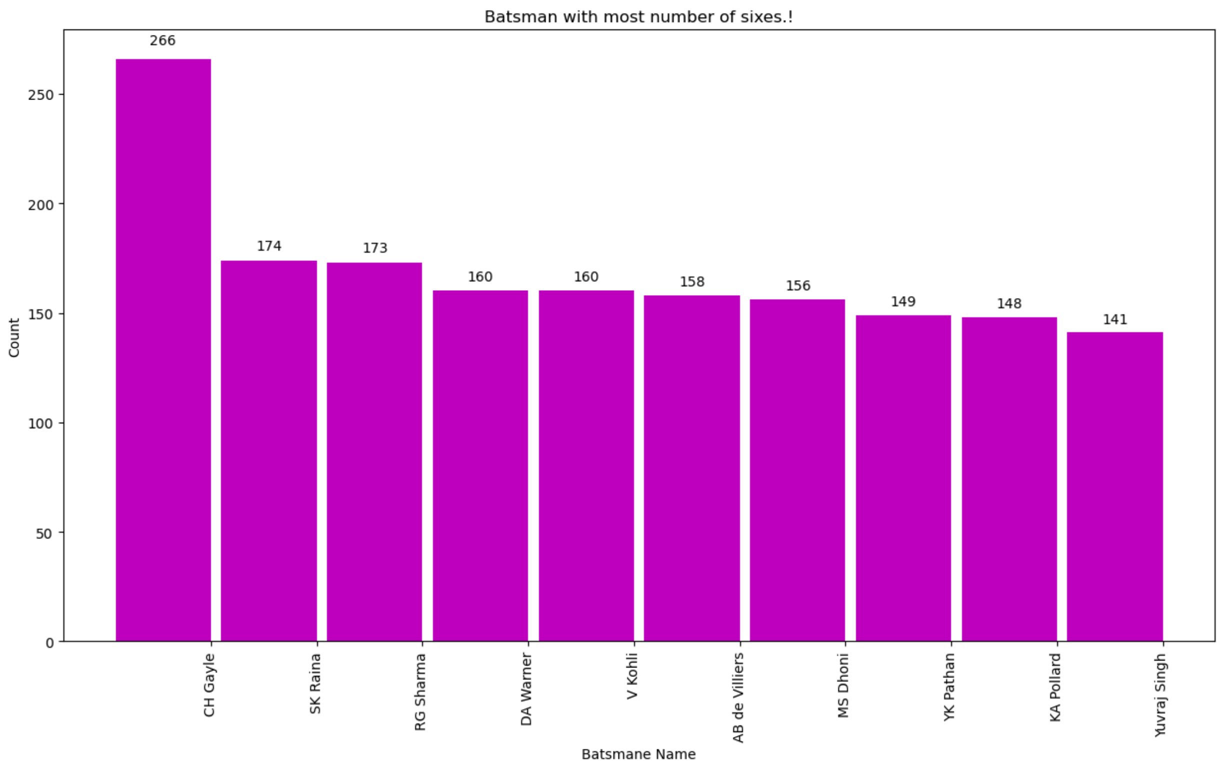
Virat Kohli is leading the chart followed closely by Raina. Gayle is the top scorer among foreign players.

In [44]:
```
# Now let us see the players with more number of boundaries in IPL.
temp_df = score_df.groupby('batsman')['batsman_runs'].agg(lambda x: (x==4).sum()
temp_df = temp_df.iloc[:10,:]

labels = np.array(temp_df['batsman'])
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_df['batsman_runs']), width=width, color='lights
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation='vertical')
ax.set_ylabel("Count")
ax.set_title("Batsman with most number of boundaries.!",fontsize = 10)
autolabel(rects)
plt.show()
```



Gambhir is way ahead of others - almost 60 boundaries more than Kohli.! Nice to Sachin in the top 10 list :)

