International Football Matches Analysis

This notebook provides a comprehensive analysis of a rich dataset containing 47,126 results of international football matches from 1872 to 2024. The dataset covers various types of matches, including FIFA World Cup games, friendly matches, and other international tournaments. It exclusively includes men's full internationals, excluding Olympic Games and matches involving B-teams, U-23 teams, or league select teams. The analysis aims to uncover trends, patterns, and insights from this extensive collection of football match data.

```
In [3]: from IPython.display import Image

        # Display the image
        Image(filename= r'C:\Users\Syed Arif\OneDrive\Desktop\Football.jpeg')
```

Out[3]:

Import Library

```
In [5]: import pandas as pd
```

```
In [6]: import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        import seaborn as sns
```

Uploading Csv fle

```
In [91]: df = pd.read_csv(r"C:\Users\Syed Arif\OneDrive\Desktop\goalscorers.csv\goalscorers.csv")
```

```
In [95]: df1 = pd.read_csv(r"C:\Users\Syed Arif\OneDrive\Desktop\results.csv\results.csv")
```

Data Preprocessing

head()

head is used show to the By default = 5 rows in the dataset

```
In [12]: df.head()
```

Out[12]:

	date	home_team	away_team	team	scorer	minute	own_goal	penalty
0	1916-07-02	Chile	Uruguay	Uruguay	José Piendibene	44.0	False	False
1	1916-07-02	Chile	Uruguay	Uruguay	Isabelino Gradín	55.0	False	False
2	1916-07-02	Chile	Uruguay	Uruguay	Isabelino Gradín	70.0	False	False
3	1916-07-02	Chile	Uruguay	Uruguay	José Piendibene	75.0	False	False
4	1916-07-06	Argentina	Chile	Argentina	Alberto Ohaco	2.0	False	False

.tail()

tail is used to show rows by Descending order

In [15]:
```
df.tail()
```

Out[15]:

	date	home_team	away_team	team	scorer	minute	own_goal	penalty
44330	2024-07-05	Germany	Spain	Spain	Dani Olmo	51.0	False	False
44331	2024-07-05	Germany	Spain	Germany	Florian Wirtz	89.0	False	False
44332	2024-07-05	Germany	Spain	Spain	Mikel Merino	119.0	False	False
44333	2024-07-05	Venezuela	Canada	Canada	Jacob Shaffelburg	13.0	False	False
44334	2024-07-05	Venezuela	Canada	Venezuela	Salomón Rondón	65.0	False	False

shape

It show the total no of rows & Column in the dataset

In [18]:
```
df.shape
```

Out[18]:
```
(44335, 8)
```

Columns

It show the no of each Column

In [21]:
```
df.columns
```

Out[21]:
```
Index(['date', 'home_team', 'away_team', 'team', 'scorer', 'minute',
       'own_goal', 'penalty'],
      dtype='object')
```

.dtypes

This Attribute show the data type of each column

In [24]:
```
df.dtypes
```

Out[24]:
```
date          object
home_team     object
away_team     object
team          object
scorer        object
minute       float64
own_goal        bool
penalty         bool
dtype: object
```

.unique()

In a column, It show the unique value of specific column.

