Project Title: FIFA World Cup Analysis

INTRODUCTION

Welcome to the FIFA World Cup Analysis project, where we uncover the hidden narratives behind football's greatest spectacle. This endeavor delves into the meticulous work of unsung analysts, utilizing tools like Python to decipher the key metrics and influences shaping the outcomes of the world's most prestigious tournament. Join us on this exciting journey as we unravel the captivating story behind every kick, goal, and triumph in the realm of global football.

Import necessary libraries

In [1]:
```
# Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
```

Loading Dataset

In [2]:
```
Matches = pd.read_csv('C:\\Users\\stati\\OneDrive\\Desktop\WorldCupMatches.csv')
pd.concat([Matches.head(2), Matches.tail(2)])
```

Out[2]:

	Year	Datetime	Stage	Stadium	City	Home Team Name	Home Team Goals	Away Team Goals	Away Team Name	Win conditions	Attendance	Half-time Home Goals	Half-time Away Goals	Referee	Assistant 1	Assistant 2	RoundID	MatchID	Home Team Initials	Away Team Initials
0	1930	13 Jul 1930 - 15:00	Group 1	Pocitos	Montevideo	France	4	1	Mexico		4444.0	3	0	LOMBARDI Domingo (URU)	CRISTOPHE Henry (BEL)	REGO Gilberto (BRA)	201	1096	FRA	MEX
1	1930	13 Jul 1930 - 15:00	Group 4	Parque Central	Montevideo	USA	3	0	Belgium		18346.0	2	0	MACIAS Jose (ARG)	MATEUCCI Francisco (URU)	WARNKEN Alberto (CHI)	201	1090	USA	BEL
850	2014	12 Jul 2014 - 17:00	Play-off for third place	Estadio Nacional	Brasilia	Brazil	0	3	Netherlands		68034.0	0	2	HAIMOUDI Djamel (ALG)	ACHIK Redouane (MAR)	ETCHIALI Abdelhak (ALG)	255957	300186502	BRA	NED
851	2014	13 Jul 2014 - 16:00	Final	Estadio do Maracana	Rio De Janeiro	Germany	1	0	Argentina	Germany win after extra time	74738.0	0	0	Nicola RIZZOLI (ITA)	Renato FAVERANI (ITA)	Andrea STEFANI (ITA)	255959	300186501	GER	ARG

In [3]:
```
Players = pd.read_csv('C:\\Users\\stati\\OneDrive\\Desktop\\WorldCupPlayers.csv')
pd.concat([Players.head(2), Players.tail(2)])
```

Out[3]:

	RoundID	MatchID	Team Initials	Coach Name	Line-up	Shirt Number	Player Name	Position	Event
0	201	1096	FRA	CAUDRON Raoul (FRA)	S	0	Alex THEPOT	GK	NaN
1	201	1096	MEX	LUQUE Juan (MEX)	S	0	Oscar BONFIGLIO	GK	NaN
37782	255959	300186501	GER	LOEW Joachim (GER)	N	21	MUSTAFI	NaN	NaN
37783	255959	300186501	ARG	SABELLA Alejandro (ARG)	N	23	BASANTA	NaN	NaN

In [4]:
```
World_cup = pd.read_csv('C:\\Users\\stati\\OneDrive\\Desktop\\WorldCups.csv')
pd.concat([World_cup.head(2), World_cup.tail(2)])
```

Out[4]:

	Year	Country	Winner	Runners-Up	Third	Fourth	GoalsScored	QualifiedTeams	MatchesPlayed	Attendance
0	1930	Uruguay	Uruguay	Argentina	USA	Yugoslavia	70	13	18	590.549
1	1934	Italy	Italy	Czechoslovakia	Germany	Austria	70	16	17	363
18	2010	South Africa	Spain	Netherlands	Germany	Uruguay	145	32	64	3.178.856
19	2014	Brazil	Germany	Argentina	Netherlands	Brazil	171	32	64	3.386.810

EDA(Exploratry Data Analysis)

Data cleaning and processing

I would like to consolidate the columns for both old and new Germany under a single name. Additionally, I intend to change the data type of the 'Attendance' column to integer.