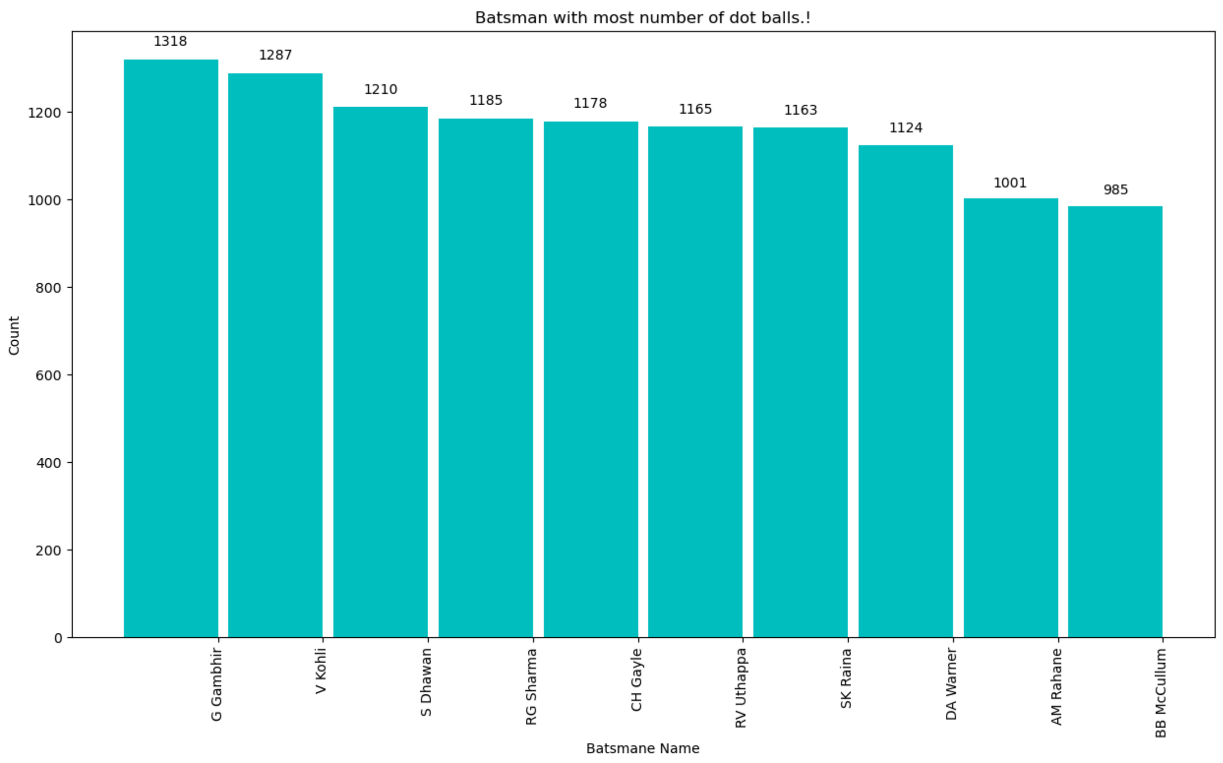
In [45]:
```
# Now let us check the number of 6's
temp_df = score_df.groupby('batsman')['batsman_runs'].agg(lambda x: (x==6).sum()
temp_df = temp_df.iloc[:10,:]

labels = np.array(temp_df['batsman'])
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_df['batsman_runs']), width=width, color='m')
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation=90)
ax.set_ylabel("Count")
ax.set_title("Batsman with most number of sixes.!")
ax.set_xlabel('Batsmane Name')
autolabel(rects)
plt.show()
```



There you see the big man. Gayle, the unassailable leader in the number of sixes.

