Virat Kohli Centuries Analysis

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
```

In [2]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
df = pd.read_csv("virat_centuries.csv")
```

In [4]:

```
df.head()
```

Out[4]:

	Score	Out/Not Out	Against	Batting Order	Inn.	Strike Rate	Venue	Column1	H/A	Date	I
0	116	Out	Australia	6	2	NaN	Adelaide Oval	Adelaide	Away	24-01-2012	
1	103	Out	New Zealand	5	2	NaN	M. Chinnaswamy Stadium	Bangalore	Home	31-08-2012	
2	103	Out	England	5	2	NaN	Vidarbha Cricket Association Stadium	Nagpur	Home	13-12-2012	
3	107	Out	Australia	5	2	NaN	M. A. Chidambaram Stadium	Chennai	Home	22-02-2013	
4	119	Out	South Africa	4	1	NaN	Wanderers Stadium	Johannesburg	Away	18-12-2013	

In [5]:

```
df.tail()
```

Out[5]:

	Score	Out/Not Out	Against	Batting Order	Inn.	Strike Rate	Venue	Column1	H/A	D
71	113	Out	Bangladesh	3	1	124.80	Zohur Ahmed Chaudhary	Chittagong	Away	2(
72	113	Out	Sri Lanka	3	1	129.89	Barsapara	Guwahati	Home	2(
73	166	Not Out	Sri Lanka	3	1	150.91	Green field	Thiruvanantpuram	Home	2(
74	186	Out	Australia	4	1	51.09	Motera	Ahemdabad	Home	2(
75	121	Out	West Indies	4	1	58.74	Queen's Park Oval	Port of Spain	Away	2(

In [6]:

```
df.shape
```

Out[6]:

```
(76, 14)
```

In [7]:

```
df.columns
```

Out[7]:

```
Index(['Score', 'Out/Not Out', 'Against', 'Batting Order', 'Inn.',
       'Strike Rate', 'Venue', 'Column1', 'H/A', 'Date', 'Result', 'Forma
t',
       'Man of the Match', 'Captain'],
      dtype='object')
```

In [8]:

```
df.duplicated().sum()
```

Out[8]:

```
0
```

In [9]:

```
df.isnull().sum()
```

Out[9]:

```
Score                0
Out/Not Out          0
Against              0
Batting Order        0
Inn.                 0
Strike Rate         27
Venue                0
Column1              0
H/A                  0
Date                 0
Result               0
Format               0
Man of the Match     0
Captain              0
dtype: int64
```

In [10]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 76 entries, 0 to 75
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Score             76 non-null     int64  
 1   Out/Not Out       76 non-null     object 
 2   Against           76 non-null     object 
 3   Batting Order     76 non-null     int64  
 4   Inn.              76 non-null     int64  
 5   Strike Rate       49 non-null     float64
 6   Venue             76 non-null     object 
 7   Column1           76 non-null     object 
 8   H/A               76 non-null     object 
 9   Date              76 non-null     object 
 10  Result            76 non-null     object 
 11  Format            76 non-null     object 
 12  Man of the Match  76 non-null     object 
 13  Captain           76 non-null     object 
dtypes: float64(1), int64(3), object(10)
memory usage: 8.4+ KB
```

In [11]:

```
mean_strike_rate = df['Strike Rate'].mean()
df['Strike Rate'].fillna(mean_strike_rate, inplace=True)
```

In [12]:

```
df.describe()
```

Out[12]:

	Score	Batting Order	Inn.	Strike Rate
count	76.000000	76.000000	76.000000	76.000000
mean	132.644737	3.513158	1.684211	112.903878
std	35.619734	0.702252	0.677457	21.944045
min	100.000000	1.000000	1.000000	51.090000
25%	107.000000	3.000000	1.000000	100.875000
50%	119.500000	3.000000	2.000000	112.903878
75%	140.250000	4.000000	2.000000	114.952500
max	254.000000	6.000000	4.000000	200.000000

In [13]:

```
new_column_names = {
    'Score': 'Batting Score',
    'Out/Not Out': 'Batting Status',
    'Against': 'Opponent Team',
    'Batting Order': 'Batting Position',
    'Inn.': 'Inning Number',
    'Strike Rate': 'Batting Strike Rate',
    'Venue': 'Match Venue',
    'Column1': 'City',
    'H/A': 'Home/Away',
    'Date': 'Match Date',
    'Result': 'Match Result',
    'Format': 'Match Format',
    'Man of the Match': 'Player of the Match',
    'Captain': 'Team Captain'
}
```

In [14]:

```
df.rename(columns=new_column_names, inplace=True)
```

In [15]:

```
df.nunique()
```

Out[15]:

```
Batting Score          50
Batting Status          2
Opponent Team          10
Batting Position        5
Inning Number           4
Batting Strike Rate    50
Match Venue            49
City                   47
Home/Away               2
Match Date             75
Match Result            6
Match Format            3
Player of the Match     2
Team Captain            2
dtype: int64
```

In [16]:

```
column_data_types = df.dtypes
```

In [17]:

```
categorical_columns = column_data_types[column_data_types == 'object'].index.tolist()
numerical_columns = column_data_types[column_data_types != 'object'].index.tolist()

print("Categorical Columns:", categorical_columns)
print()
print("Numerical Columns:", numerical_columns)
```

```
Categorical Columns: ['Batting Status', 'Opponent Team', 'Match Venue', 'C
ity', 'Home/Away', 'Match Date', 'Match Result', 'Match Format', 'Player o
f the Match', 'Team Captain']

Numerical Columns: ['Batting Score', 'Batting Position', 'Inning Number',
'Batting Strike Rate']
```

In [18]:

```
for i in categorical_columns:
    print(i, '- unique values are:')
    print(df[i].unique())
    print()
```

```
Batting Status - unique values are:
['Out' 'Not Out']

Opponent Team - unique values are:
['Australia' 'New Zealand' 'England' 'South Africa' 'Sri Lanka'
 'West Indies' 'Bangladesh' 'Pakistan' 'Zimbabwe' 'Afganistan']

Match Venue - unique values are:
['Adelaide Oval' 'M. Chinnaswamy Stadium'
 'Vidarbha Cricket Association Stadium' 'M. A. Chidambaram Stadium'
 'Wanderers Stadium' 'Basin Reserve' 'Melbourne Cricket Ground'
 'Sydney Cricket Ground' 'Galle International Stadium'
 'Sir Vivian Richards Stadium' 'Holkar Stadium' 'ACA-VDCA Cricket Stadium'
 'Wankhede Stadium' 'Rajiv Gandhi International Cricket Stadium'
 'Eden Gardens' 'Feroz Shah Kotla Ground' 'SuperSport Park'
 'Edgbaston Cricket Ground' 'Trent Bridge'
 'Saurashtra Cricket Association Stadium' 'Perth Stadium'
 'Maharashtra Cricket Association Stadium' 'Sher-e-Bangla Cricket Stadium'
 'APCA-VDCA Stadium' 'Nehru Stadium' 'Sophia Gardens' 'Bellerive Oval'
 'MRIC Stadium' 'R. Premadasa Stadium' "Queen's Park Oval"
 'Harare Sports Club' 'Sawai Mansingh Stadium' 'VCA Stadium' 'McLean Park'
 'Khan Shaheb Osman Ali Stadium' 'HPCA Stadium'
 'JSCA International Stadium' 'Manuka Oval'
 'Punjab Cricket Association IS Bindra Stadium' 'Sabina Park'
 'Green Park Stadium' 'Kingsmead Cricket Ground' 'Newlands Cricket Ground'
 'ACA Stadium' 'Dubai International Cricket Stadium'
 'Zohur Ahmed Chaudhary' 'Barsapara' 'Green field' 'Motera']

City - unique values are:
[' Adelaide' ' Bangalore' ' Nagpur' ' Chennai' ' Johannesburg'
 ' Wellington' ' Melbourne' ' Sydney' ' Galle' ' North Sound' ' Indore'
 ' Visakhapatnam' ' Mumbai' ' Hyderabad' ' Kolkata' ' Delhi' ' Centurion'
 ' Birmingham' ' Nottingham' ' Rajkot' ' Perth' ' Pune' ' Dhaka'
 ' Guwahati' ' Cardiff' ' Hobart' ' Hambantota' ' Colombo'
 ' Port of Spain' ' Harare' ' Jaipur' ' Napier' ' Fatullah' ' Dharamshala'
 ' Ranchi' ' Canberra' ' Mohali' ' Kingston' ' Kanpur' ' Durban'
 ' Cape Town' 'Dubai' 'Chittagong' 'Guwahati' 'Thiruvanantpuram'
 'Ahemdabad' 'Port of Spain']

Home/Away - unique values are:
['Away' 'Home']

Match Date - unique values are:
['24-01-2012' '31-08-2012' '13-12-2012' '22-02-2013' '18-12-2013'
 '14-02-2014' '09-12-2014' '26-12-2014' '06-01-2015' '12-08-2015'
 '21-07-2016' '08-10-2016' '17-11-2016' '08-12-2016' '09-02-2017'
 '26-07-2017' '16-11-2017' '24-11-2017' '02-12-2017' '13-01-2018'
 '01-08-2018' '18-08-2018' '04-10-2018' '14-12-2018' '10-10-2019'
 '22-11-2019' '24-12-2009' '11-01-2010' '20-10-2010' '28-11-2010'
 '19-02-2011' '16-09-2011' '17-10-2011' '02-12-2011' '28-02-2012'
 '13-03-2012' '18-03-2012' '21-07-2012' '31-07-2012' '05-07-2013'
 '24-07-2013' '16-10-2013' '30-10-2013' '19-01-2014' '26-02-2014'
 '17-10-2014' '16-11-2014' '15-02-2015' '22-10-2015' '17-01-2016'
 '20-01-2016' '23-10-2016' '15-01-2017' '06-07-2017' '31-08-2017'
 '03-09-2017' '22-10-2017' '29-10-2017' '01-02-2018' '07-02-2018'
 '16-02-2018' '21-10-2018' '24-10-2018' '27-10-2018' '15-01-2019'
 '05-03-2019' '08-03-2019' '11-08-2019' '14-08-2019' '08-09-2022'
 '10-12-2022' '10-01-2023' '15-01-2023' '12-03-2023' '21-07-2023']

Match Result - unique values are:
['Lost' 'Won' 'Drawn' 'Lost (D/L)' 'Won (D/L)' 'Tied']
```

```
Match Format - unique values are:
['Test' 'ODI' 'T20I']

Player of the Match - unique values are:
['No' 'Yes']