In [5]:
```
Matches.dropna(subset=['Year'], inplace=True)
```

In [6]:
```
Matches['Home Team Name'].value_counts()
```

Out[6]:
```
Home Team Name
Brazil                          82
Italy                           57
Argentina                       54
Germany FR                      43
England                         35
                                ..
Wales                            1
Norway                           1
rn">United Arab Emirates         1
Haiti                            1
rn">Bosnia and Herzegovina       1
Name: count, Length: 78, dtype: int64
```

In [7]:
```
names = Matches[Matches['Home Team Name'].str.contains('rn">')]['Home Team Name'].value_counts()
names
```

Out[7]:
```
Home Team Name
rn">Republic of Ireland       5
rn">United Arab Emirates      1
rn">Trinidad and Tobago       1
rn">Serbia and Montenegro     1
rn">Bosnia and Herzegovina    1
Name: count, dtype: int64
```

In [8]:
```
wrong = list(names.index)
wrong
```

Out[8]:
```
['rn">Republic of Ireland',
 'rn">United Arab Emirates',
 'rn">Trinidad and Tobago',
 'rn">Serbia and Montenegro',
 'rn">Bosnia and Herzegovina']
```

In [9]:
```
correct = [name.split('>')[1] for name in wrong]
correct
```

Out[9]:
```
['Republic of Ireland',
 'United Arab Emirates',
 'Trinidad and Tobago',
 'Serbia and Montenegro',
 'Bosnia and Herzegovina']
```

In [10]:
```
# Standardizing team names
team_name_mapping = {
    'Germany FR': 'Germany',
    'Maracan\x00 - Est\x00dio Jornalista M\x00rio Filho': 'Maracan Stadium',
    'Estadio do Maracana': 'Maracan Stadium'
}

for wrong, correct in team_name_mapping.items():
    World_cup.replace(wrong, correct, inplace=True)
    Matches.replace(wrong, correct, inplace=True)
    Players.replace(wrong, correct, inplace=True)
```

In [11]:
```
for wrong, correct in team_name_mapping.items():
    World_cup.replace(wrong, correct, inplace=True)
    Matches.replace(wrong, correct, inplace=True)
    Players.replace(wrong, correct, inplace=True)
```

In [12]:
```
names = Matches[Matches['Home Team Name'].str.contains('rn">')]['Home Team Name'].value_counts()
names
```

Out[12]:
```
Home Team Name
rn">Republic of Ireland       5
rn">United Arab Emirates      1
rn">Trinidad and Tobago       1
rn">Serbia and Montenegro     1
rn">Bosnia and Herzegovina    1
Name: count, dtype: int64
```

In [13]:
```
winner = World_cup['Winner'].value_counts()
winner
```

Out[13]:
```
Winner
Brazil       5
Italy        4
Germany      4
Uruguay      2
Argentina    2
England      1
France       1
Spain        1
Name: count, dtype: int64
```

In [14]:
```
runnerup = World_cup['Runners-Up'].value_counts()
runnerup
```

Out[14]:
```
Runners-Up
Germany           4
Argentina         3
Netherlands       3
Czechoslovakia    2
Hungary           2
Brazil            2
Italy             2
Sweden            1
France            1
Name: count, dtype: int64
```

In [15]:
```
third = World_cup['Third'].value_counts()
third
```

Out[15]:
```
Third
Germany        4
Brazil         2
Sweden         2
France         2
Poland         2
USA            1
Austria        1
Chile          1
Portugal       1
Italy          1
Croatia        1
Turkey         1
Netherlands    1
Name: count, dtype: int64
```

In [16]:
```
teams = pd.concat([winner, runnerup, third], axis=1)
teams.fillna(0, inplace=True)
teams = teams.astype(int)
teams
```

Out[16]:

	count	count	count
Brazil	5	2	2
Italy	4	2	1
Germany	4	4	4
Uruguay	2	0	0
Argentina	2	3	0
England	1	0	0
France	1	1	2
Spain	1	0	0
Netherlands	0	3	1
Czechoslovakia	0	2	0
Hungary	0	2	0
Sweden	0	1	2
Poland	0	0	2
USA	0	0	1
Austria	0	0	1
Chile	0	0	1
Portugal	0	0	1
Croatia	0	0	1
Turkey	0	0	1

