use this data set and give me atleast 8 complex but clear queries in pandas with visualization

https://www.kaggle.com/datasets/davidcariboo/player-scores

i have multiple files with column name

1.transfers(player\_id transfer\_date transfer\_season from\_club\_id to\_club\_id from\_club\_name to\_club\_name transfer\_fee market\_value\_in\_eur player\_name)

2.competitions(competition\_id competition\_code name sub\_type type country\_id country\_name domestic\_league\_code confederation url is\_major\_national\_league)

3.clubs(club\_id club\_code name domestic\_competition\_id total\_market\_value squad\_size average\_age foreigners\_number foreigners\_percentage national\_team\_players stadium\_name stadium\_seats net\_transfer\_record coach\_name last\_season filename url)

4.club\_games(game\_id club\_id own\_goals own\_position own\_manager\_name opponent\_id opponent\_goals opponent\_position opponent\_manager\_name hosting is\_win)

5.appearences(appearance\_id game\_id player\_id player\_club\_id player\_current\_club\_id date player\_name competition\_id yellow\_cards red\_cards goals assists minutes\_played)

6.game\_events(game\_event\_id date game\_id minute type club\_id player\_id description player\_in\_id player\_assist\_id)

7.players(player\_id first\_name last\_name name last\_season current\_club\_id player\_code country\_of\_birth city\_of\_birth country\_of\_citizenship date\_of\_birth sub\_position position foot height\_in\_cm contract\_expiration\_date agent\_name image\_url url current\_club\_domestic\_competition\_id current\_club\_name market\_value\_in\_eur highest\_market\_value\_in\_eur)

8.player\_valuation(player\_id date market\_value\_in\_eur current\_club\_id player\_club\_domestic\_competition\_id)

9.games(game\_id competition\_id season round date home\_club\_id away\_club\_id home\_club\_goals away\_club\_goals home\_club\_position away\_club\_position home\_club\_manager\_name away\_club\_manager\_name stadium attendance referee url home\_club\_formation away\_club\_formation home\_club\_name away\_club\_name aggregate competition\_type)

10.game\_lineups(game\_lineups\_id date game\_id player\_id club\_id player\_name type position number team\_captain)

import pandas as pd

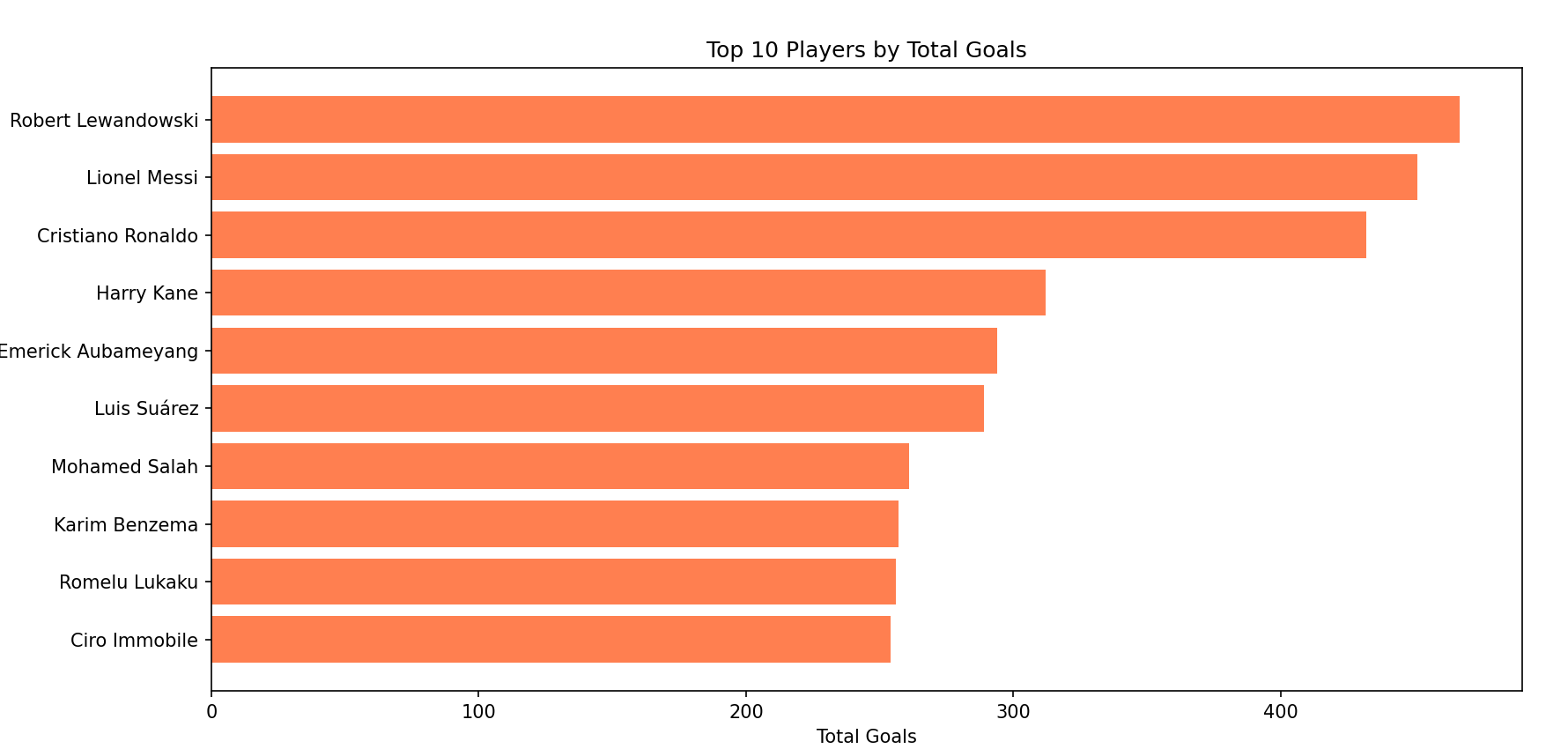
df = pd.read\_csv('abfss://aamna@youexcel.dfs.core.windows.net/football/goalscorers.csv',