Team Captain - unique values are:
['No' 'Yes']
```

In [19]:

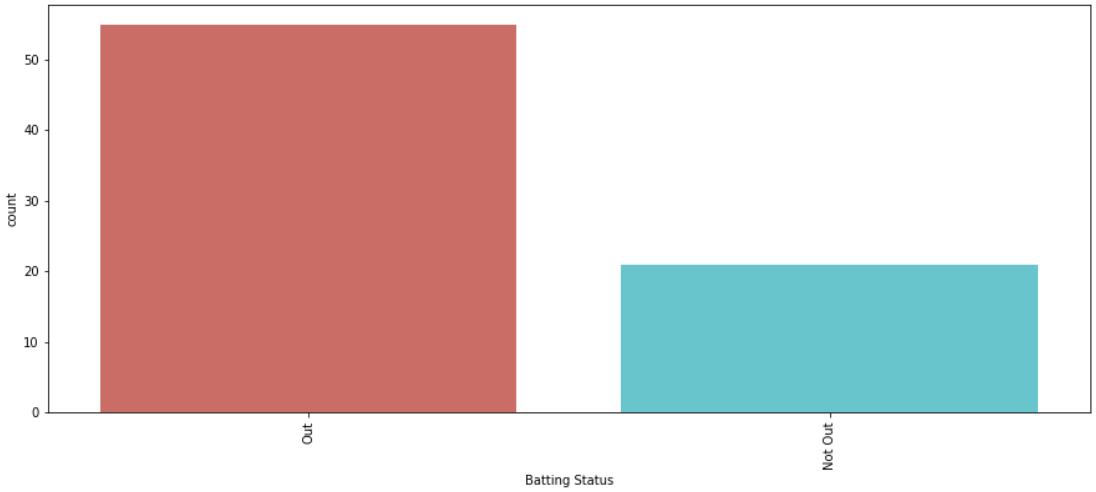
```
for i in categorical_columns:
    print(i, '- value counts are:')
    print(df[i].value_counts())
    print()
```

```
Batting Status - value counts are:
Out        55
Not Out    21
Name: Batting Status, dtype: int64

Opponent Team - value counts are:
Australia       16
Sri Lanka       15
West Indies     12
New Zealand      8
England          8
South Africa     7
Bangladesh       6
Pakistan         2
Zimbabwe         1
Afganistan       1
Name: Opponent Team, dtype: int64

Match Venue - value counts are:
```

In [20]:

```
for i in categorical_columns:
    if i != 'Match Date':
        print(i, '- Countplot:')
        plt.figure(figsize=(15,6))
        sns.countplot(df[i], data = df, palette = 'hls')
        plt.xticks(rotation = 90)
        plt.show()
```

```
Batting Status - Countplot:
```

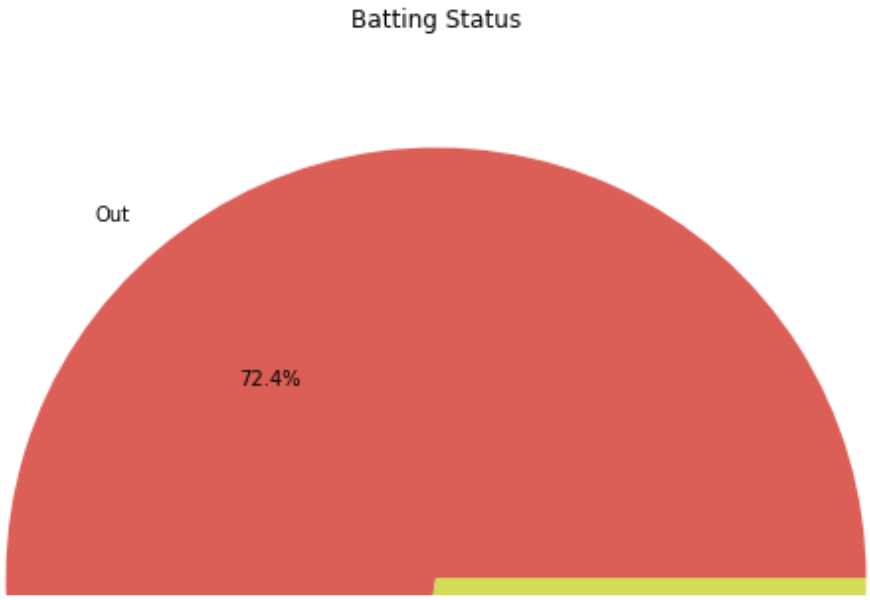



```
Opponent Team - Countplot:
```

In [21]:

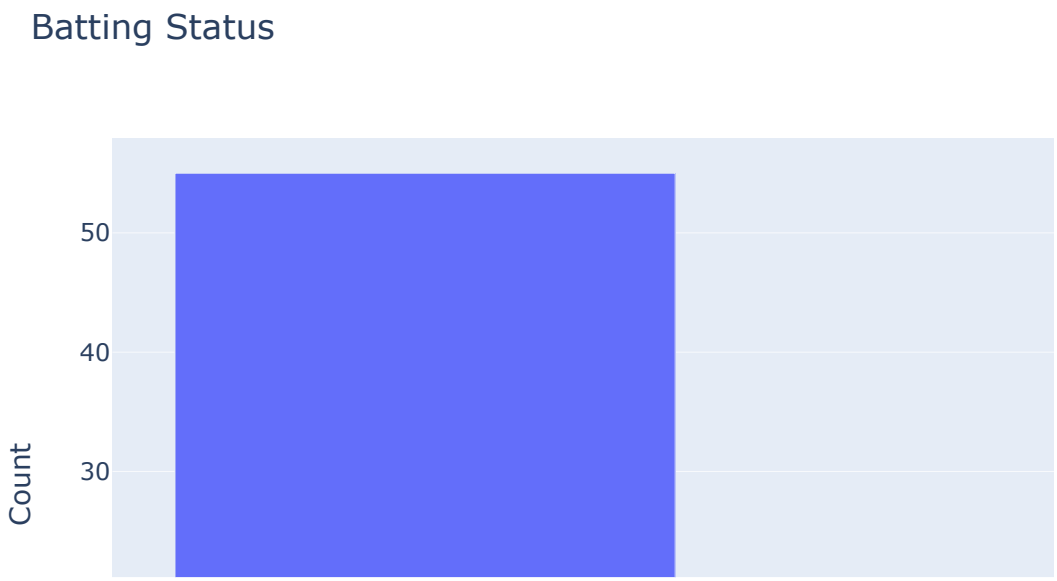
```
for i in categorical_columns:
    if i != 'Match Date':
        print(i, '- Pieplot:')
        plt.figure(figsize=(10, 10))
        counts = df[i].value_counts()
        plt.pie(counts, labels=counts.index, autopct='%1.1f%%', colors=sns.color_palette
        plt.title(i)
        plt.show()
```

```
Batting Status - Pieplot:
```



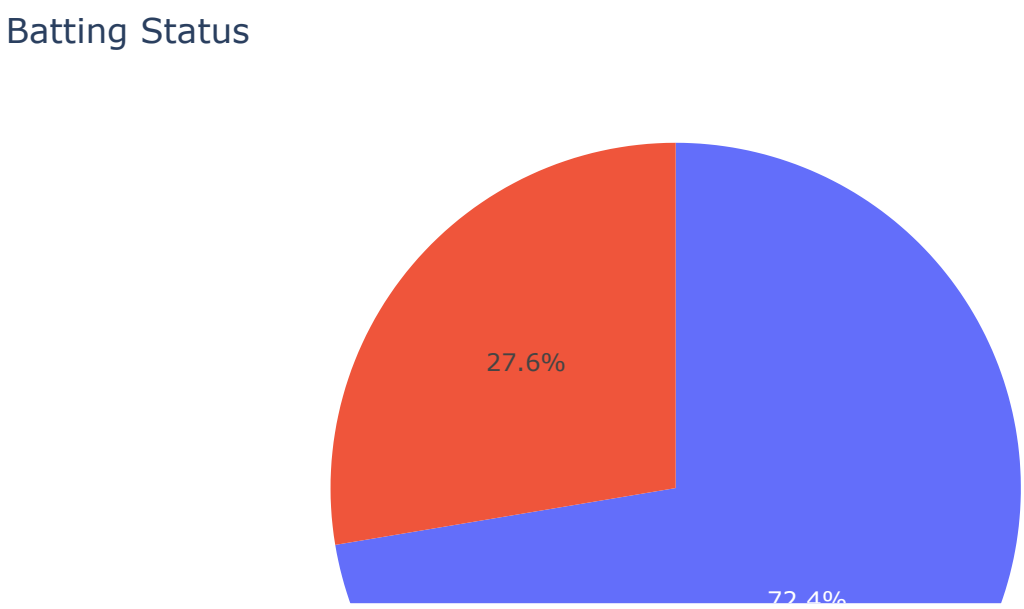
In [22]:

```
for i in categorical_columns:
    if i != 'Match Date':
        fig = go.Figure(data=[go.Bar(x=df[i].value_counts().index, y=df[i].value_counts(
        fig.update_layout(title=i,xaxis_title="Categories",yaxis_title="Count")
        fig.show()
```



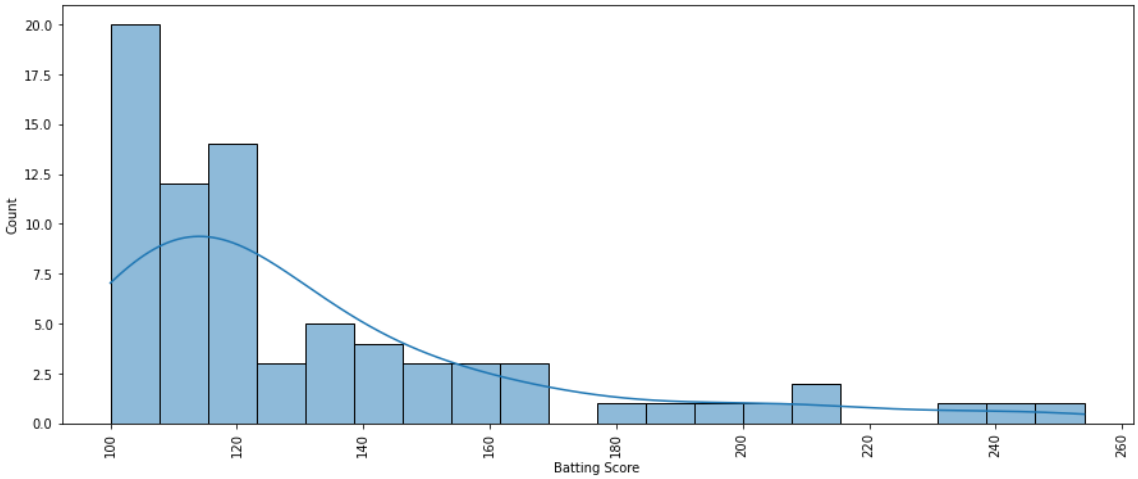
In [23]:

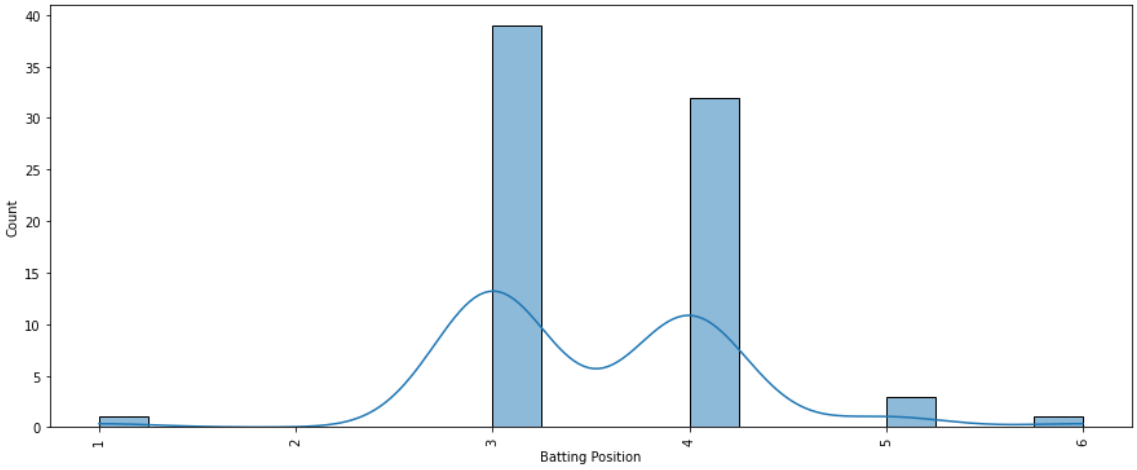
```
for i in categorical_columns:
    if i != 'Match Date':
        counts = df[i].value_counts()
        fig = go.Figure(data=[go.Pie(labels=counts.index, values=counts)])
        fig.update_layout(title=i)
        fig.show()
```

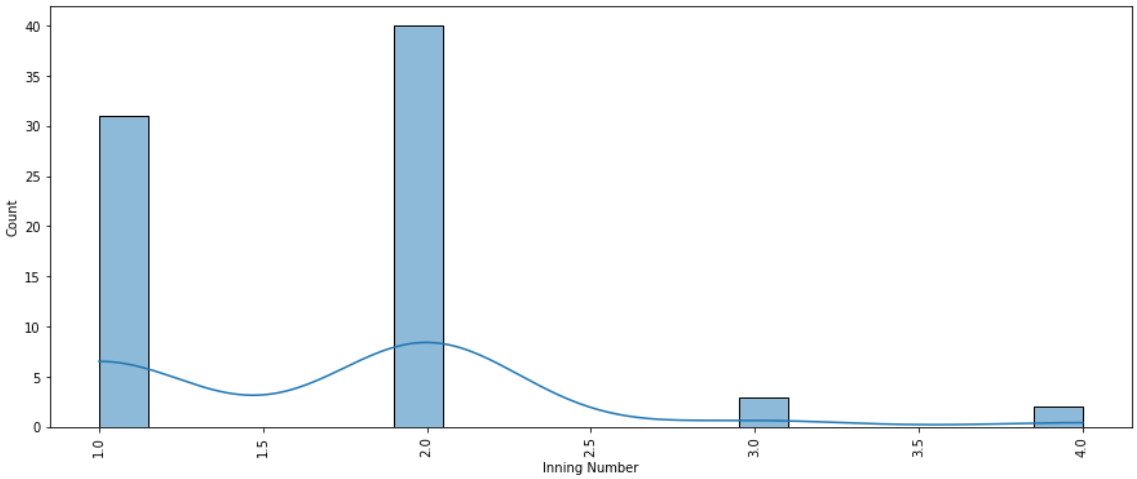


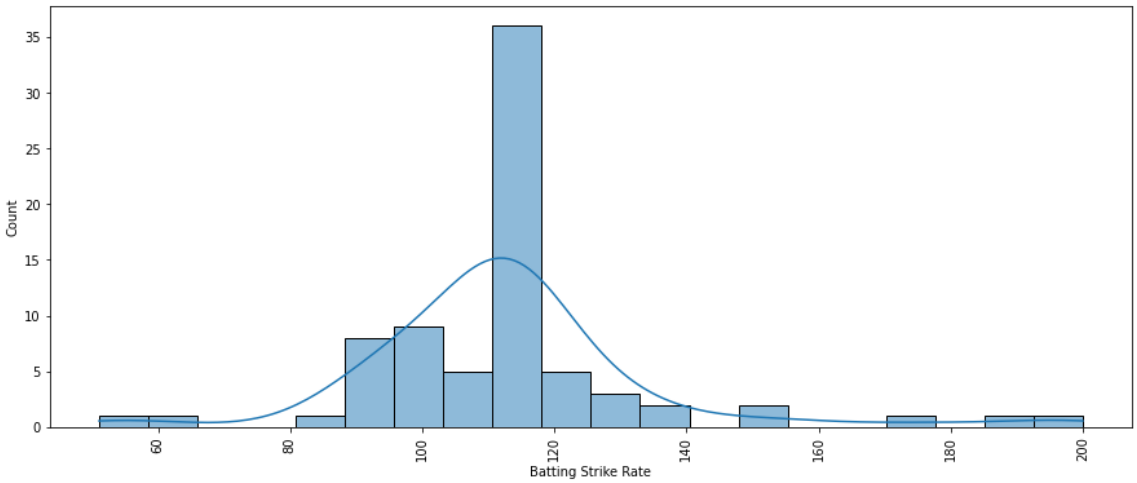
In [24]:

```
for i in numerical_columns:
    plt.figure(figsize=(15, 6))
    sns.histplot(df[i], kde=True, bins=20, palette='hls')
    plt.xticks(rotation=90)
    plt.show()
```



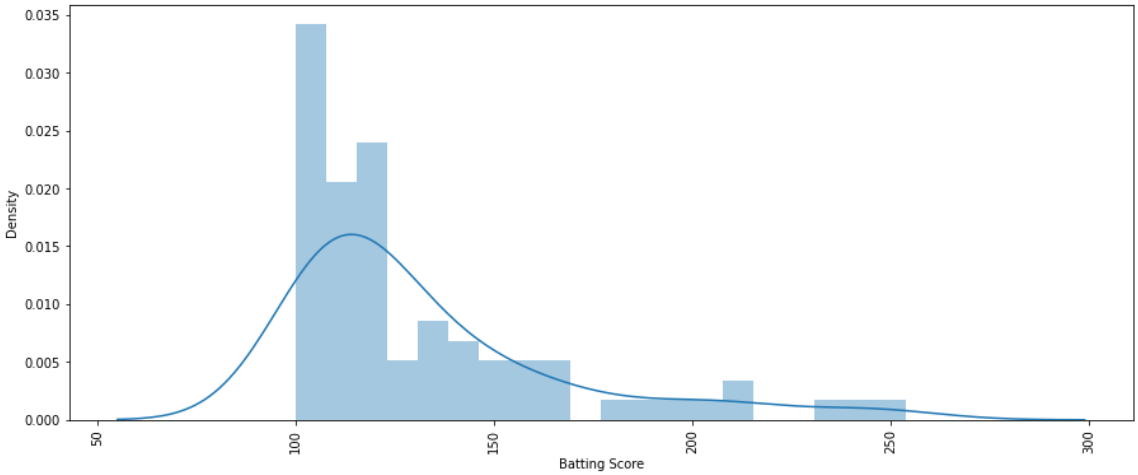


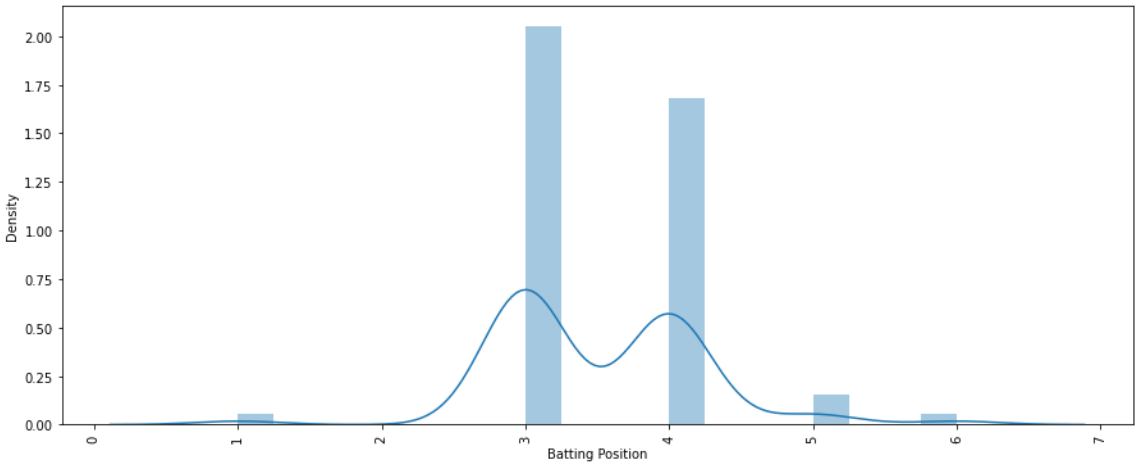


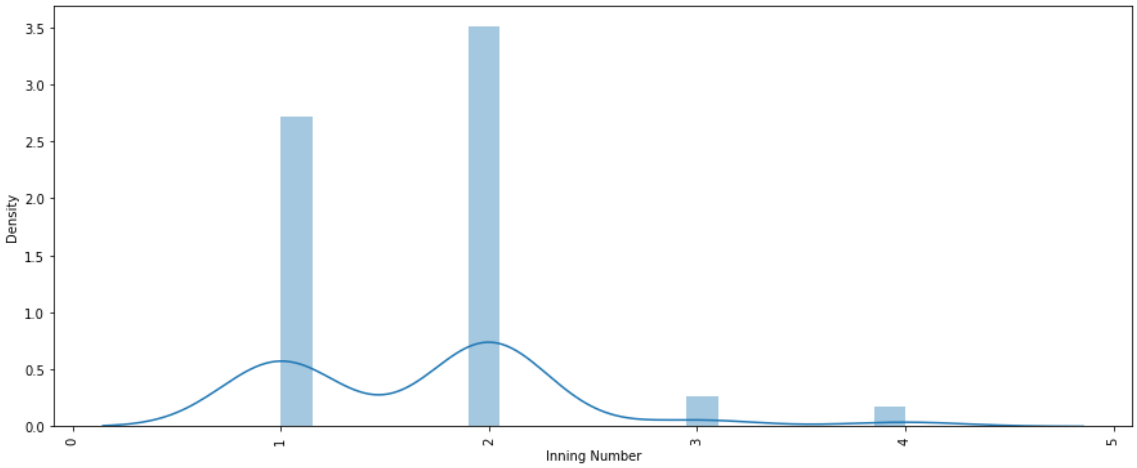

Count
Batting Strike Rate
0
5
10
15
20
25
30
35
60
80
100
120
140
160
180
200

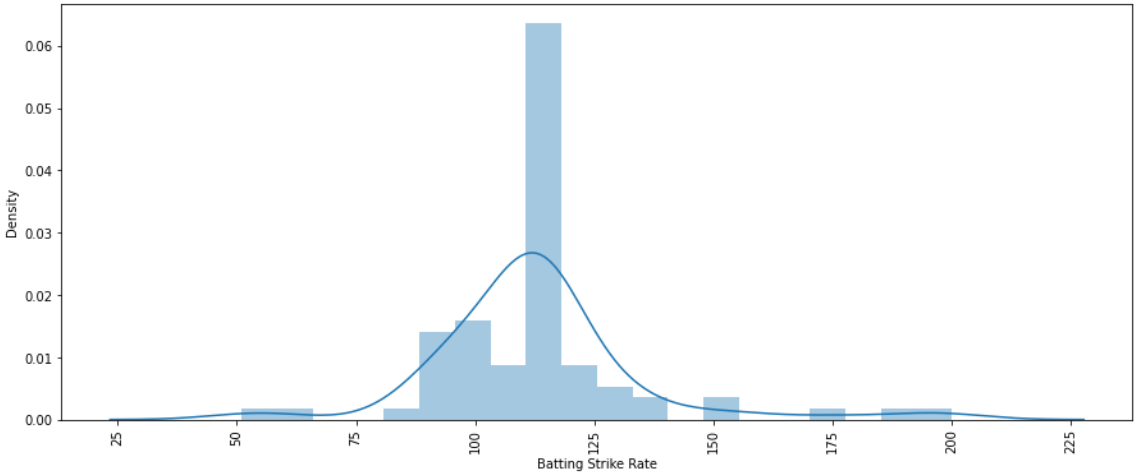
In [25]:

```
for i in numerical_columns:
    plt.figure(figsize=(15, 6))
    sns.distplot(df[i], kde = True, bins = 20)
    plt.xticks(rotation=90)
    plt.show()
```



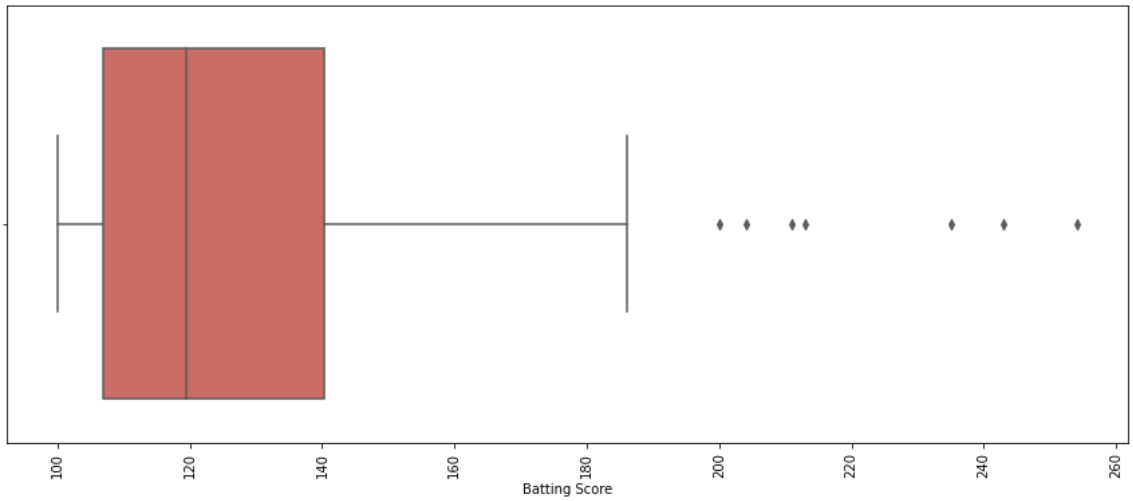


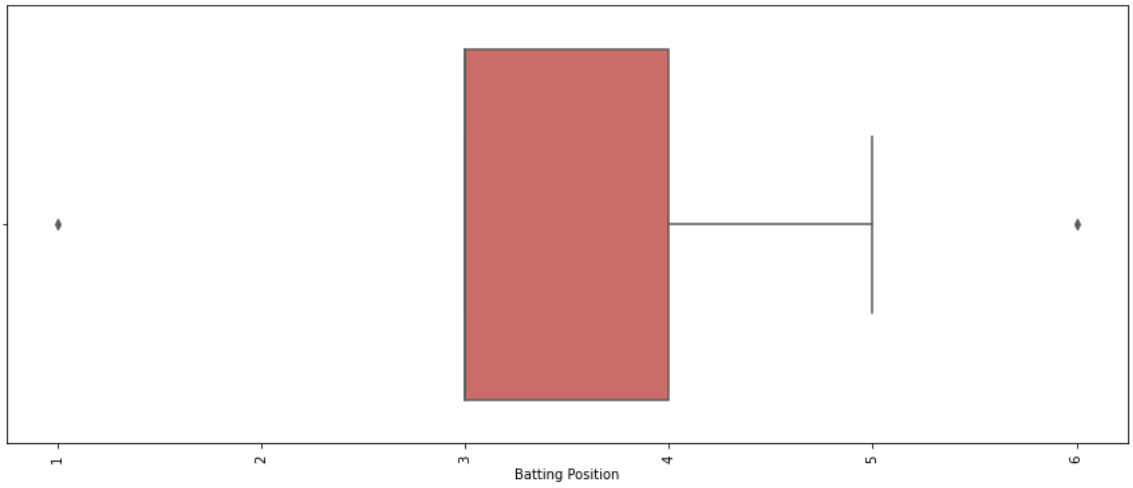


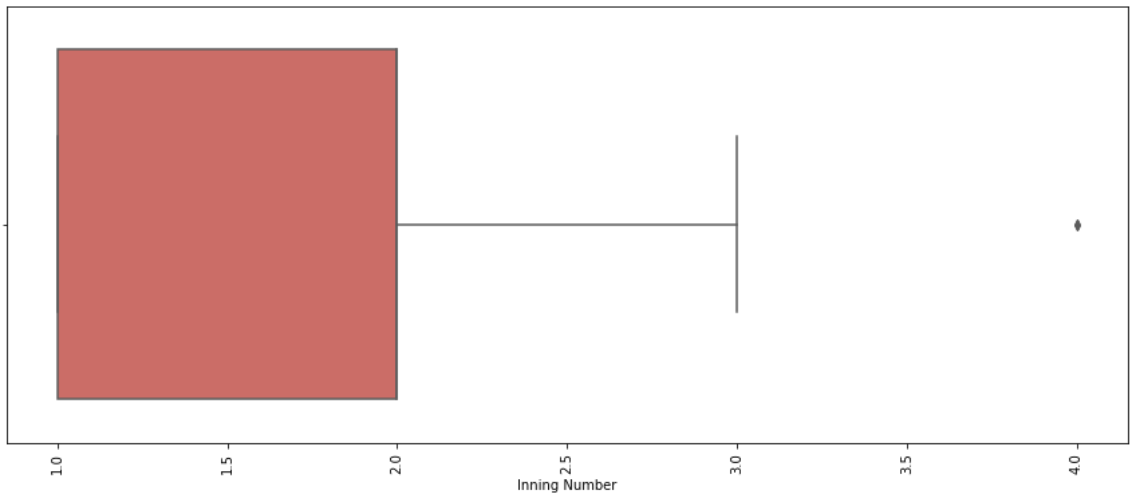

Density
0.06
0.05
0.04
0.03
0.02
0.01
0.00
25
50
75
100
125
150
175
200
225
Batting Strike Rate

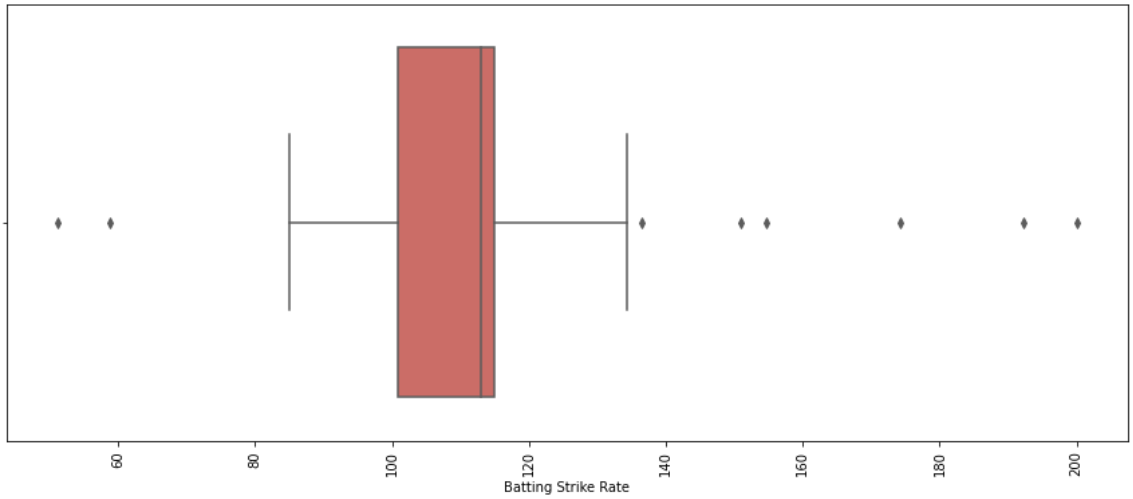
In [26]:

```
for i in numerical_columns:
    plt.figure(figsize=(15, 6))
    sns.boxplot(df[i], data = df, palette='hls')
    plt.xticks(rotation=90)
    plt.show()
```



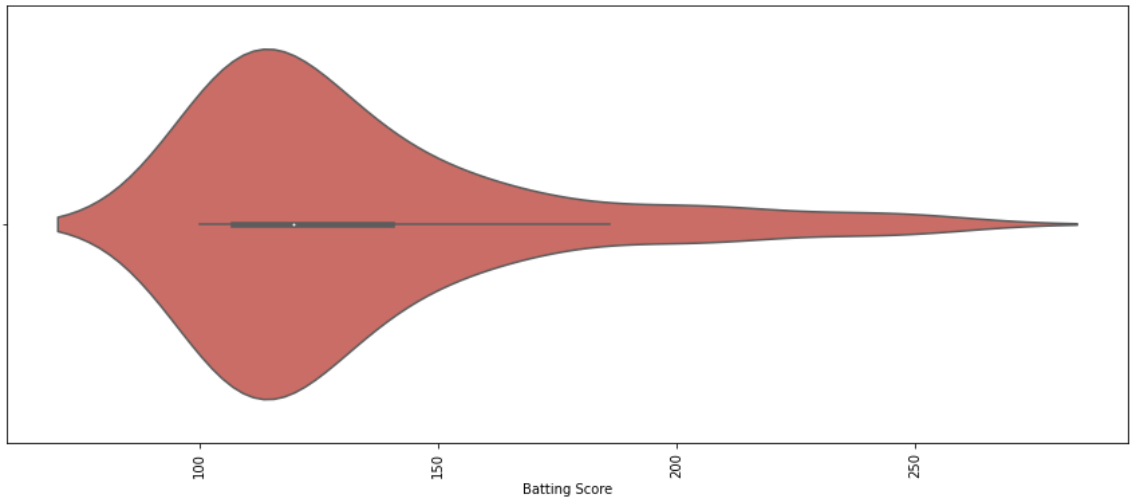


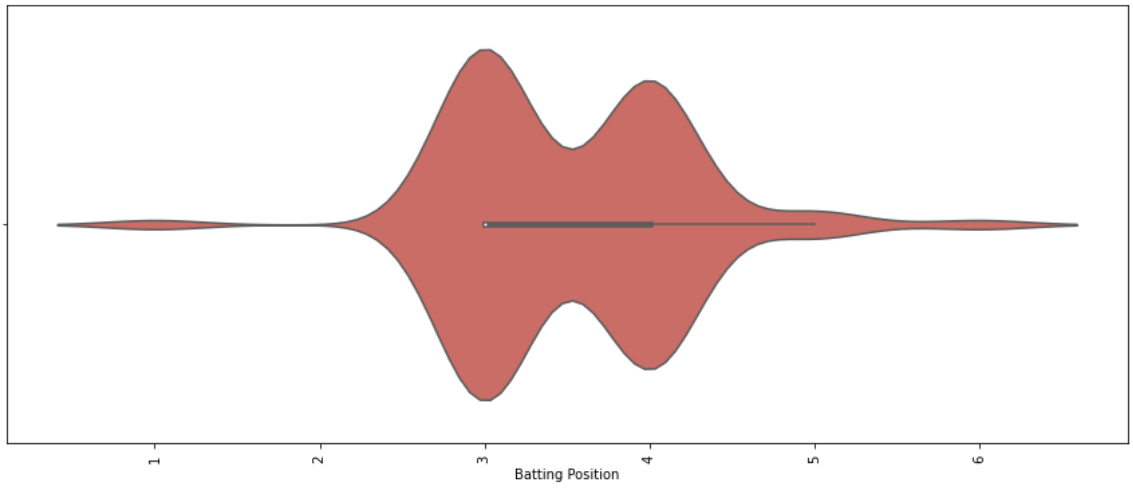


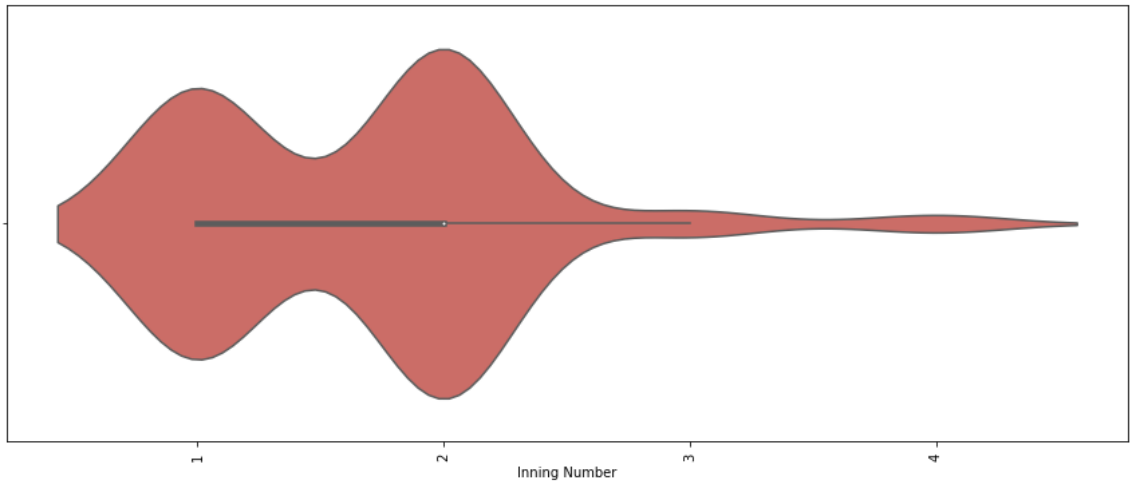

60
80
100
120
140
160
180
200
Batting Strike Rate

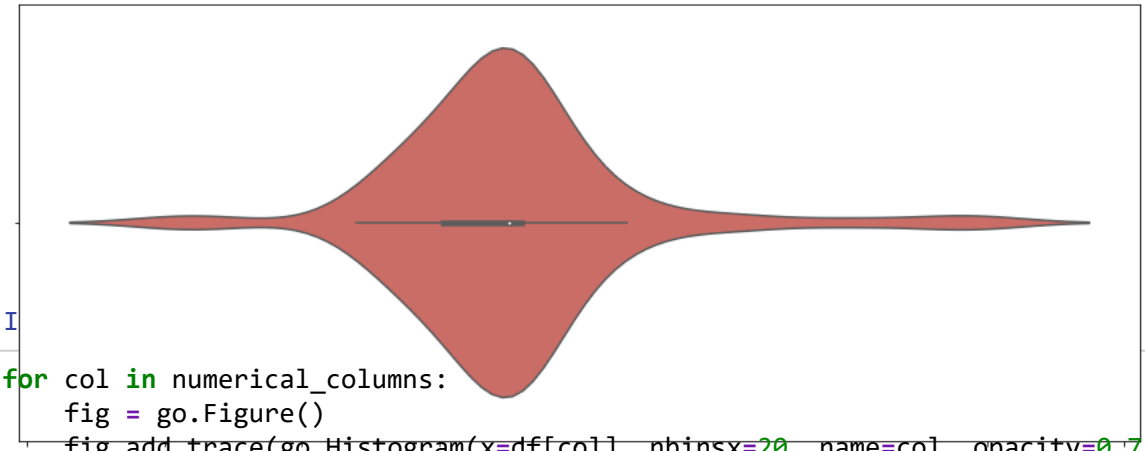
In [27]:

```
for i in numerical_columns:
    plt.figure(figsize=(15, 6))
    sns.violinplot(df[i], data = df, palette='hls')
    plt.xticks(rotation=90)
    plt.show()
```





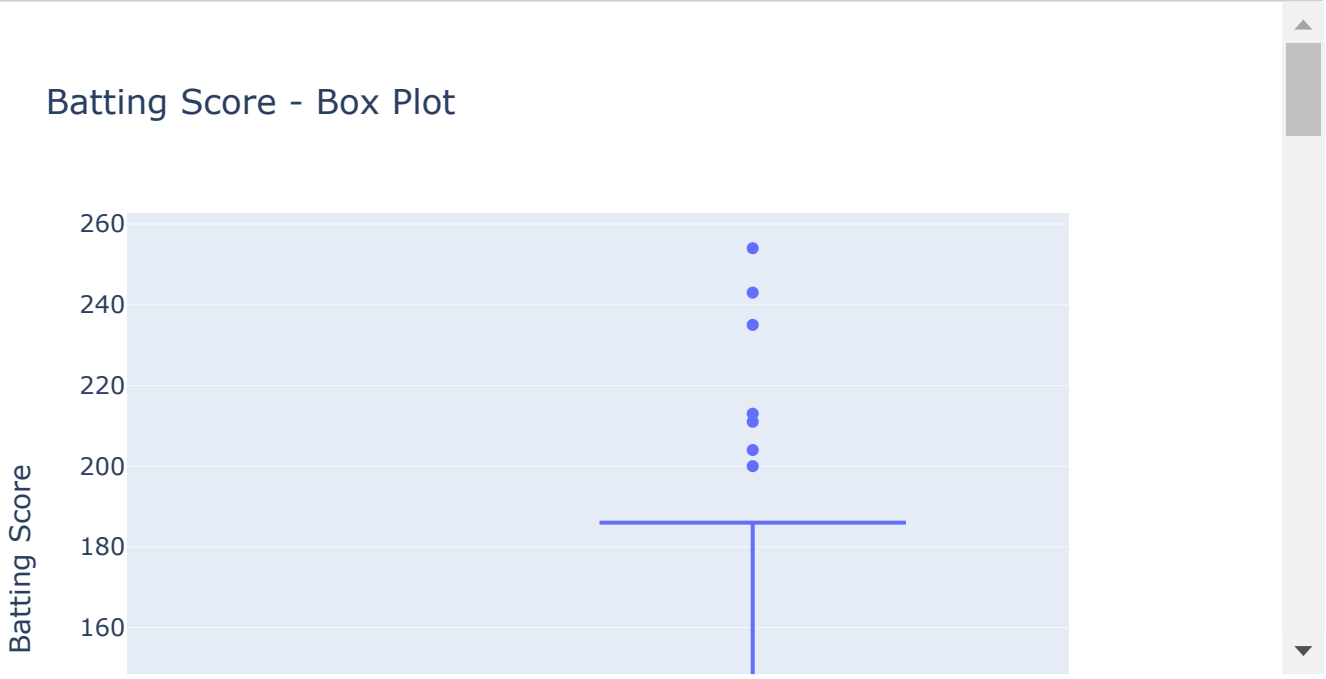