In [17]:
```
# Assuming 'World_cup' is your DataFrame
World_cup['Attendance'] = World_cup['Attendance'].astype(str)
World_cup['Attendance'] = World_cup['Attendance'].str.replace('.', '')
World_cup['Attendance'] = pd.to_numeric(World_cup['Attendance'], errors='coerce', downcast='integer')

# Display the resulting DataFrame
print(World_cup[['Year', 'Attendance']])
```

```
    Year  Attendance
0   1930      590549
1   1934         363
2   1938        3757
3   1950     1045246
4   1954      768607
5   1958       81981
6   1962      893172
7   1966     1563135
8   1970     1603975
9   1974     1865753
10  1978     1545791
11  1982     2109723
12  1986     2394031
13  1990     2516215
14  1994     3587538
15  1998     2785100
16  2002     2705197
17  2006     3359439
18  2010     3178856
19  2014     3386810
```

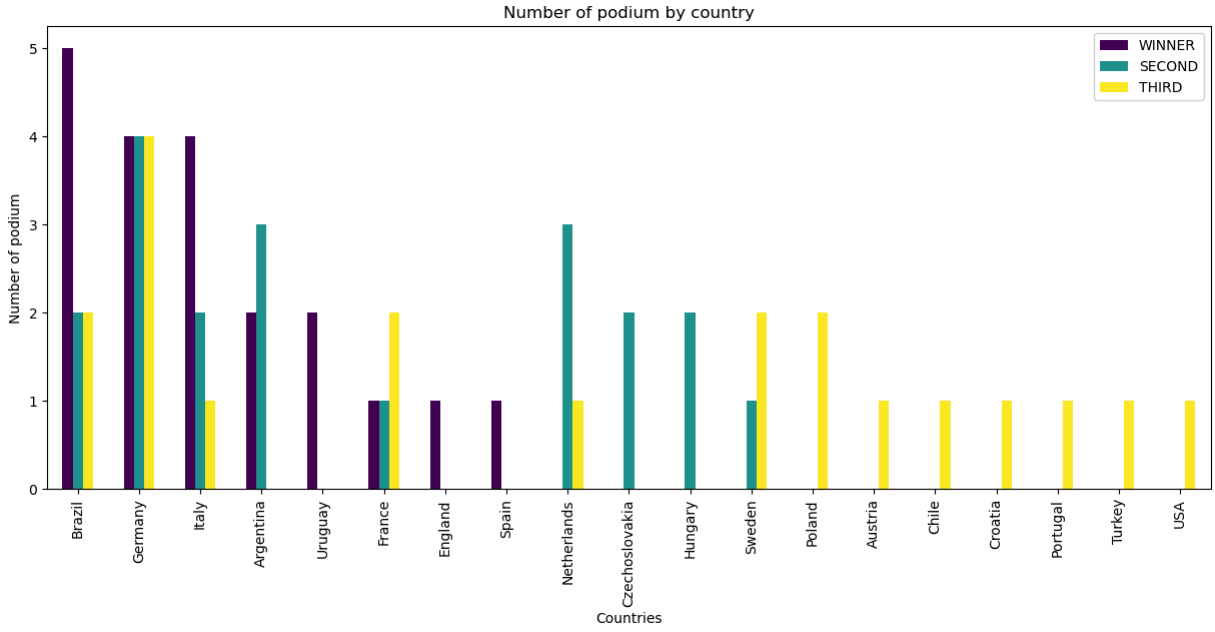
Vizualization

Countries That Won the Cup

In [18]:
```
# Visualization of countries that won, came second, and came third

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

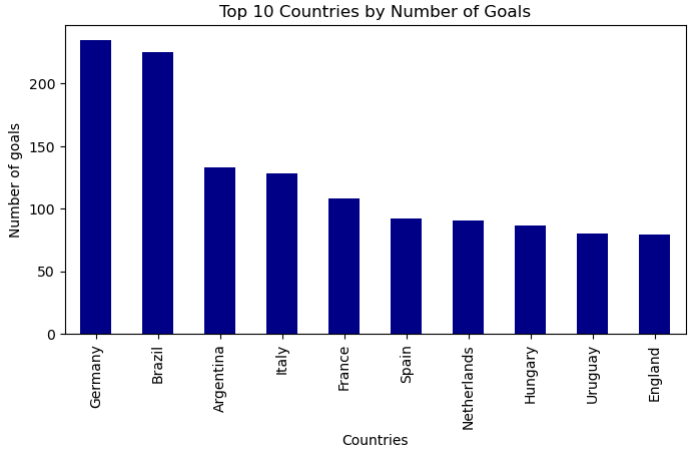
In [19]:
```
# Assuming 'world_cup' is DataFrame with World Cup data
# Replace 'Winner', 'Runners-Up', and 'Third' with actual column names

(pd.DataFrame({