Raina is third in both number of 4's and 6's

In [46]:
```
# Now let us see the batsman who has played the most number of dot balls.
temp_df = score_df.groupby('batsman')['batsman_runs'].agg(lambda x: (x==0).sum()
temp_df = temp_df.iloc[:10,:]

labels = np.array(temp_df['batsman'])
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_df['batsman_runs']), width=width, color='c')
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation='vertical')
ax.set_ylabel("Count")
ax.set_title("Batsman with most number of dot balls.!")
ax.set_xlabel('Batsmane Name')
autolabel(rects)
plt.show()
```

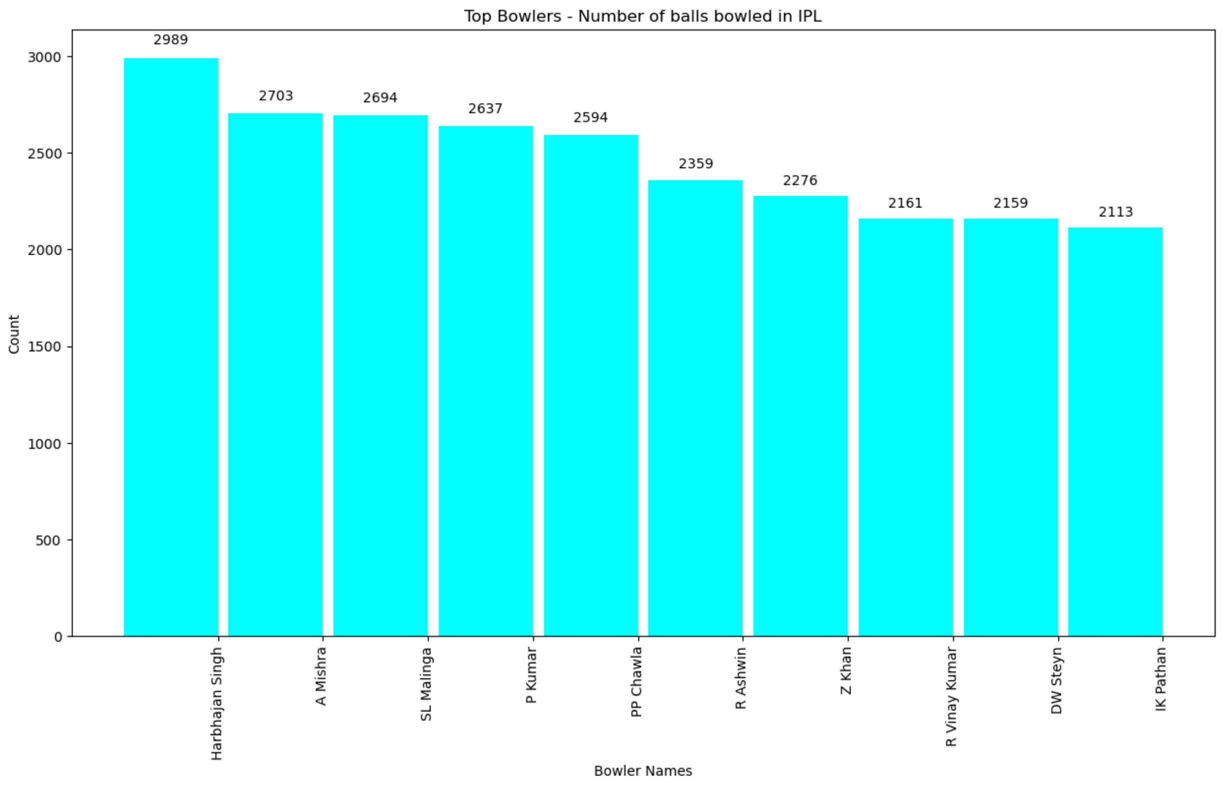


It is interesting to see that the same names repeat again here as well. I think since these guys have played more number of balls, they have more dot balls as well.

Bowler Analysis

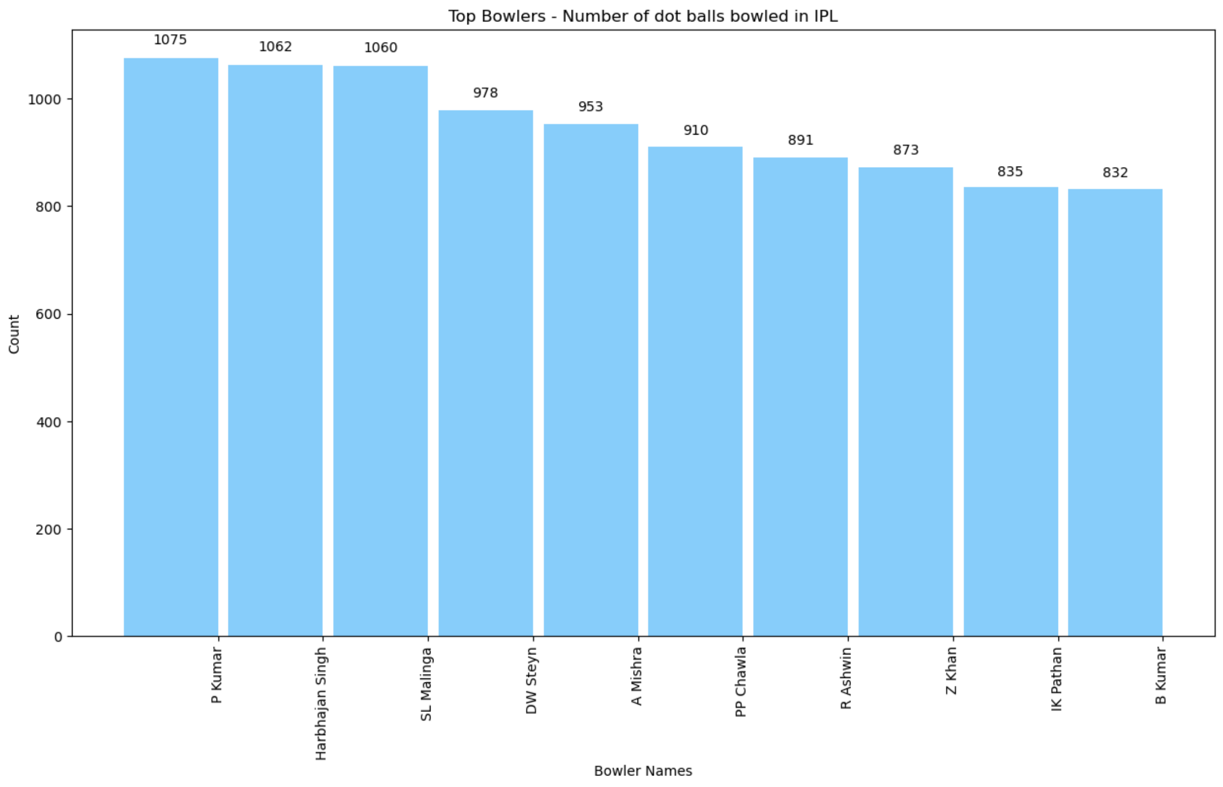
Now let us see the bowlers who has bowled most number of balls in IPL.

In [48]:
```
temp_df = score_df.groupby('bowler')['ball'].agg('count').reset_index().sort_valu
temp_df = temp_df.iloc[:10,:]

labels = np.array(temp_df['bowler'])
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_df['ball']), width=width, color='cyan')
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation='vertical')
ax.set_ylabel("Count")
ax.set_title("Top Bowlers - Number of balls bowled in IPL")
ax.set_xlabel('Bowler Names')
autolabel(rects)
plt.show()
```