```
for col in numerical_columns:
    fig = go.Figure()
    fig.add_trace(go.Histogram(x=df[col], nbinsx=20, name=col, opacity=0.7))
    fig.update_layout(title=f'{col} - Histogram with Bar Lines', xaxis_title=col, yaxis_
    fig.show()
```

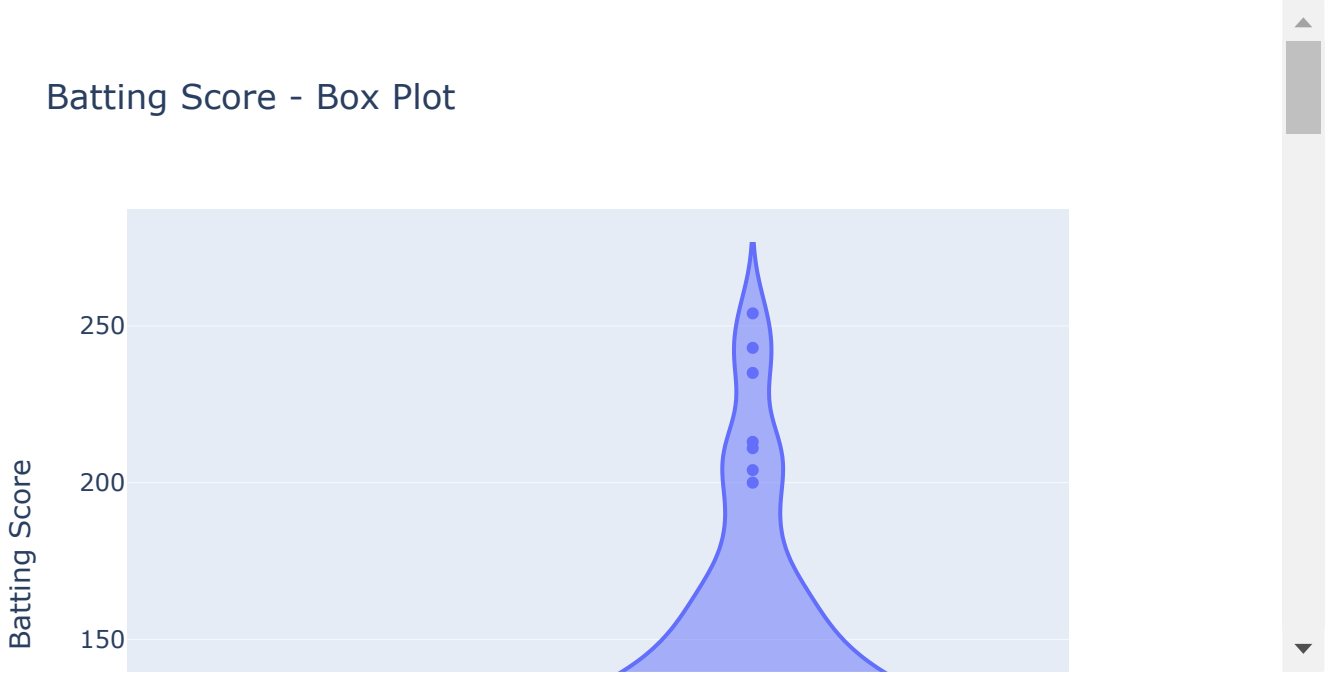
In [29]:

```
for col in numerical_columns:
    fig = px.box(df, y=col, title=f'{col} - Box Plot')
    fig.show()
```



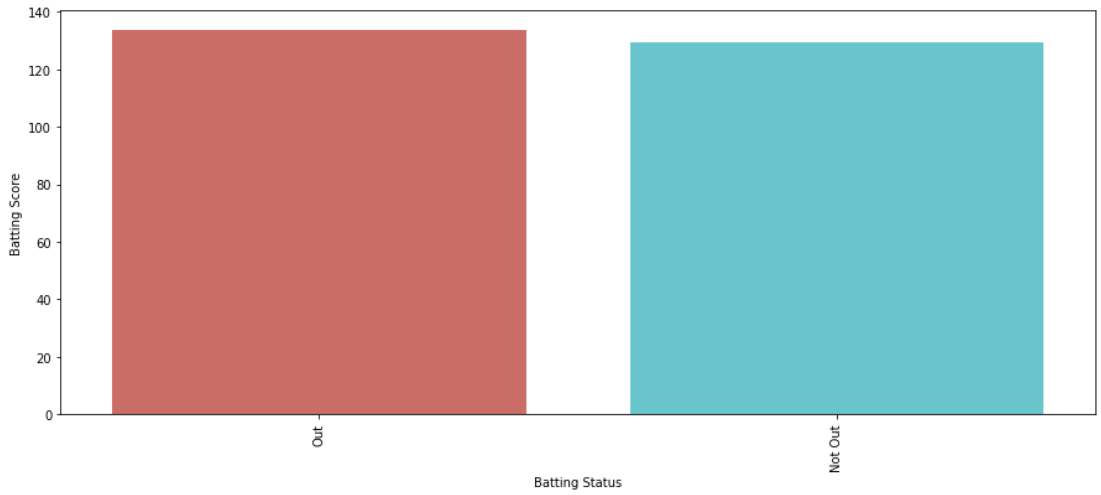
In [30]:

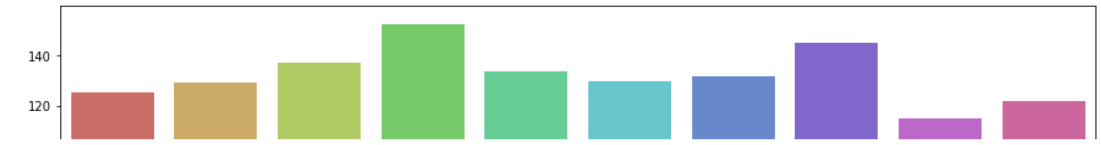
```
for col in numerical_columns:
    fig = px.violin(df, y=col, title=f'{col} - Box Plot')
    fig.show()
```



In [31]:

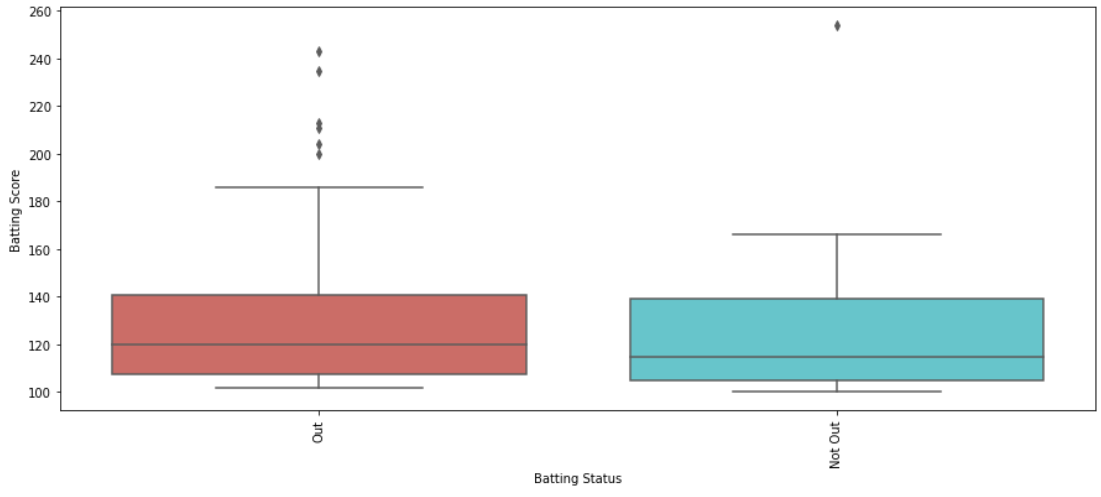
```
for i in numerical_columns:
    for j in categorical_columns:
        if j != 'Match Date':
            plt.figure(figsize=(15, 6))
            sns.barplot(x = df[j], y = df[i], ci = None, data = df, palette='hls')
            plt.xticks(rotation=90)
            plt.show()
```

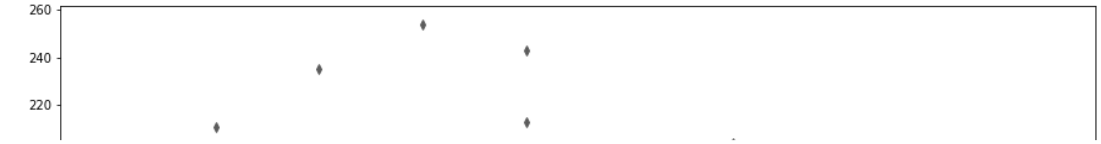




In [32]:

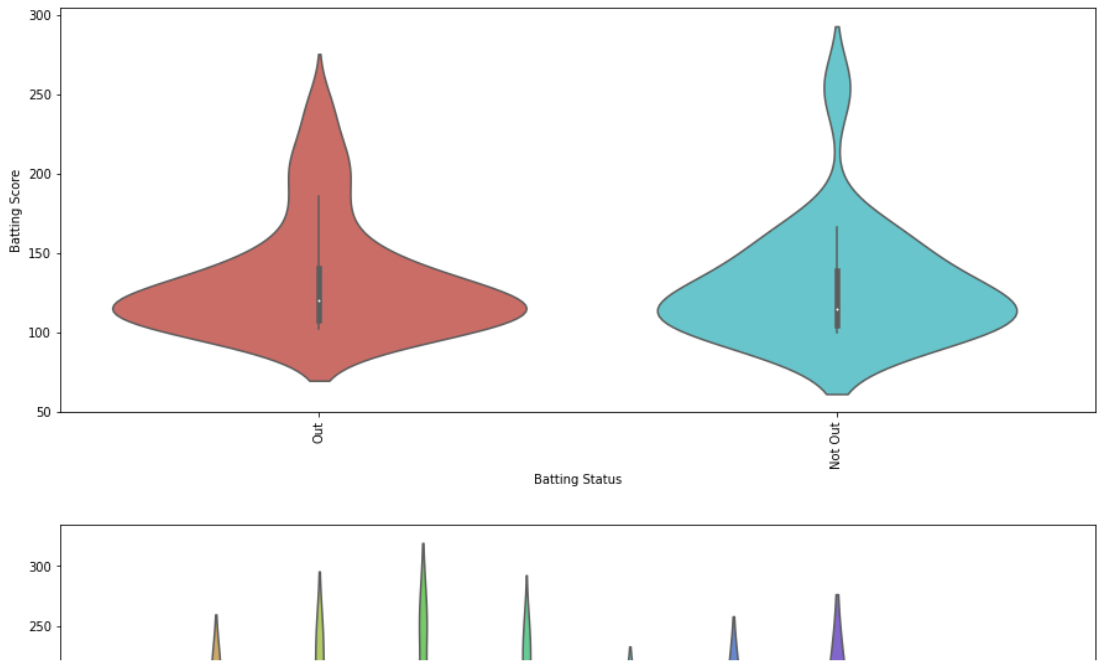
```
for i in numerical_columns:
    for j in categorical_columns:
        if j != 'Match Date':
            plt.figure(figsize=(15, 6))
            sns.boxplot(x = df[j], y = df[i], data = df, palette='hls')
            plt.xticks(rotation=90)
            plt.show()
```





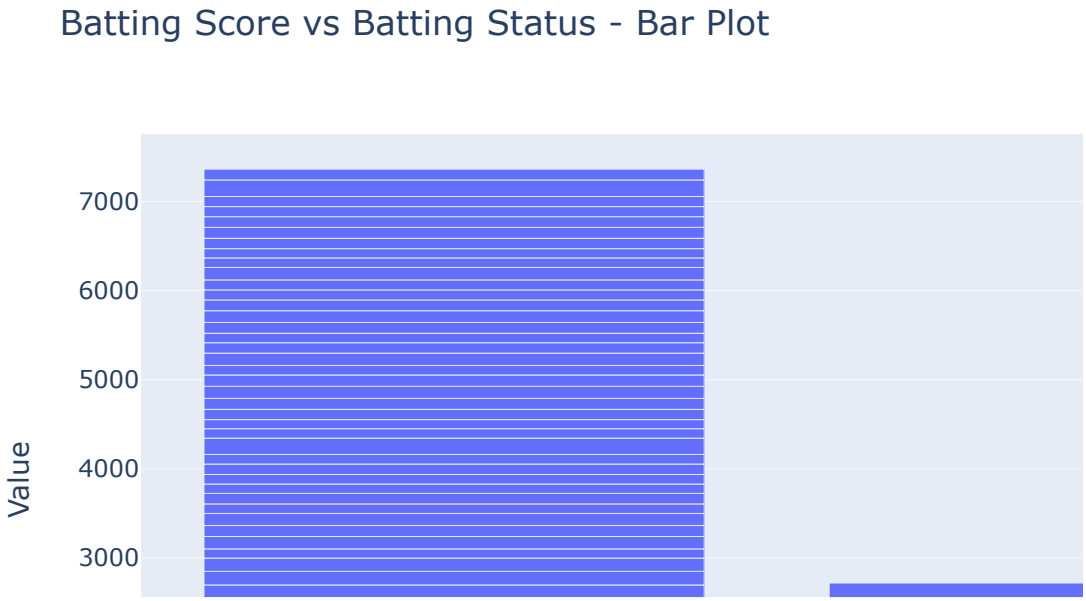
In [33]:

```
for i in numerical_columns:
    for j in categorical_columns:
        if j != 'Match Date':
            plt.figure(figsize=(15, 6))
            sns.violinplot(x = df[j], y = df[i], data = df, palette='hls')
            plt.xticks(rotation=90)
            plt.show()
```



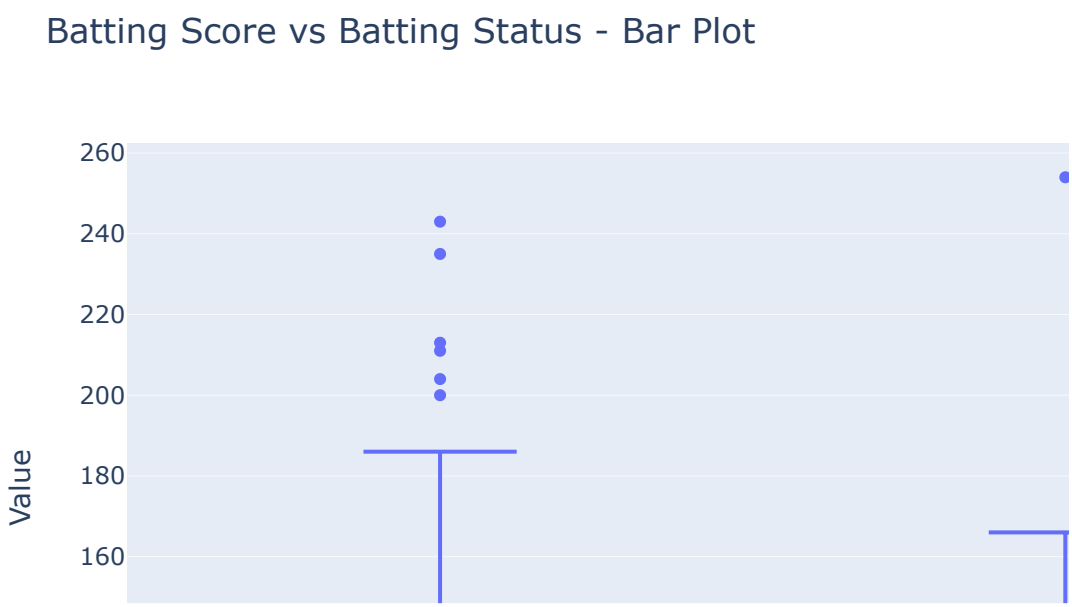
In [34]:

```
for i in numerical_columns:
    for j in categorical_columns:
        fig = px.bar(df, x=j, y=i, title=f'{i} vs {j} - Bar Plot', labels={j: 'Category'
        fig.show()
```



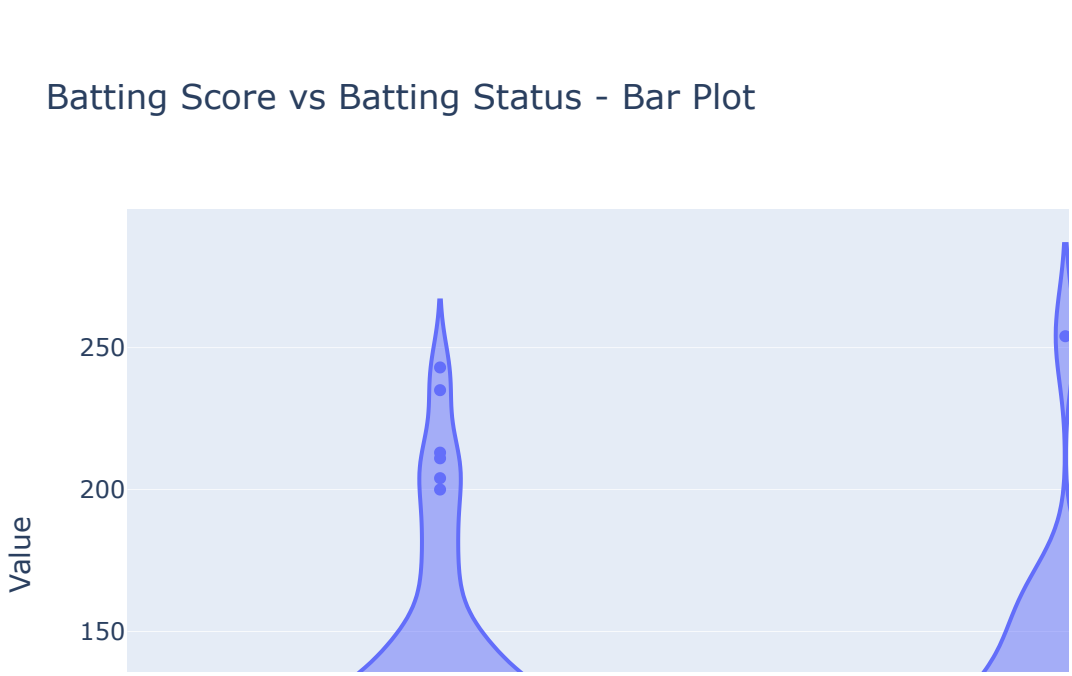
In [35]:

```
for i in numerical_columns:
    for j in categorical_columns:
        fig = px.box(df, x=j, y=i, title=f'{i} vs {j} - Bar Plot', labels={j: 'Category'
        fig.show()
```



In [36]:

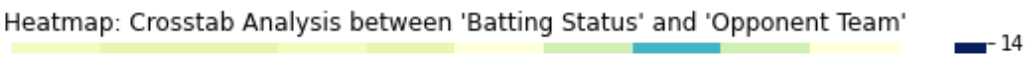
```
for i in numerical_columns:
    for j in categorical_columns:
        fig = px.violin(df, x=j, y=i, title=f'{i} vs {j} - Bar Plot', labels={j: 'Catego
        fig.show()
```



In [37]:

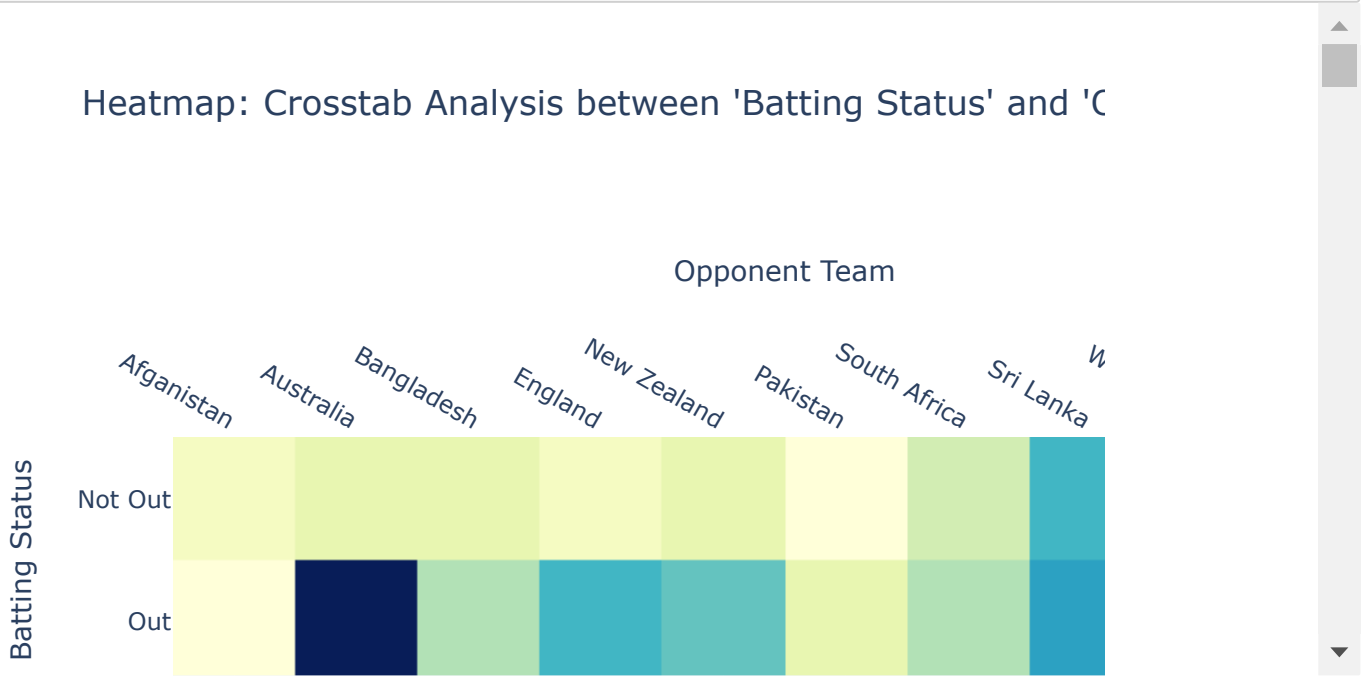
```
for col1 in categorical_columns:
    for col2 in categorical_columns:
        if col1 != col2:
            crosstab_result = pd.crosstab(df[col1], df[col2])
            print(f"Crosstab Analysis between '{col1}' and '{col2}':\n")
            print(crosstab_result)
            print("\n")

            plt.figure(figsize=(10, 6))
            sns.heatmap(crosstab_result, annot=True, cmap="YlGnBu", fmt='d')
            plt.title(f"Heatmap: Crosstab Analysis between '{col1}' and '{col2}'")
            plt.xlabel(col2)
            plt.ylabel(col1)
            plt.show()
```

```
Crosstab Analysis between 'Batting Status' and 'Opponent Team':

Opponent Team   Afganistan  Australia  Bangladesh  England  New Zealand
\
Batting Status
Not Out                  1          2           2        1            2
Out                      0         14           4        7            6