    'WINNER': World_cup['Winner'].value_counts(),
    'SECOND': World_cup['Runners-Up'].value_counts(),
    'THIRD': World_cup['Third'].value_counts()
}).fillna(0).astype('int64').sort_values(by=['WINNER', 'SECOND', 'THIRD'], ascending=False)
  .plot(y=['WINNER', 'SECOND', 'THIRD'], kind="bar",
        colormap='viridis', figsize=(15, 6),
        title='Number of podium by country')).set(xlabel='Countries', ylabel='Number of podium')
```

Out[19]:
```
[Text(0.5, 0, 'Countries'), Text(0, 0.5, 'Number of podium')]
```



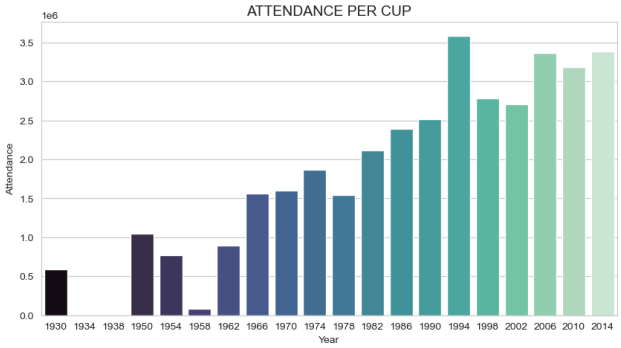
Number of Goals Per Country

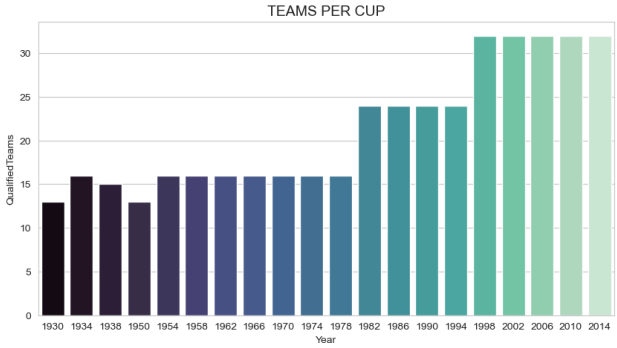
In [20]:
```
# Assuming 'Matches' is your DataFrame with FIFA World Cup match data
goal_per_country = pd.concat([Matches[[
    'Home Team Name', 'Home Team Goals']].dropna().rename(columns={
    'Home Team Name': 'countries', 'Home Team Goals': 'goals'}),
                              Matches[[
                                  'Away Team Name', 'Away Team Goals']].dropna().rename(columns={
                                  'Away Team Name': 'countries', 'Away Team Goals': 'goals'})])