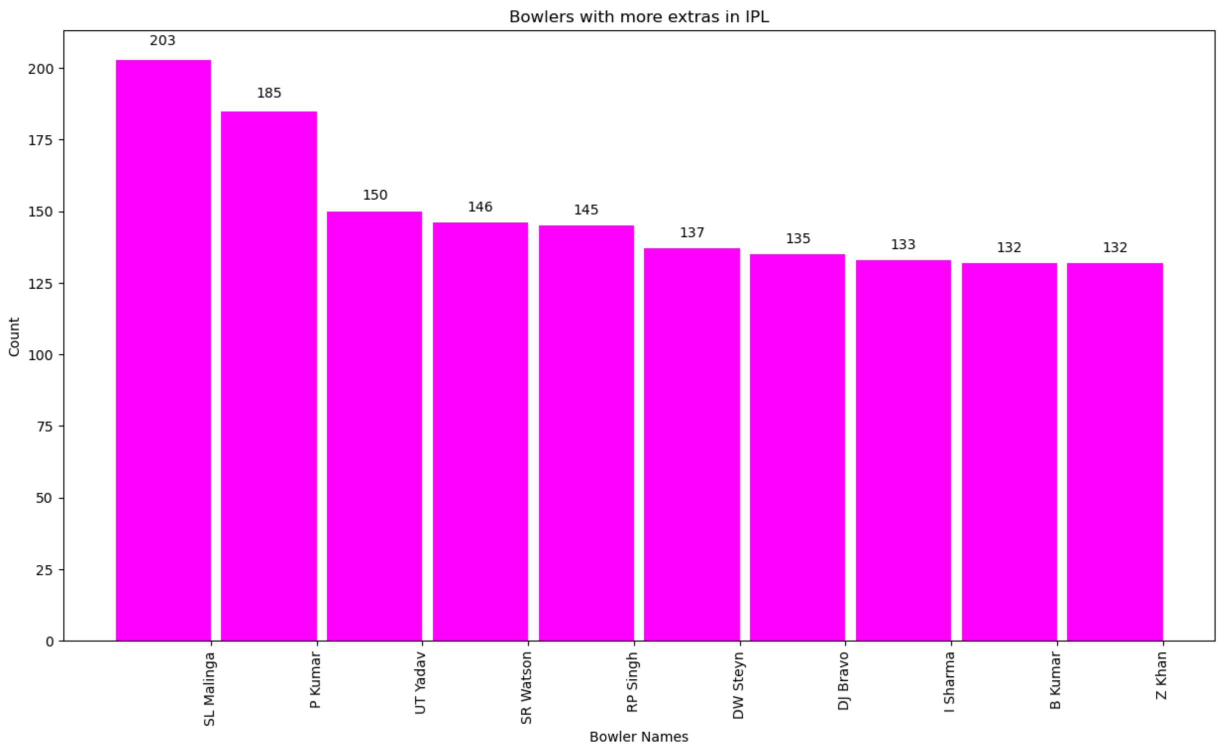
Harbhajan Singh is the the bowler with most number of balls bowled in IPL matches. Now let us see the bowler with more number of dot balls.

In [49]:
```
temp_df = score_df.groupby('bowler')['total_runs'].agg(lambda x: (x==0).sum()).re
temp_df = temp_df.iloc[:10,:]

labels = np.array(temp_df['bowler'])
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_df['total_runs']), width=width, color='lightsky
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation='vertical')
ax.set_ylabel("Count")
ax.set_title("Top Bowlers - Number of dot balls bowled in IPL")
ax.set_xlabel('Bowler Names')
autolabel(rects)
plt.show()
```



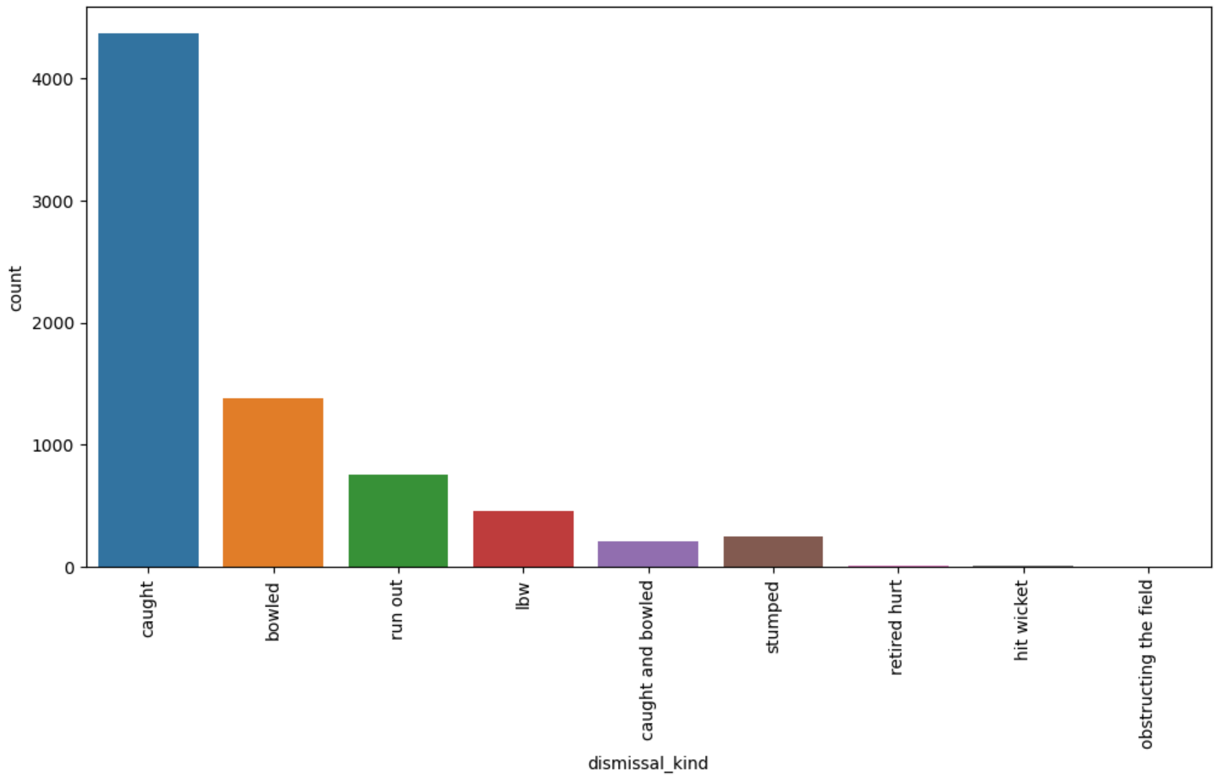
Pravin Kumar is the one with more number of dot balls followed by Steyn and Malinga

In [50]:
```
#Now let us see the bowlers who has bowled more number of extras in IPL.
temp_df = score_df.groupby('bowler')['extra_runs'].agg(lambda x: (x>0).sum()).res
temp_df = temp_df.iloc[:10,:]

labels = np.array(temp_df['bowler'])
ind = np.arange(len(labels))
width = 0.9
fig, ax = plt.subplots(figsize=(15,8))
rects = ax.bar(ind, np.array(temp_df['extra_runs']), width=width, color='magenta
ax.set_xticks(ind+((width)/2.))
ax.set_xticklabels(labels, rotation='vertical')
ax.set_ylabel("Count")
ax.set_title("Bowlers with more extras in IPL")
ax.set_xlabel('Bowler Names')
autolabel(rects)
plt.show()
```



Malinga tops the chart with 221 extra runs followed by Pravin Kumar.

In [51]:
```
# Now let us see most common dismissal types in IPL.
plt.figure(figsize=(12,6))
sns.countplot(x='dismissal_kind', data=score_df)
plt.xticks(rotation='vertical')
plt.show()
```



Caught is the most common dismissal type in IPL followed by Bowled. There are very few instances of hit wicket as well. 'Obstructing the field' is one of the dismissal type as well in IPL.!