Opponent Team   Pakistan  South Africa  Sri Lanka  West Indies  Zimbabw
e
Batting Status
Not Out                0             3          7            3
0
Out                    2             4          8            9
1
```



In [38]:

```
for col1 in categorical_columns:
    for col2 in categorical_columns:
        if col1 != col2:
            crosstab_result = pd.crosstab(df[col1], df[col2])

            fig = px.imshow(crosstab_result,
                            x=crosstab_result.columns,
                            y=crosstab_result.index,
                            color_continuous_scale="YlGnBu",
                            title=f"Heatmap: Crosstab Analysis between '{col1}' and '{co
            fig.update_xaxes(side="top")
            fig.show()
```



In [39]:

```
df['Match Date'] = pd.to_datetime(df['Match Date'])
```

In [40]:

```
df['Year'] = df['Match Date'].dt.year
df['Month'] = df['Match Date'].dt.month_name()
```

In [41]:

```
matches_per_year = df.groupby('Year').size().reset_index(name='Matches')
matches_per_month = df.groupby(['Year', 'Month']).size().reset_index(name='Matches')
```

In [42]:

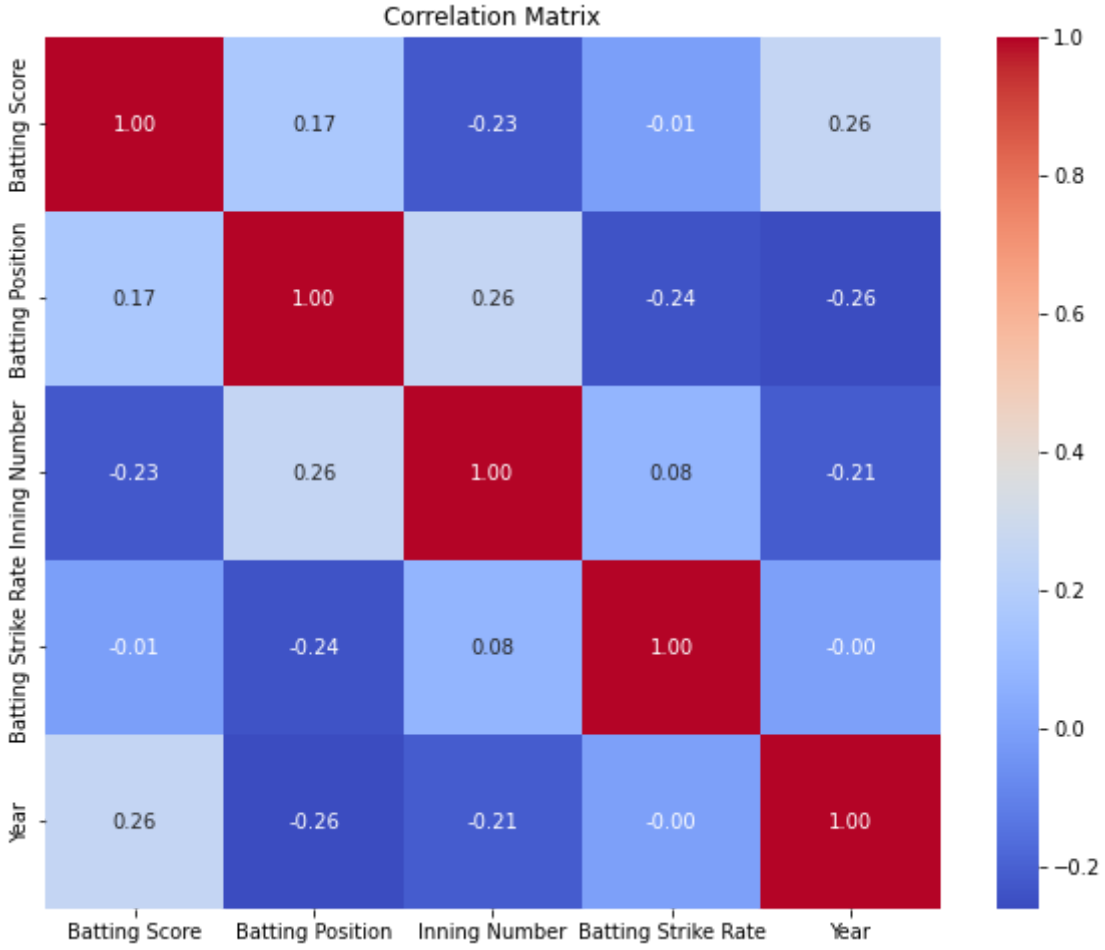
```
fig_year = px.line(matches_per_year, x='Year', y='Matches', title='Matches per Year')
fig_year.show()
```

In [43]:

```
for year in df['Year'].unique():
    matches_year = matches_per_month[matches_per_month['Year'] == year]
    fig = px.line(matches_year, x='Month', y='Matches', title=f'Matches per Month ({year
    fig.update_xaxes(type='category', categoryorder='category ascending')
    fig.show()
```

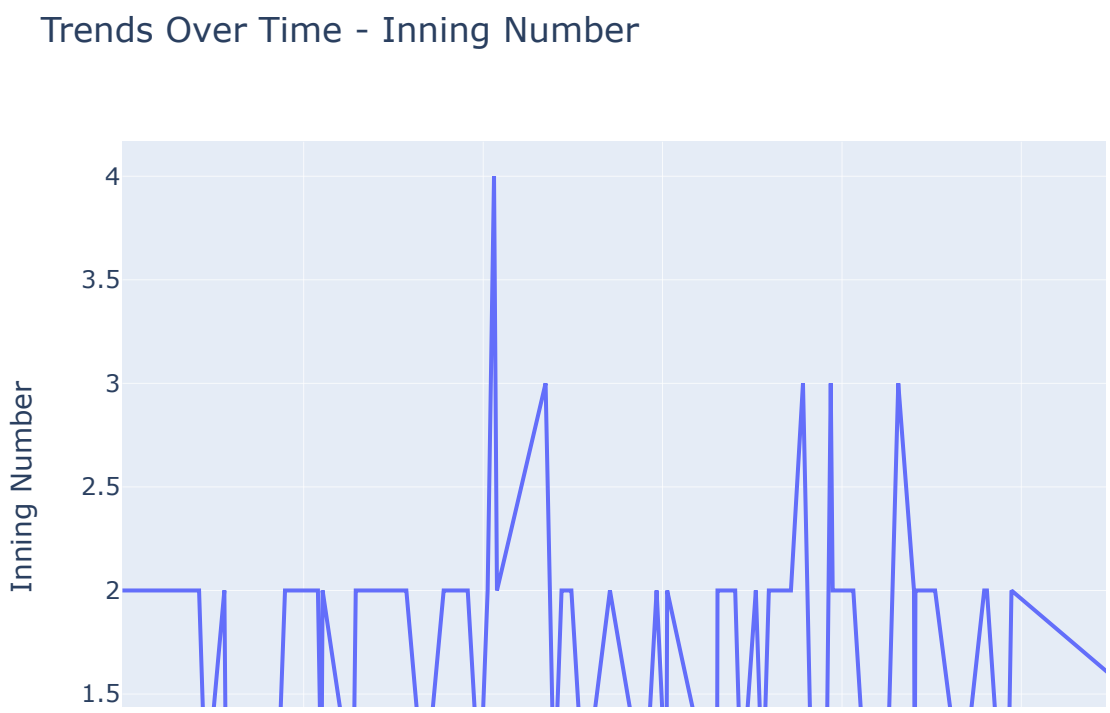
In [44]:

```
corr_matrix = df.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
```



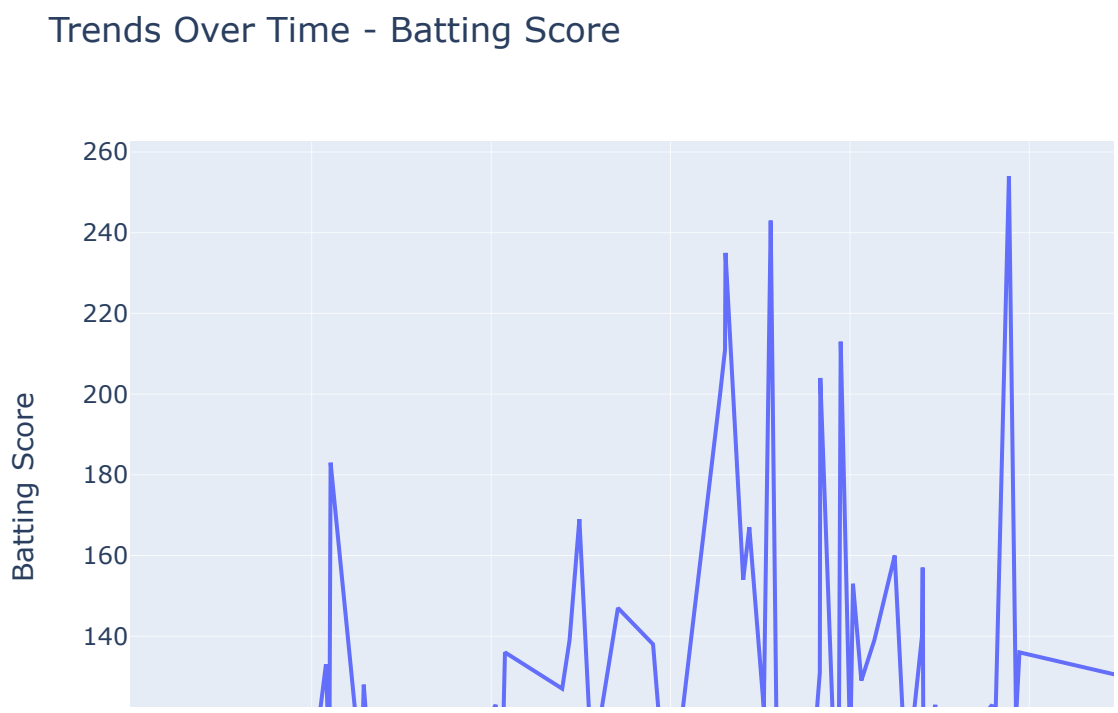
In [45]:

```
inning_trends = df.groupby('Match Date')['Inning Number'].mean().reset_index()
fig_inning = px.line(inning_trends, x='Match Date', y='Inning Number', title='Trends Ove
fig_inning.show()
```



In [46]:

```
score_trends = df.groupby('Match Date')['Batting Score'].mean().reset_index()
fig_score = px.line(score_trends, x='Match Date', y='Batting Score', title='Trends Over
fig_score.show()
```



Virat Kohli's centuries are not just numbers on a scoreboard; they embody dedication, skill, and a burning passion for the sport. Through our Virat Kohli Centuries Analysis, we've endeavored to unravel the magic behind these centuries – the countless hours of practice, the strategic acumen, and the unyielding determination that define this cricketing icon