(goal_per_country.groupby('countries')['goals'].sum().nlargest(10)
 .plot(kind='bar', color='darkblue', figsize=(8, 4),
       fontsize=10, title='Top 10 Countries by Number of Goals'))

plt.xlabel('Countries')
plt.ylabel('Number of goals')
plt.show()
```

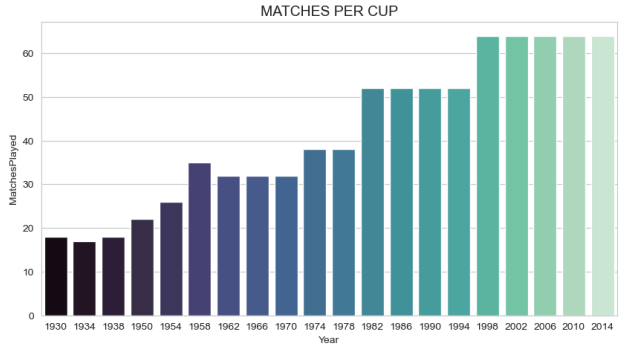


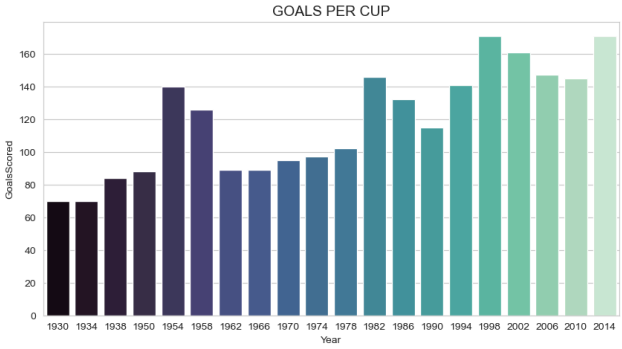
Cup Statistics Over the Years

In [21]:
```
plt.figure(figsize=(22, 12))
sns.set_style("whitegrid")

plots = ["Attendance", "QualifiedTeams", "MatchesPlayed", "GoalsScored"]
titles = ["ATTENDANCE", "TEAMS", "MATCHES", "GOALS"]

for i, (plot, title) in enumerate(zip(plots, titles), 1):
    plt.subplot(2, 2, i)
    sns.barplot(x="Year", y=plot, data=World_cup, palette="mako")
    plt.title(f"{title} PER CUP", fontsize=14)

plt.subplots_adjust(wspace=0.2, hspace=0.4, top=0.9)
plt.show()
```









Teams with Most Goals per Cup

I aim to analyze the goal-scoring patterns of teams in each World Cup. To achieve this, I'll create a new dataset derived from the world_cups_matches set. The process involves a kind of "map-reduce" operation:

Step 1 - Extract the year, home team name, and home team goals, then sum the goals per year and team name.

Step 2 - Repeat the same operation as in step 1, but with the away team.

Step 3 - Join the two datasets based on team name and year.

T his will enable me to examine the number of goals per team per cup and aggregate the results across all the cups.

In [22]:
```
Home_goals = Matches.groupby(['Year', 'Home Team Name'])['Home Team Goals'].sum()
```

In [23]:
```
Away_goals = Matches.groupby(['Year', 'Away Team Name'])['Away Team Goals'].sum()
```

In [24]:
```
# Assuming 'Home_goals' and 'Away_goals' are DataFrames
goals = pd.concat([Home_goals, Away_goals], axis=1).fillna(0)
goals['Goals'] = goals['Home Team Goals'] + goals['Away Team Goals']
goals = goals.drop(['Home Team Goals', 'Away Team Goals'], axis=1)
```

In [25]:
```
goals = goals.reset_index()
```

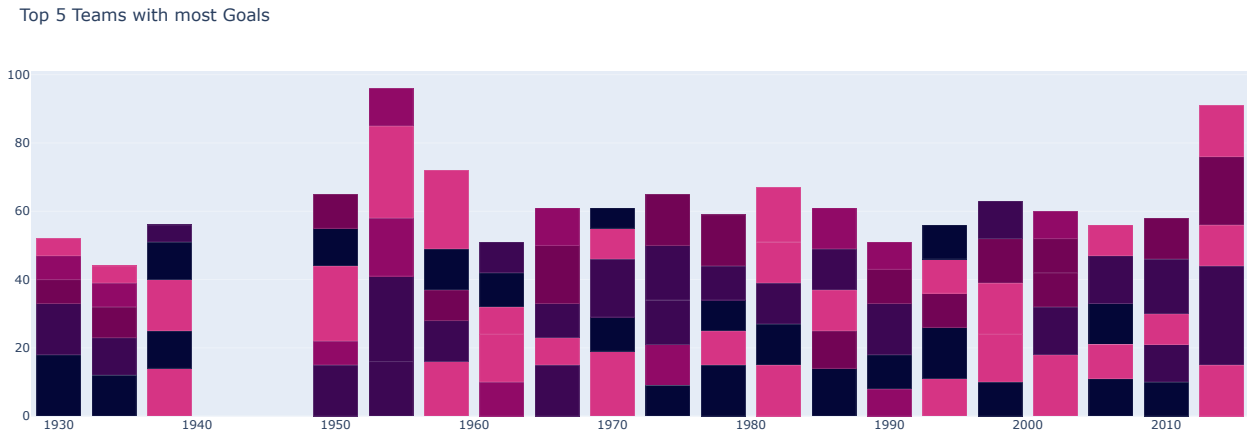
In [26]:
```
# Assuming 'goals' is a DataFrame
goals.columns = ['Year', 'Country', 'Goals']
goals = goals.sort_values(by=['Year', 'Goals'], ascending=[True, False])
```

In [27]:
```
Top5 = goals.groupby('Year').head()
```

In [28]:
```
import plotly.graph_objects as go
```

In [29]:
```
x, y = goals['Year'].values, goals['Goals'].values
```

In [30]:
```
import plotly.graph_objects as go