storage\_options = {'account\_key':'/dhYW+OiOeruQ2q2lRlOVQHX+VYV50A1MDZYASkOS6BP6NqNMeL12c2uqhhUzDp92mNhSurH6uI++AStKYyqpQ=='})

display(df)

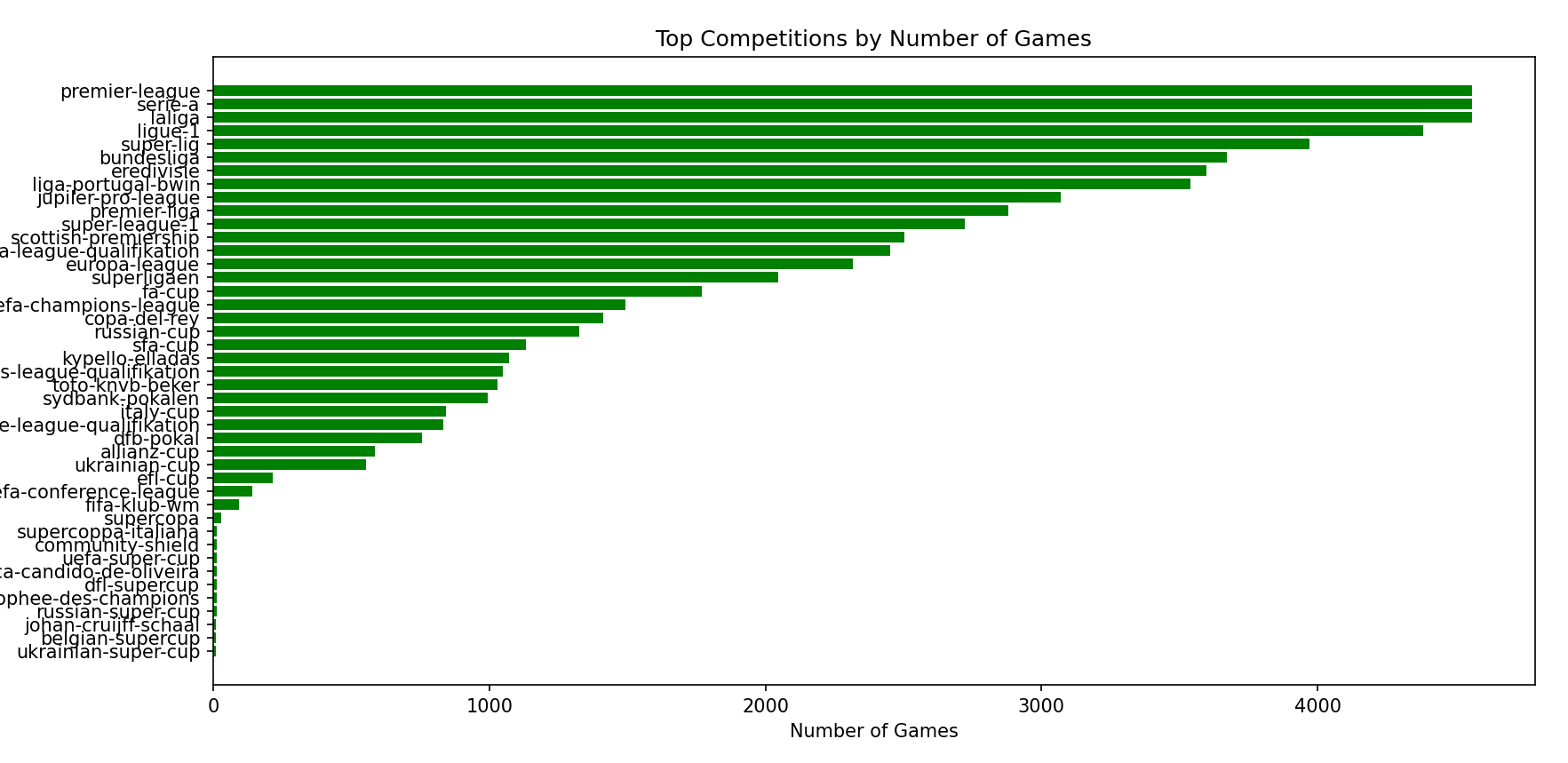
1. **Top Players by Total Goals**

import pandas as pd  
import matplotlib.pyplot as plt  
  
# Load datasets  
appearances = pd.read\_csv('appearances.csv')  
  
# Aggregate total goals by player  
top\_scorers = appearances.groupby('player\_id')['goals'].sum().reset\_index()  
top\_scorers = top\_scorers.sort\_values(by='goals', ascending=False).head(10)  
  
# Load player names for better readability  
players = pd.read\_csv('players.csv')  
top\_scorers = top\_scorers.merge(players[['player\_id', 'name']], on='player\_id')  
  
# Visualization  
plt.figure(figsize=(12, 8))  
plt.barh(top\_scorers['name'], top\_scorers['goals'], color='coral')  
plt.title('Top 10 Players by Total Goals')  
plt.xlabel('Total Goals')  
plt.ylabel('Player Name')  
plt.gca().invert\_yaxis()  
plt.show()



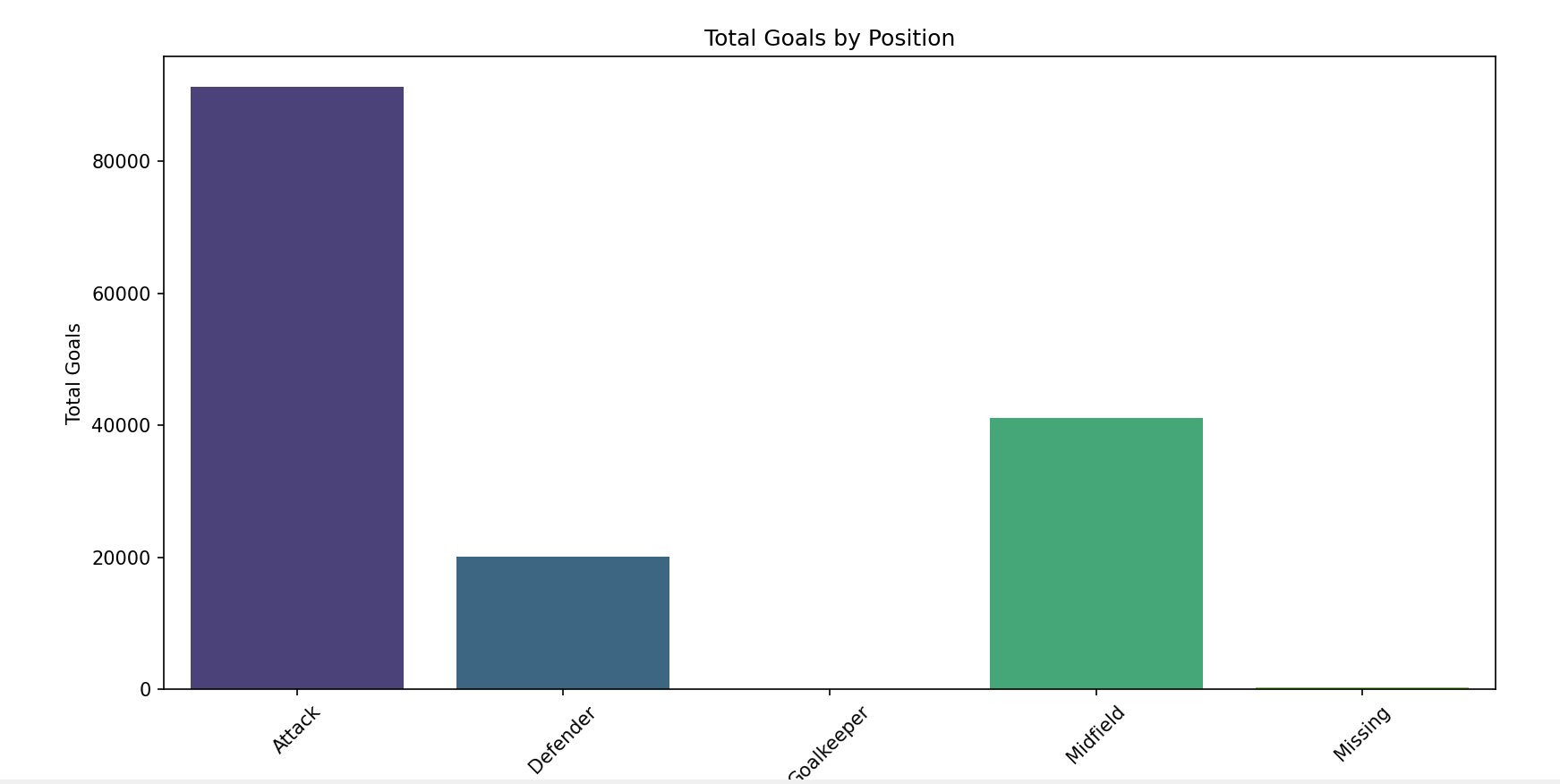
2. **Top Competitions by Number of Games**

import pandas as pd  
import matplotlib.pyplot as plt  
  
import seaborn as sns  
  
# Load datasets  
games = pd.read\_csv('games.csv')  
competitions = pd.read\_csv('competitions.csv')  
  
# Count games per competition  
games\_per\_competition = games['competition\_id'].value\_counts().reset\_index()  
games\_per\_competition.columns = ['competition\_id', 'number\_of\_games']  
  
# Merge with competition names  
games\_per\_competition = games\_per\_competition.merge(competitions[['competition\_id', 'name']], on='competition\_id')  
  
# Visualization  
plt.figure(figsize=(12, 8))  
plt.barh(games\_per\_competition['name'], games\_per\_competition['number\_of\_games'], color='green')  
plt.title('Top Competitions by Number of Games')  
plt.xlabel('Number of Games')  
plt.ylabel('Competition Name')  
plt.gca().invert\_yaxis()  
plt.show()



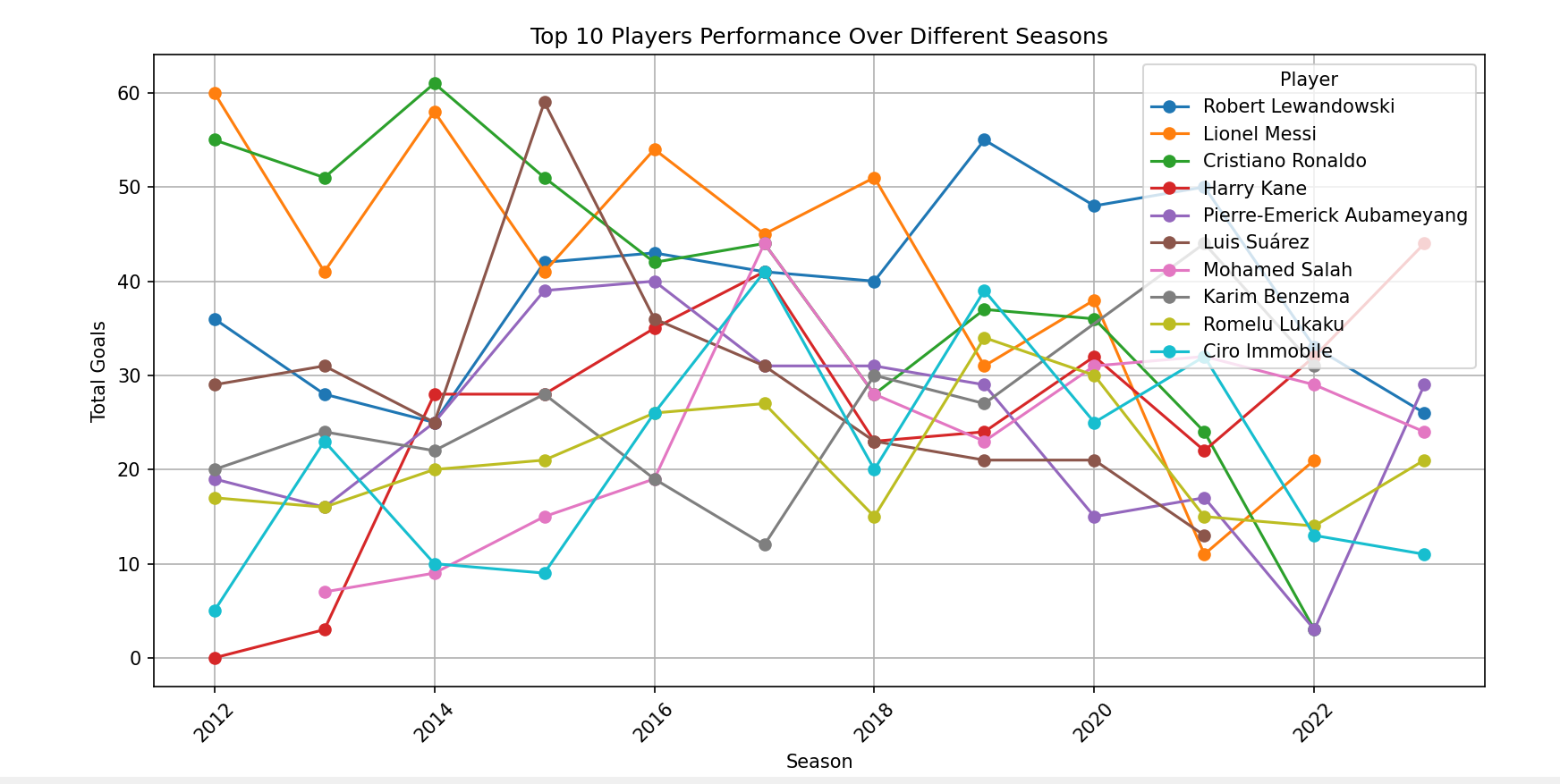
3. **Goal Distribution by Position**

import pandas as pd  
import matplotlib.pyplot as plt  
  
import seaborn as sns  
  
# Load datasets  
appearances = pd.read\_csv('appearances.csv')  
players = pd.read\_csv('players.csv')  
  
# Merge to get player positions  
player\_goals = appearances.merge(players[['player\_id', 'position']], on='player\_id')  
  
# Aggregate goals by position  
goals\_by\_position = player\_goals.groupby('position')['goals'].sum().reset\_index()  
  
# Visualization  
plt.figure(figsize=(12, 8))  
sns.barplot(data=goals\_by\_position, x='position', y='goals', palette='viridis')  
plt.title('Total Goals by Position')  
plt.xlabel('Position')  
plt.ylabel('Total Goals')  
plt.xticks(rotation=45)  
plt.show()



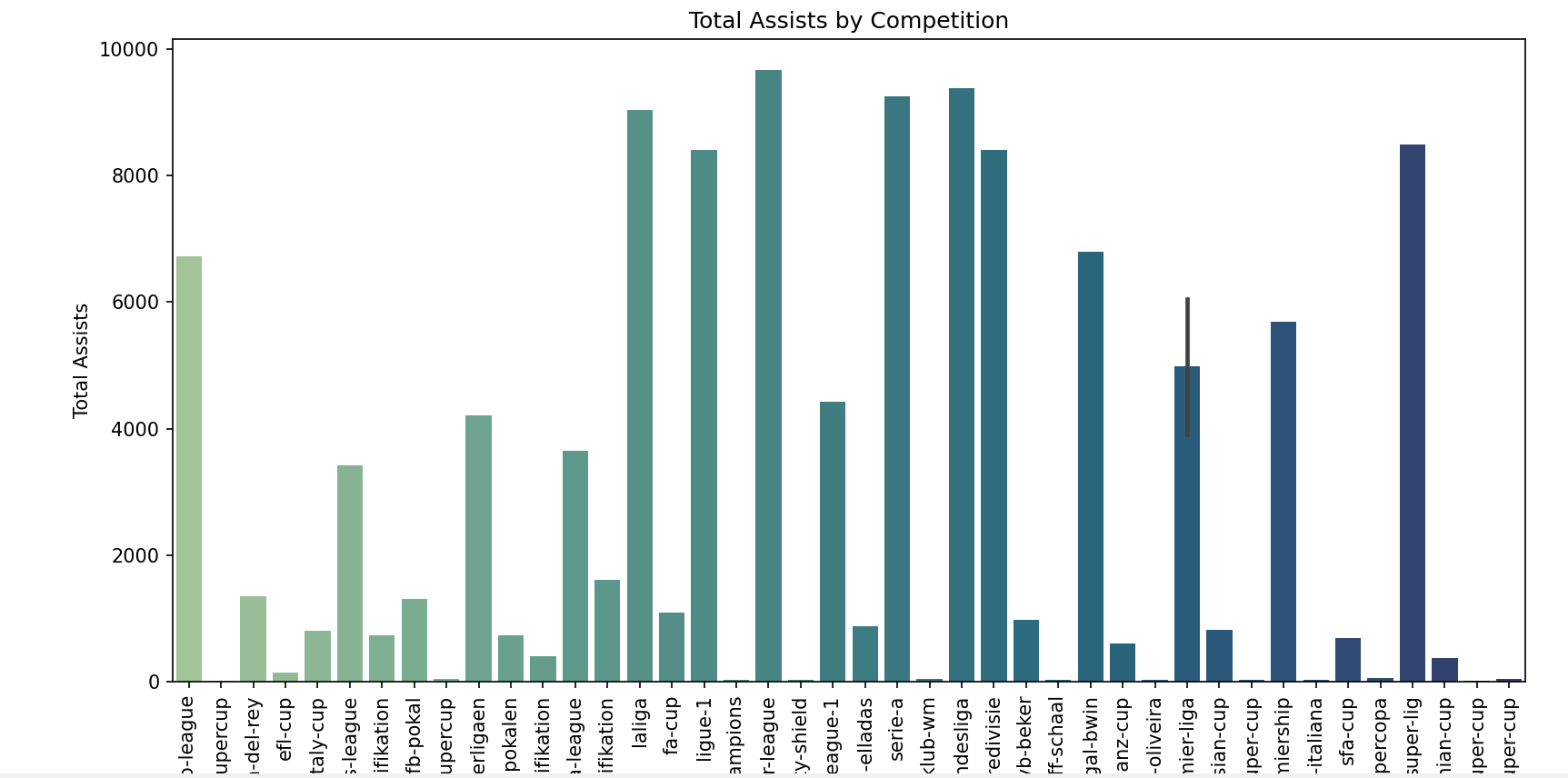
4. **Player Performance Over Different Seasons**

import pandas as pd  
import matplotlib.pyplot as plt  
  
# Load datasets  
appearances = pd.read\_csv('appearances.csv')  
players = pd.read\_csv('players.csv')  
games = pd.read\_csv('games.csv')  
  
# Merge datasets to get season information  
player\_performance = appearances.merge(games[['game\_id', 'season']], on='game\_id')  
player\_performance = player\_performance.groupby(['player\_id', 'season'])['goals'].sum().reset\_index()  
  
# Get top 10 players by total goals  
top\_players = player\_performance.groupby('player\_id')['goals'].sum().nlargest(10).index  
top\_players\_performance = player\_performance[player\_performance['player\_id'].isin(top\_players)]  
  
# Merge with player names  
top\_players\_performance = top\_players\_performance.merge(players[['player\_id', 'name']], on='player\_id')  
  
# Visualization  
plt.figure(figsize=(14, 10))  
for player in top\_players:  
 player\_data = top\_players\_performance[top\_players\_performance['player\_id'] == player]  
 plt.plot(player\_data['season'], player\_data['goals'], marker='o', label=player\_data['name'].iloc[0])  
  
plt.title('Top 10 Players Performance Over Different Seasons')  
plt.xlabel('Season')  
plt.ylabel('Total Goals')  
plt.legend(title='Player')  
plt.grid(True)  
plt.xticks(rotation=45)  
plt.show()



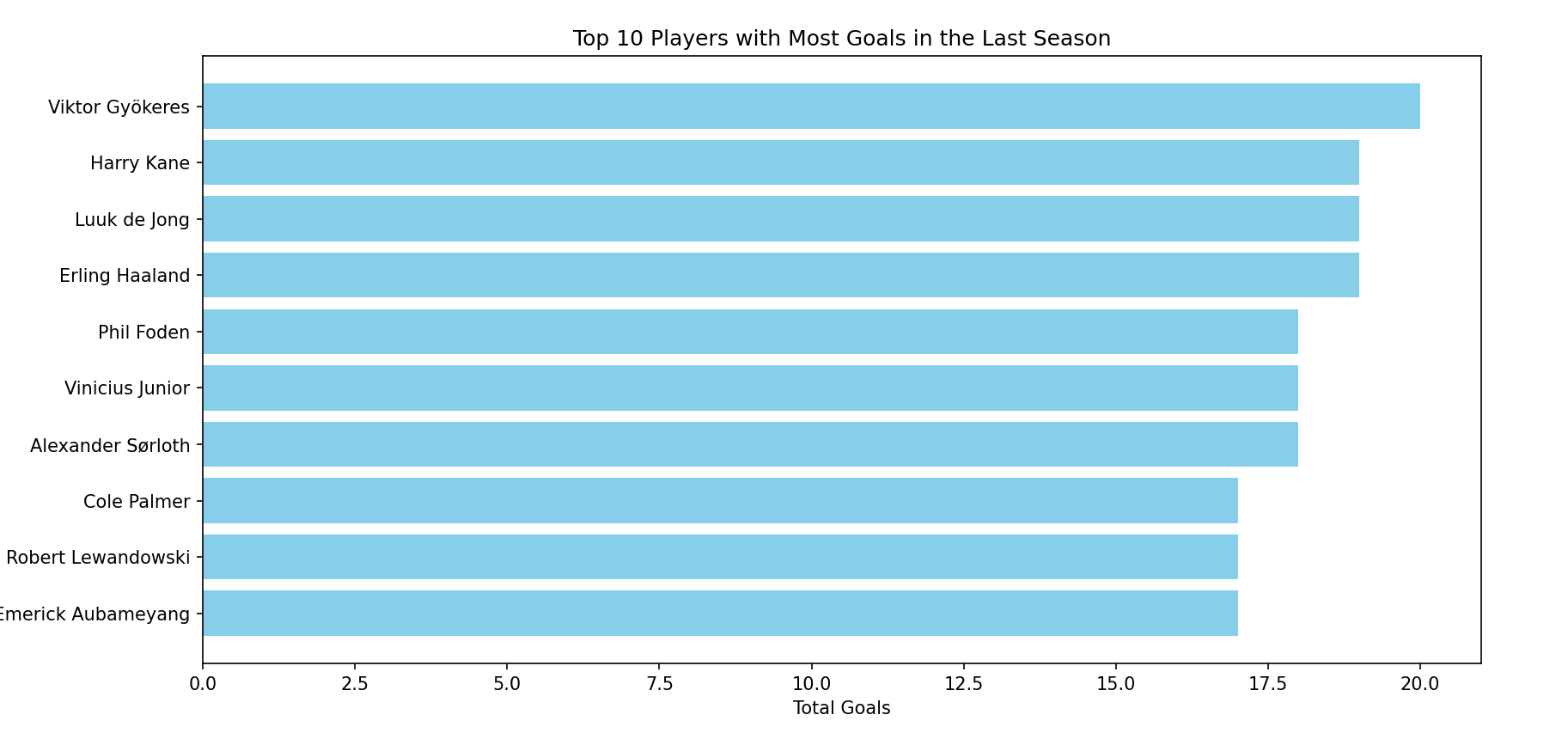
5. **Comparison of Total Assists by Competition**

import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
# Load datasets  
appearances = pd.read\_csv('appearances.csv')  
competitions = pd.read\_csv('competitions.csv')  
  
# Merge to get competition names  
assist\_per\_competition = appearances[['competition\_id', 'assists']].groupby('competition\_id').sum().reset\_index()  
assist\_per\_competition = assist\_per\_competition.merge(competitions[['competition\_id', 'name']], on='competition\_id')  
  
# Visualization  
plt.figure(figsize=(14, 8))  
sns.barplot(data=assist\_per\_competition, x='name', y='assists', palette='crest')  
plt.title('Total Assists by Competition')  
plt.xlabel('Competition')  
plt.ylabel('Total Assists')  
plt.xticks(rotation=90)  
plt.show()



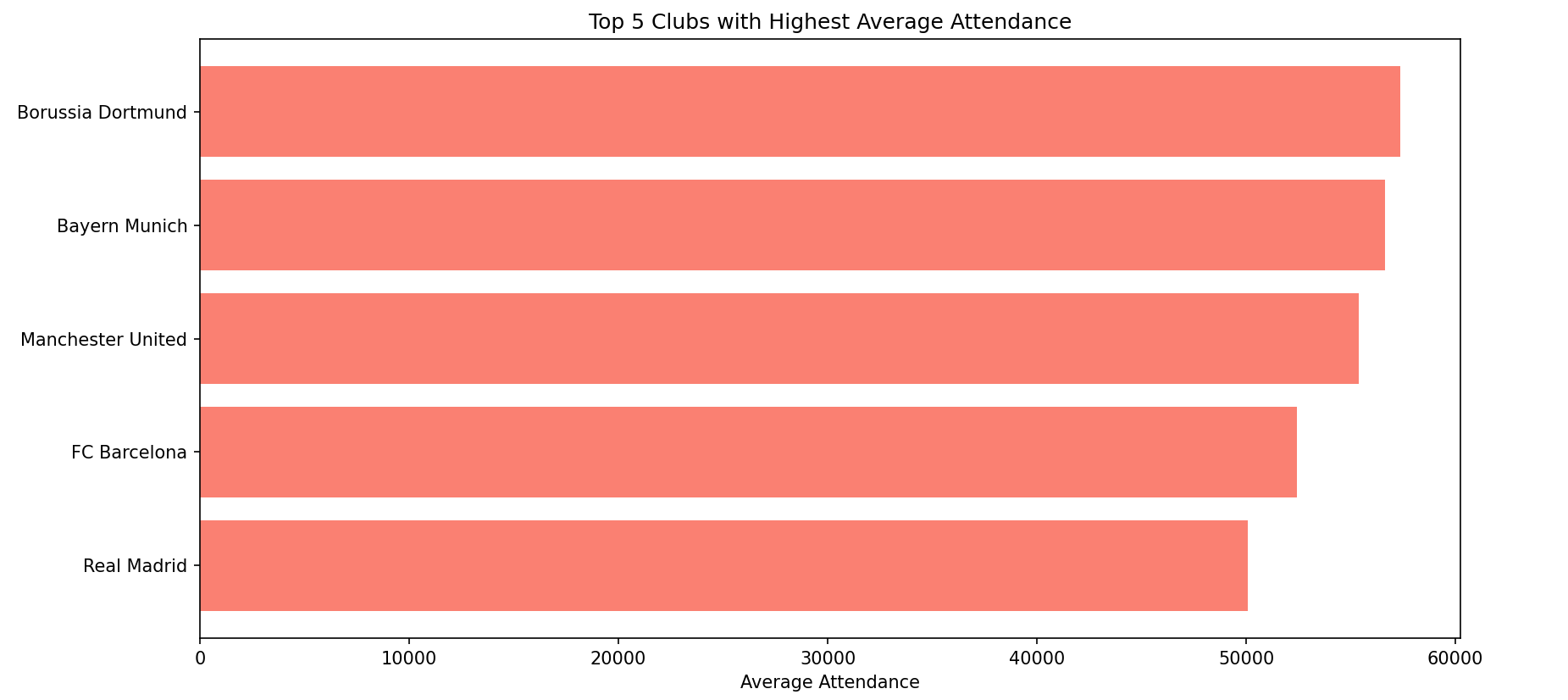
6. **Top 10 Players with Most Goals in the Last Season**

import pandas as pd  
import matplotlib.pyplot as plt  
  
# Load the data  
appearances = pd.read\_csv('appearances.csv')  
players = pd.read\_csv('players.csv')  
  
# Filter for the last season  
last\_season = appearances['date'].max().split('-')[0] # assuming the latest date corresponds to the last season  
appearances\_last\_season = appearances[appearances['date'].str.startswith(last\_season)]  
  
# Aggregate goals by player  
player\_goals = appearances\_last\_season.groupby('player\_id')['goals'].sum().reset\_index()  
player\_goals = player\_goals.sort\_values(by='goals', ascending=False).head(10)  
  
# Merge with players data for names  
player\_goals = player\_goals.merge(players[['player\_id', 'name']], on='player\_id')  
  
# Plot  
plt.figure(figsize=(12, 8))  
plt.barh(player\_goals['name'], player\_goals['goals'], color='skyblue')  
plt.xlabel('Total Goals')  
plt.title('Top 10 Players with Most Goals in the Last Season')  
plt.gca().invert\_yaxis()  
plt.show()



7. **Top 5 Clubs with Highest Average Attendance**

import pandas as pd  
import matplotlib.pyplot as plt  
  
# Load the data  
games = pd.read\_csv('games.csv')  
clubs = pd.read\_csv('clubs.csv')  
  
# Group by home and away clubs, and aggregate attendance  
home\_attendance = games.groupby('home\_club\_id')['attendance'].mean().reset\_index()  
away\_attendance = games.groupby('away\_club\_id')['attendance'].mean().reset\_index()  
  
# Combine and average attendance  
attendance = pd.concat([home\_attendance.rename(columns={'home\_club\_id': 'club\_id', 'attendance': 'average\_attendance'}),  
 away\_attendance.rename(columns={'away\_club\_id': 'club\_id', 'attendance': 'average\_attendance'})])  
attendance = attendance.groupby('club\_id')['average\_attendance'].mean().reset\_index()  
  
# Merge with club names  
attendance\_clubs = attendance.merge(clubs[['club\_id', 'name']], left\_on='club\_id', right\_on='club\_id')  
top\_clubs = attendance\_clubs.sort\_values(by='average\_attendance', ascending=False).head(5)  
  
# Plot  
plt.figure(figsize=(12, 8))  
plt.barh(top\_clubs['name'], top\_clubs['average\_attendance'], color='salmon')  
plt.xlabel('Average Attendance')  
plt.title('Top 5 Clubs with Highest Average Attendance')  
plt.gca().invert\_yaxis()  
plt.show()



8. **Comparison of Average Player Market Value by Nationality**

import pandas as pd  
import seaborn as sns  
import matplotlib.pyplot as plt  
  
# Load the data  
players = pd.read\_csv('players.csv')  
  
# Aggregate average market value by nationality  
market\_value\_by\_nationality = players.groupby('country\_of\_citizenship')['market\_value\_in\_eur'].mean().reset\_index()  
  
# Sort and plot  
market\_value\_by\_nationality = market\_value\_by\_nationality.sort\_values(by='market\_value\_in\_eur', ascending=False).head(10)  
  
plt.figure(figsize=(14, 8))  
sns.barplot(data=market\_value\_by\_nationality, x='country\_of\_citizenship', y='market\_value\_in\_eur', palette='coolwarm')  
plt.title('Comparison of Average Player Market Value by Nationality')  
plt.xlabel('Nationality')  
plt.ylabel('Average Market Value (EUR)')  
plt.xticks(rotation=45)  
plt.show()