In [27]:
```
df["home_team"].unique()
```

```
Out[27]: array(['Chile', 'Argentina', 'Brazil', 'Uruguay', 'Paraguay',
        'Czechoslovakia', 'Italy', 'Switzerland', 'United States',
        'Hungary', 'France', 'Netherlands', 'Republic of Ireland', 'Egypt',
        'Sweden', 'Bolivia', 'Peru', 'Belgium', 'Portugal', 'Germany',
        'Spain', 'Latvia', 'Estonia', 'Lithuania', 'Yugoslavia', 'Poland',
        'Haiti', 'Mexico', 'Luxembourg', 'Bulgaria', 'Israel', 'Austria',
        'Romania', 'Finland', 'Norway', 'Greece', 'Cuba', 'Ecuador',
        'Colombia', 'Northern Ireland', 'Wales', 'Scotland', 'England',
        'Turkey', 'Japan', 'Saarland', 'South Korea', 'Hong Kong',
        'Guatemala', 'Sudan', 'Costa Rica', 'Indonesia', 'Denmark',
        'German DR', 'Syria', 'China PR', 'Canada', 'Russia', 'Curaçao',
        'Iceland', 'Ethiopia', 'Ghana', 'Honduras', 'Nigeria', 'Suriname',
        'Taiwan', 'Morocco', 'Cyprus', 'Tunisia', 'Malta', 'El Salvador',
        'Jamaica', 'Nicaragua', 'Panama', 'Albania', 'India',
        'Trinidad and Tobago', 'Venezuela', 'DR Congo', 'Ivory Coast',
        'North Korea', 'Australia', 'Algeria', 'Congo', 'Iran', 'Myanmar',
        'Zambia', 'Bermuda', 'Cameroon', 'Senegal', 'Libya', 'Zimbabwe',
        'Mali', 'Kenya', 'Guinea', 'Thailand', 'Cambodia', 'Benin',
        'Sierra Leone', 'Tanzania', 'Mauritius', 'New Caledonia',
        'New Zealand', 'Fiji', 'Vanuatu', 'Iraq', 'Vietnam Republic',
        'Uganda', 'Burkina Faso', 'Niger', 'Mauritania', 'Malawi',
        'Kuwait', 'Togo', 'Saudi Arabia', 'Singapore', 'Malaysia',
        'Bahrain', 'Qatar', 'Solomon Islands', 'Lesotho', 'Mozambique',
        'Somalia', 'Bangladesh', 'United Arab Emirates', 'Madagascar',
        'Angola', 'Gambia', 'Liberia', 'Macau', 'Brunei', 'Nepal',
        'Jordan', 'Yemen', 'Yemen DPR', 'Oman', 'Pakistan', 'Gabon',
        'Faroe Islands', 'San Marino', 'Dominican Republic', 'Saint Lucia',
        'Puerto Rico', 'Saint Vincent and the Grenadines', 'Barbados',
        'Guyana', 'Antigua and Barbuda', 'Tahiti', 'Burundi', 'Eswatini',
        'Namibia', 'South Africa', 'Botswana', 'Sri Lanka', 'Vietnam',
        'Czech Republic', 'Lebanon', 'Georgia', 'Liechtenstein',
        'North Macedonia', 'Slovenia', 'Ukraine', 'Croatia', 'Belarus',
        'Moldova', 'Azerbaijan', 'Slovakia', 'Armenia', 'Dominica',
        'Aruba', 'Grenada', 'Serbia', 'Saint Kitts and Nevis',
        'Cayman Islands', 'Guinea-Bissau', 'Belize', 'Rwanda',
        'Papua New Guinea', 'Philippines', 'Bosnia and Herzegovina',
        'Tonga', 'Samoa', 'Uzbekistan', 'Tajikistan', 'Turkmenistan',
        'Kazakhstan', 'Maldives', 'Kyrgyzstan', 'Andorra', 'Anguilla',
        'British Virgin Islands', 'Bahamas', 'Montserrat',
        'United States Virgin Islands', 'Djibouti',
        'Central African Republic', 'Seychelles', 'São Tomé and Príncipe',
        'Chad', 'Equatorial Guinea', 'Mongolia', 'Palestine',
        'American Samoa', 'Laos', 'Cook Islands', 'Martinique',
        'Cape Verde', 'Afghanistan', 'Turks and Caicos Islands',
        'Guadeloupe', 'Timor-Leste', 'Comoros', 'Montenegro', 'Eritrea',
        'Gibraltar', 'Bhutan', 'Guam', 'South Sudan', 'Kosovo',
        'French Guiana'], dtype=object)
```

.nuique()

It will show the total no of unque value from whole data frame

```
In [30]: df.nunique()
```

```
Out[30]: date          4627
home_team      220
away_team      220
team           220
scorer       14331
minute         121
own_goal         2
penalty          2
dtype: int64
```

.describe()

It show the Count, mean , median etc

```
In [33]: df.describe()
```

Out[33]:

	minute
count	44076.000000
mean	50.012478
std	26.354402
min	1.000000
25%	28.000000
50%	51.000000
75%	73.000000
max	122.000000

.value_counts

It Shows all the unique values with their count

In [36]:
```
df["home_team"].value_counts()
```

Out[36]:
```
home_team
Brazil              1023
Argentina            989
Germany              798
Mexico               707
France               667
                    ... 
Somalia                5
Yemen DPR              5
Vietnam Republic       4
South Sudan            4
Saarland               4
Name: count, Length: 220, dtype: int64
```

isnull()

It shows the how many null values

In [39]:
```
df.isnull()
```

Out[39]:

	date	home_team	away_team	team	scorer	minute	own_goal	penalty
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
44330	False	False	False	False	False	False	False	False
44331	False	False	False	False	False	False	False	False
44332	False	False	False	False	False	False	False	False
44333	False	False	False	False	False	False	False	False
44334	False	False	False	False	False	False	False	False

44335 rows × 8 columns

.info()

To Show Data type of each column

In [42]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44335 entries, 0 to 44334
Data columns (total 8 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   date       44335 non-null  object 
 1   home_team  44335 non-null  object 
 2   away_team  44335 non-null  object 
 3   team       44335 non-null  object 
 4   scorer     44286 non-null  object 
 5   minute     44076 non-null  float64
 6   own_goal   44335 non-null  bool   
 7   penalty    44335 non-null  bool   
dtypes: bool(2), float64(1), object(5)
memory usage: 2.1+ MB
```

Which teams have played the most matches in the dataset ?

In [44]:
```python
df['team'].value_counts(ascending=False).reset_index()
```

Out[44]:

	team	count
0	Brazil	1051
1	Germany	974
2	Argentina	945
3	Spain	890
4	Mexico	852
...	...	...
215	Eritrea	4
216	Somalia	3
217	Anguilla	2
218	South Sudan	2
219	French Guiana	2

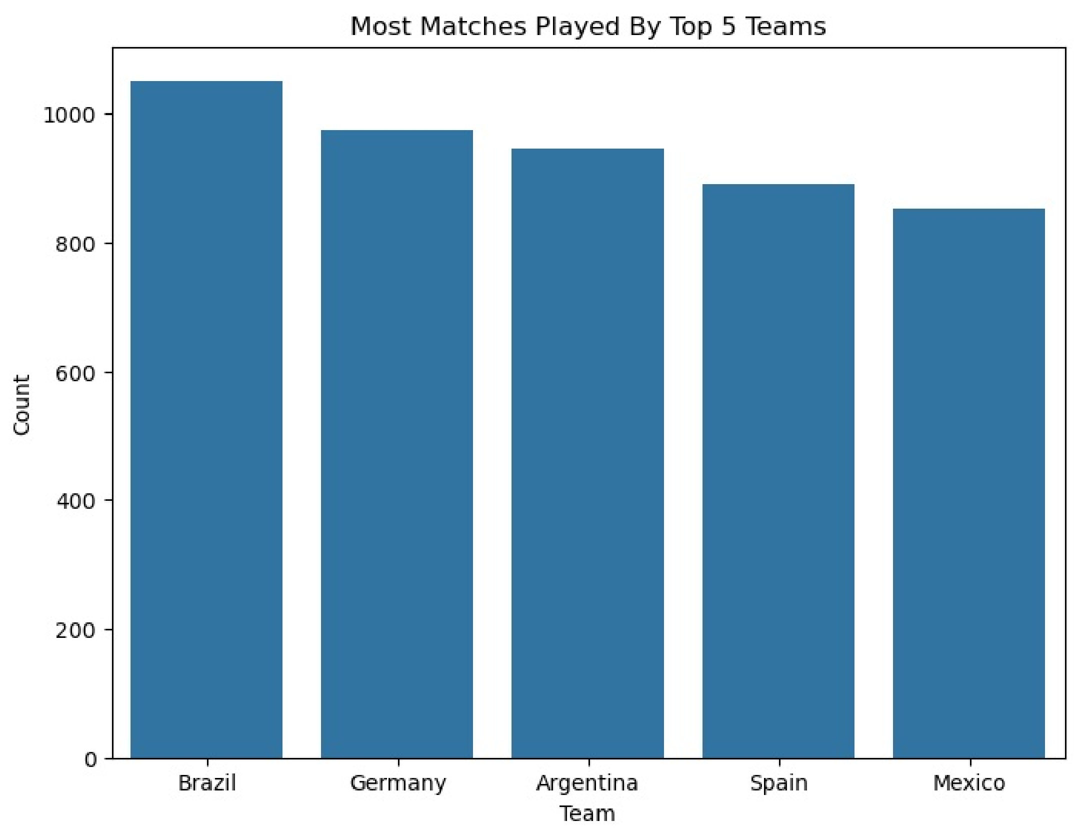
220 rows × 2 columns

Most Matches Played By Top 5 Teams

In [46]:
```python
top_teams = df['team'].value_counts().nlargest(5).index

# Filter the DataFrame to include only the top 5 teams
filtered_df = df[df['team'].isin(top_teams)]

# Plot the count of matches played by the top 5 teams
plt.figure(figsize=(8, 6))
sns.countplot(data=filtered_df, x='team', order=top_teams)
plt.xlabel('Team')
plt.ylabel('Count')
plt.title('Most Matches Played By Top 5 Teams')
plt.show()
```



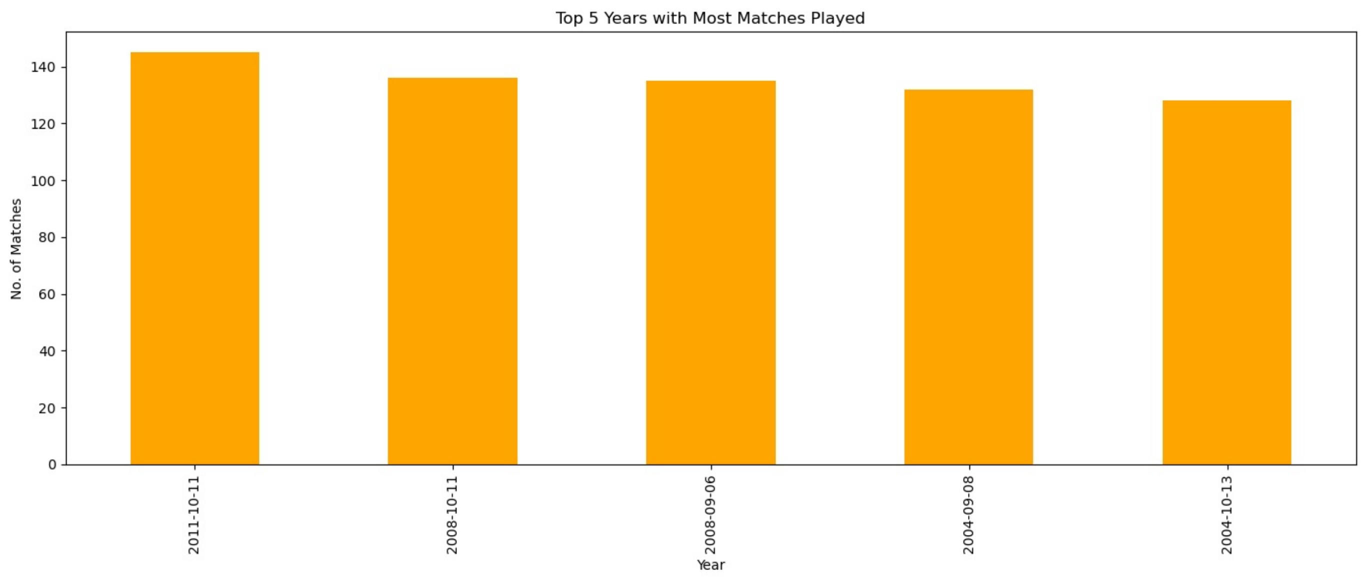
What is the distribution of matches played over the years ?

In [48]:
```
# First, get the number of matches played each year
matches_by_year = df.groupby('date').size()

# Select the top 5 years by the number of matches played
top_5_years = matches_by_year.nlargest(5).index

# Filter the Series to include only the top 5 years
filtered_matches_by_year = matches_by_year[top_5_years]

# Plot the bar chart for the top 5 years
plt.figure(figsize=(14, 6))
filtered_matches_by_year.plot(kind='bar', color='orange')
plt.title('Top 5 Years with Most Matches Played')
plt.xlabel('Year')
plt.ylabel('No. of Matches')
plt.tight_layout()
plt.show()
```



In [49]:
```
df.dtypes
```

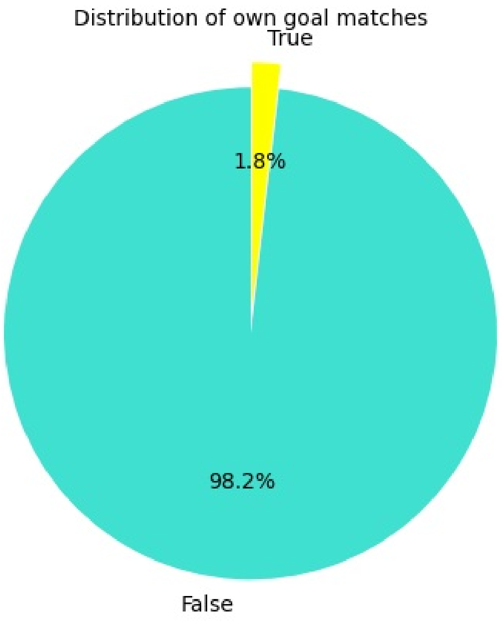
```
Out[49]: date          object
         home_team     object
         away_team     object
         team          object
         scorer        object
         minute       float64
         own_goal        bool
         penalty         bool
         dtype: object
```

How many matches resulted in own goals ?

```
In [57]: df['own_goal'].value_counts()
```

```
Out[57]: own_goal
         False    43515
         True       820
         Name: count, dtype: int64
```

```
In [65]: goals = df['own_goal'].value_counts()
         plt.pie(goals.values, labels= goals.index, autopct='%1.1f%%', startangle=90, explode=[0, 0.1], colors=['turquois
         plt.title('Distribution of own goal matches', size=10)
         plt.axis('equal')
         plt.show()
```



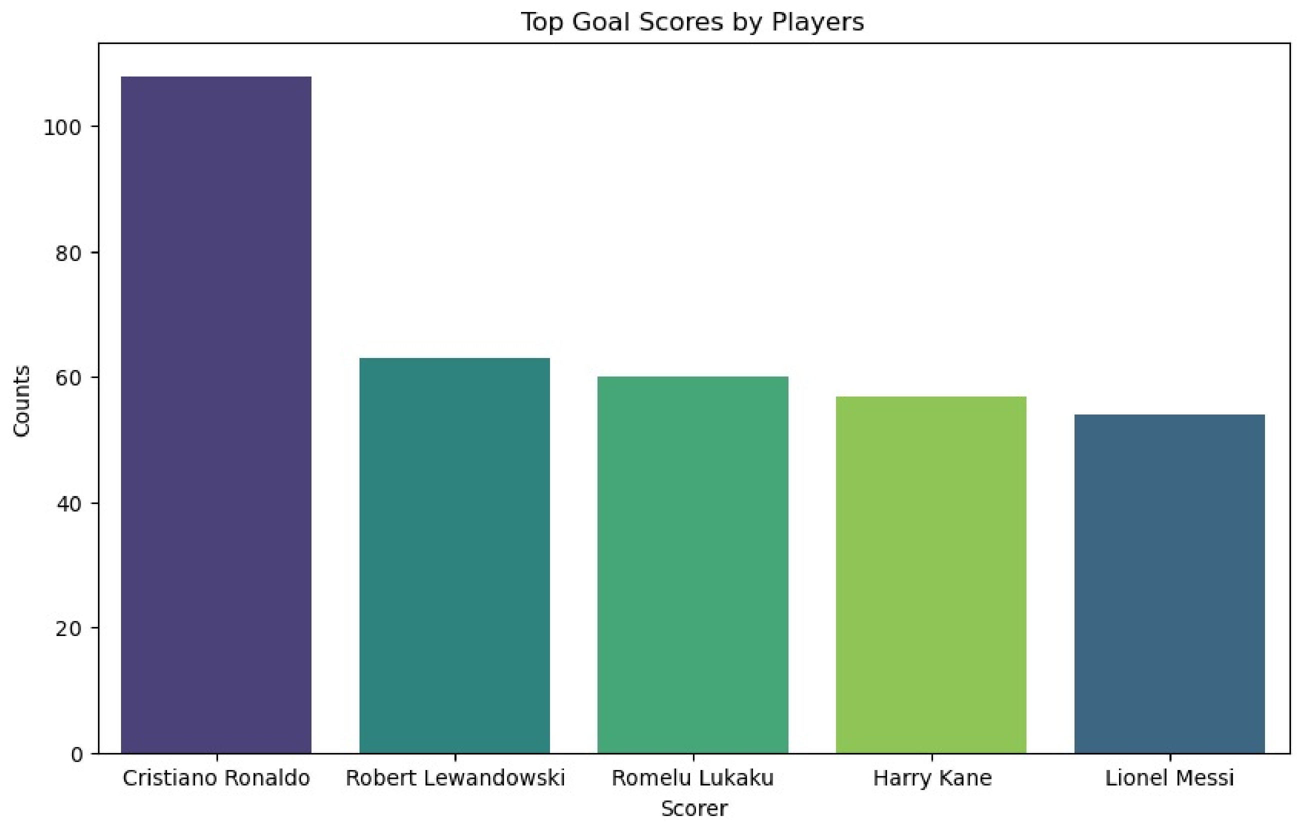
Top Goal Scorer's

```
In [78]: # Calculate the value counts for 'scorer'
         Top_players = df['scorer'].value_counts()

         # Get the top 5 scorers
         Top_goal_score = Top_players.nlargest(5).index

         # Filter the DataFrame to include only the top 5 scorers
         filtered_df = df[df['scorer'].isin(Top_goal_score)]

         # Plot the count plot for the top 5 scorers with a beautiful color palette
         plt.figure(figsize=(10, 6))
         sns.countplot(data=filtered_df, x='scorer', hue='scorer', order=Top_goal_score, palette='viridis', dodge=False,
         plt.xlabel('Scorer')
         plt.ylabel('Counts')
         plt.title('Top Goal Scores by Players')
         plt.show()
```



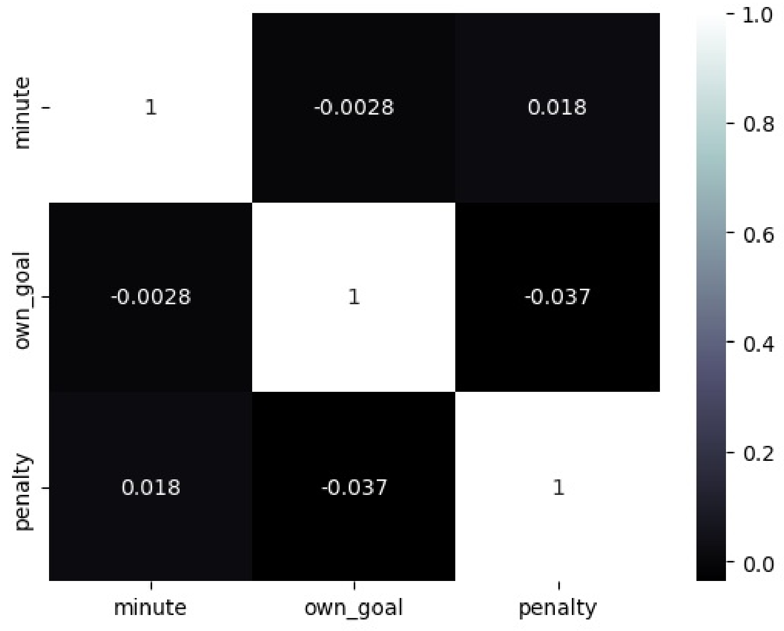
What is the frequency of penalty goals in matches?

In [81]:
```
df['penalty'].value_counts().reset_index()
```

Out[81]:

	penalty	count
0	False	41376
1	True	2959

In [85]:
```
sns.heatmap(df.corr(numeric_only=True),annot=True, cmap='bone')
plt.show()
```



In [97]:
```
df1
```

Out[97]:

	date	home_team	away_team	home_score	away_score	tournament	city	country	neutral
0	1872-11-30	Scotland	England	0.0	0.0	Friendly	Glasgow	Scotland	False
1	1873-03-08	England	Scotland	4.0	2.0	Friendly	London	England	False
2	1874-03-07	Scotland	England	2.0	1.0	Friendly	Glasgow	Scotland	False
3	1875-03-06	England	Scotland	2.0	2.0	Friendly	London	England	False
4	1876-03-04	Scotland	England	3.0	0.0	Friendly	Glasgow	Scotland	False
...	...	...	...	...	...	...	...	...	...
47376	2024-07-10	NaN	NaN	NaN	NaN	UEFA Euro	Dortmund	Germany	True
47377	2024-07-10	NaN	NaN	NaN	NaN	Copa América	Charlotte	United States	True
47378	2024-07-13	NaN	NaN	NaN	NaN	Copa América	Charlotte	United States	True
47379	2024-07-14	NaN	NaN	NaN	NaN	UEFA Euro	Berlin	Germany	True
47380	2024-07-14	NaN	NaN	NaN	NaN	Copa América	Miami Gardens	United States	True

47381 rows × 9 columns

Summary of the dataset

In [108…

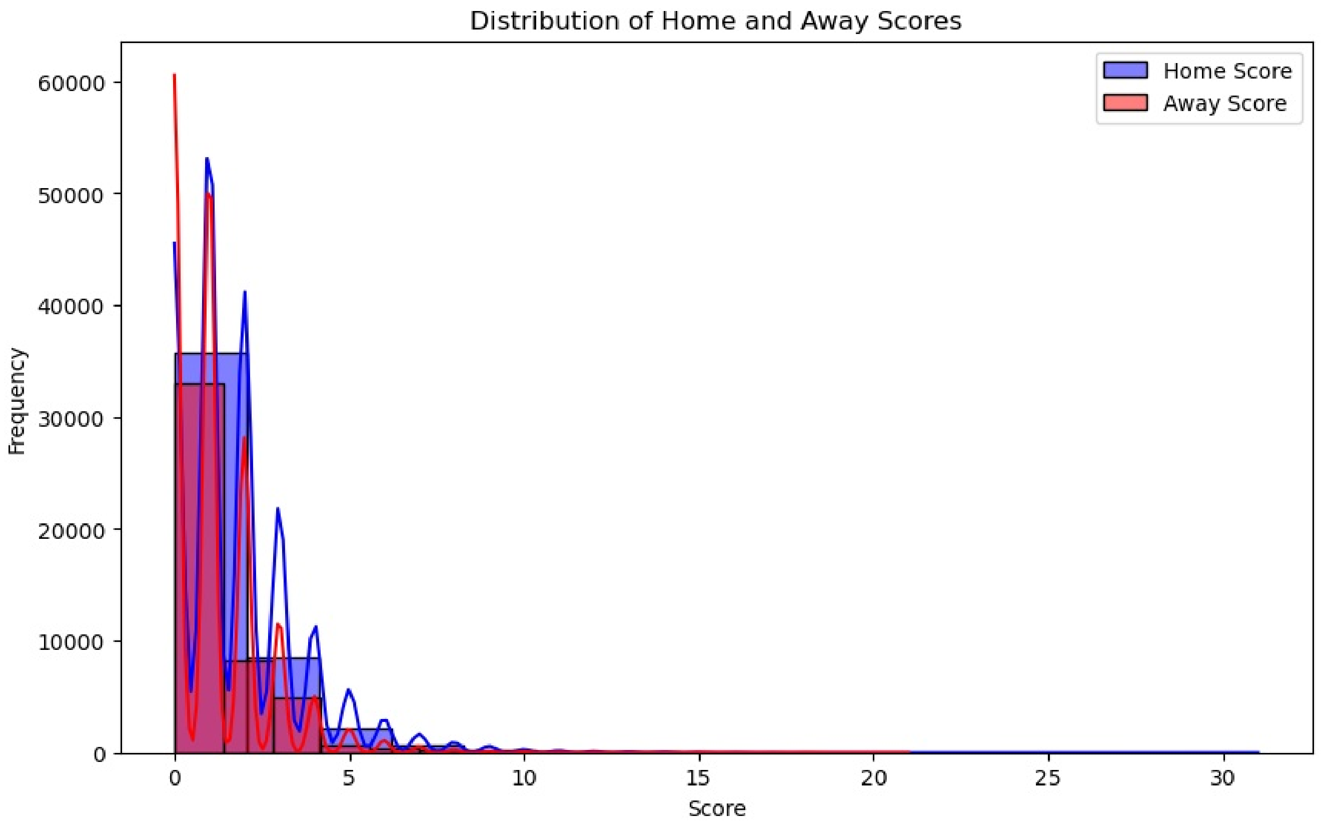
```python
# Summary of the dataset
num_matches = len(df1)
print(f"Number of matches: {num_matches}")
```

```
Number of matches: 47381
```

What is the distribution of home and away scores?

In [113…

```python
# Distribution of home and away scores
plt.figure(figsize=(10, 6))
sns.histplot(data=df1, x='home_score', bins=15, kde=True, color='blue', label='Home Score')
sns.histplot(data=df1, x='away_score', bins=15, kde=True, color='red', label='Away Score')
plt.title('Distribution of Home and Away Scores')
plt.xlabel('Score')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

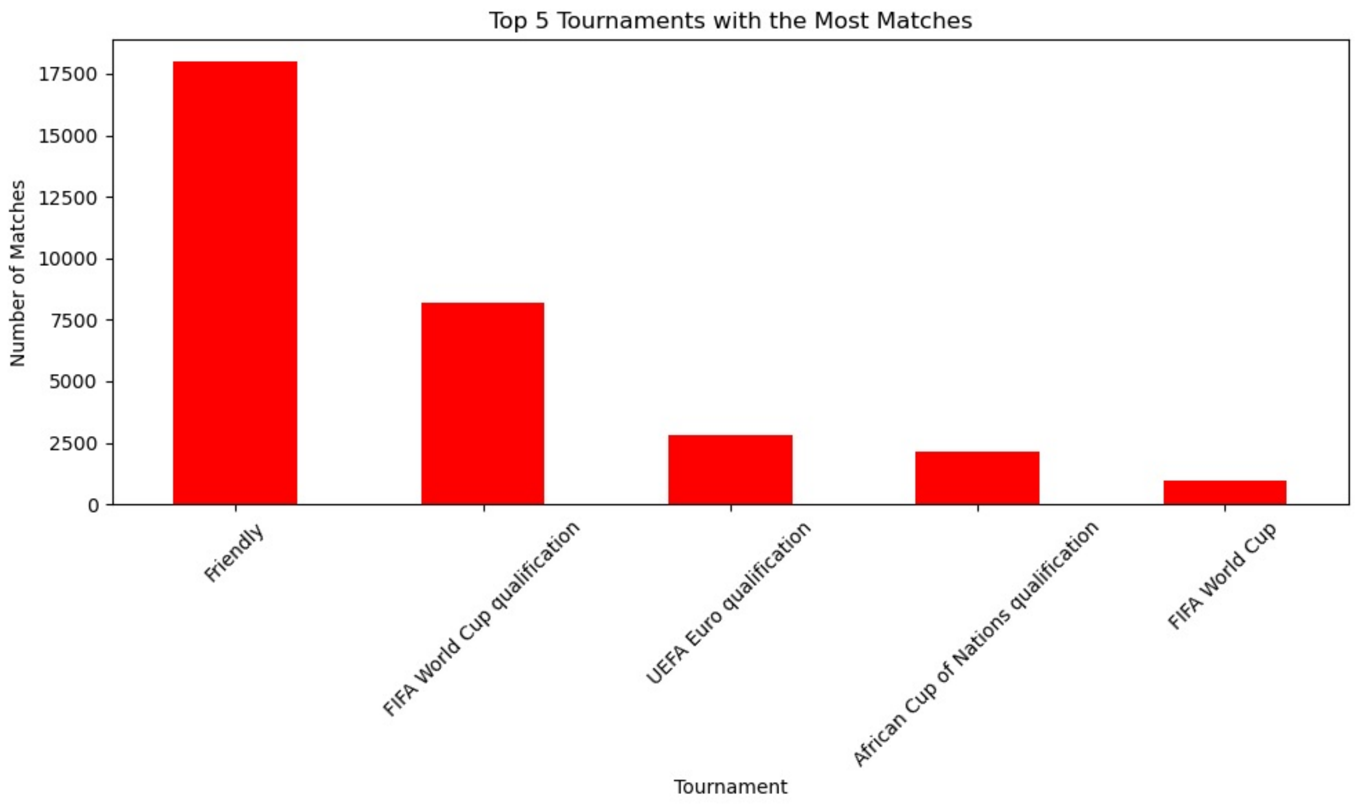


Which tournaments have the most matches recorded?

In [120…

```python
# Top 5 tournaments with the most matches
top_tournaments = df1['tournament'].value_counts().nlargest(5)

plt.figure(figsize=(10, 6))
top_tournaments.plot(kind='bar', color='red')
plt.title('Top 5 Tournaments with the Most Matches')
plt.xlabel('Tournament')
plt.ylabel('Number of Matches')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Which cities/countries have hosted the most matches?

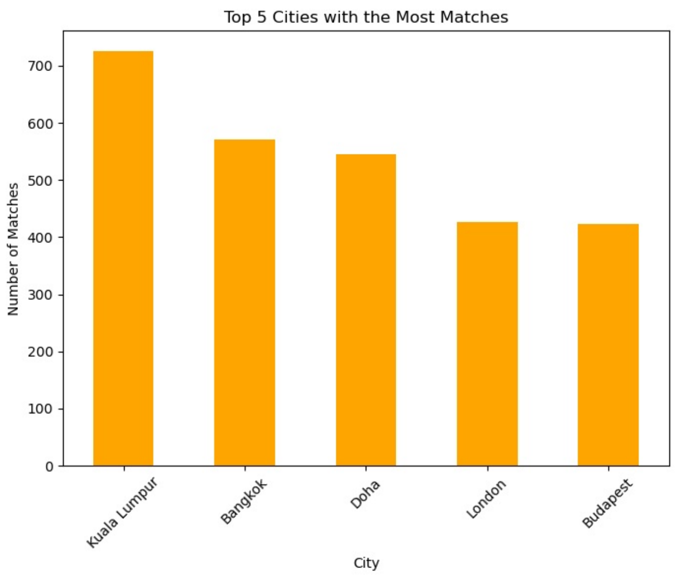
In [122…

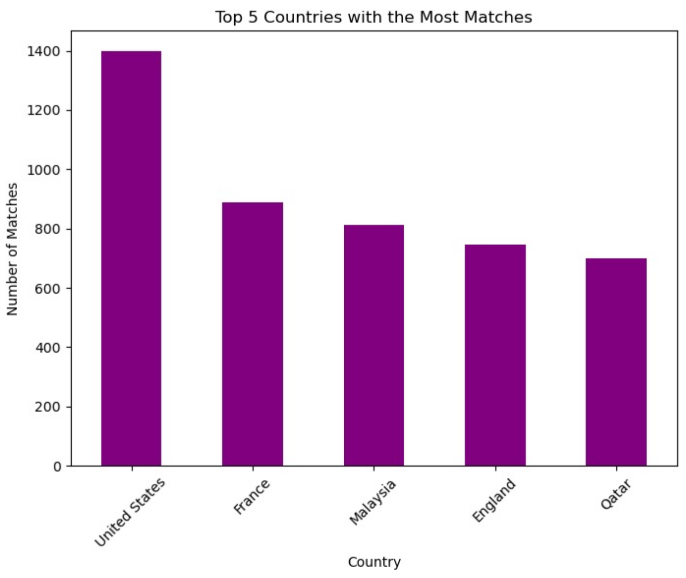
```python
# City and country analysis
top_cities = df1['city'].value_counts().nlargest(5)
top_countries = df1['country'].value_counts().nlargest(5)

plt.figure(figsize=(14, 6))
plt.subplot(1, 2, 1)
top_cities.plot(kind='bar', color='orange')
plt.title('Top 5 Cities with the Most Matches')
plt.xlabel('City')
plt.ylabel('Number of Matches')
plt.xticks(rotation=45)

plt.subplot(1, 2, 2)
top_countries.plot(kind='bar', color='purple')
plt.title('Top 5 Countries with the Most Matches')
plt.xlabel('Country')
plt.ylabel('Number of Matches')
plt.xticks(rotation=45)

plt.tight_layout()
plt.show()
```

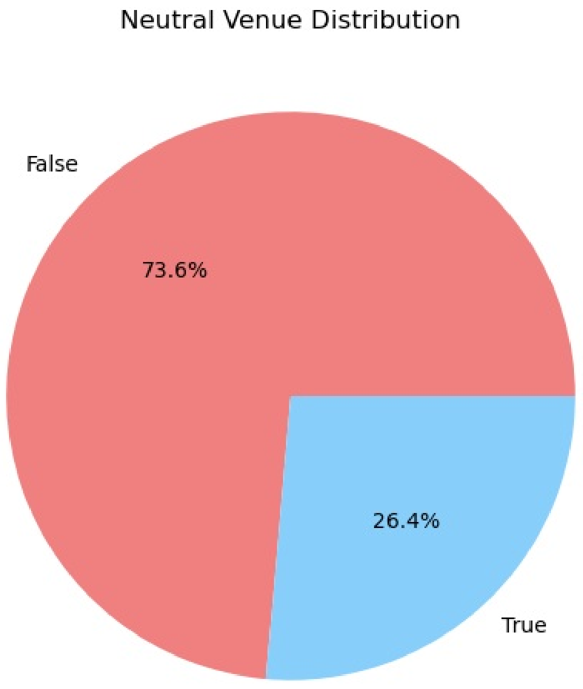




How does neutral venue status affect match outcomes?

```
In [128…
# Neutral venue analysis
neutral_matches = df1['neutral'].value_counts()

plt.figure(figsize=(6, 6))
plt.pie(neutral_matches, labels=neutral_matches.index, autopct='%1.1f%%', colors=['lightcoral', 'lightskyblue']
plt.title('Neutral Venue Distribution')
plt.show()
```



Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js