# Ensure the number of colors matches the number of unique teams
colors = ['#030637', '#3C0753', '#720455', '#910A67', '#D63484']

data = [go.Bar(x=Top5[Top5['Country'] == team]['Year'],
               y=Top5[Top5['Country'] == team]['Goals'],
               name=team,
               marker_color=colors[i % len(colors)])
        for i, team in enumerate(Top5['Country'].drop_duplicates().values)]

layout = go.Layout(barmode='stack', title='Top 5 Teams with most Goals', showlegend=False)
fig = go.Figure(data=data, layout=layout)
fig.show()
```



Additional Analysis

Winning Teams Word Cloud

To generate a word cloud of teams with the most wins, I have augmented the Matches dataset by introducing three additional columns:

result: Indicates whether there is a winner or if the result is a draw.

winner: Specifies the team that emerged victorious in a match.

looser: Identifies the team that suffered defeat in a match.

These columns are instrumental in capturing the outcomes of each match, enabling subsequent analysis and visualization to highlight teams with the most wins.

In [31]:
```
winner_home = Matches['Home Team Goals'] > Matches['Away Team Goals']
winner_away = Matches['Home Team Goals'] < Matches['Away Team Goals']
win_penalties = Matches['Win conditions'].str.len() > 1

Matches['result'] = np.where(winner_home | winner_away | win_penalties, 'win', 'draw')
Matches['Winner'] = np.where(winner_home, Matches['Home Team Name'],
                             np.where(winner_away, Matches['Away Team Name'],
                                      np.where(win_penalties,
                                               np.where(Matches['Win conditions'].str.split(
                                                   pat='\(|\)|-', expand=True)[1] > Matches[
                                                   'Win conditions'].str.split(pat='\(|\)|-', expand=True)[2],
                                                        Matches['Home Team Name'], Matches['Away Team Name']),
                                               '')))
Matches['Looser'] = np.where(Matches['result'] != 'draw',
                             np.where(Matches['Winner'] == Matches['Home Team Name'],
                                      Matches['Away Team Name'],
                                      Matches['Home Team Name']), '')
```

In [32]:
```
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import seaborn as sns

# Use dark colors for the text
wc_cup = WordCloud(width=400, height=200, background_color="white", max_words=50000)

wc_cup.generate(' '.join(Matches['Winner'].dropna().tolist()))

plt.figure(figsize=(15, 5))
sns.set_style("ticks")


plt.title('Word cloud of the team with the most wins', fontsize=15)
plt.imshow(wc_cup, interpolation='bilinear')
plt.axis("off")

plt.show()
```

In [ ]: