EPL 2020 - 2021
Data Analysis

Importing the required libraries

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

Loading the dataset

In [2]:
```
epl_df = pd.read_csv("EPL_20_21.csv")
epl_df.head()
```

Out[2]:

	Name	Club	Nationality	Position	Age	Matches	Starts	Mins	Goals	Assists	Passes_Attempte
0	Mason Mount	Chelsea	ENG	MF,FW	21	36	32	2890	6	5	188
1	Edouard Mendy	Chelsea	SEN	GK	28	31	31	2745	0	0	100
2	Timo Werner	Chelsea	GER	FW	24	35	29	2602	6	8	82
3	Ben Chilwell	Chelsea	ENG	DF	23	27	27	2286	3	5	180
4	Reece James	Chelsea	ENG	DF	20	32	25	2373	1	2	198

Exploring the columns

In [3]:
```
epl_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 532 entries, 0 to 531
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Name                   532 non-null    object
 1   Club                   532 non-null    object
 2   Nationality            532 non-null    object
 3   Position               532 non-null    object
 4   Age                    532 non-null    int64
 5   Matches                532 non-null    int64
 6   Starts                 532 non-null    int64
 7   Mins                   532 non-null    int64
 8   Goals                  532 non-null    int64
 9   Assists                532 non-null    int64
 10  Passes_Attempted       532 non-null    int64
 11  Perc_Passes_Completed  532 non-null    float64
 12  Penalty_Goals          532 non-null    int64
 13  Penalty_Attempted      532 non-null    int64
 14  xG                     532 non-null    float64
 15  xA                     532 non-null    float64
 16  Yellow_Cards           532 non-null    int64
 17  Red_Cards              532 non-null    int64
dtypes: float64(3), int64(11), object(4)
memory usage: 74.9+ KB
```

Getting some summary statistics of our dataset

In [4]:
```
epl_df.describe()
```

Out[4]:

	Age	Matches	Starts	Mins	Goals	Assists	Passes_Attempted	Perc_
count	532.000000	532.000000	532.000000	532.000000	532.000000	532.000000	532.000000	
mean	25.500000	19.535714	15.714286	1411.443609	1.853383	1.287594	717.750000	
std	4.319404	11.840459	11.921161	1043.171856	3.338009	2.095191	631.372522	
min	16.000000	1.000000	0.000000	1.000000	0.000000	0.000000	0.000000	
25%	22.000000	9.000000	4.000000	426.000000	0.000000	0.000000	171.500000	
50%	26.000000	21.000000	15.000000	1345.000000	1.000000	0.000000	573.500000	
75%	29.000000	30.000000	27.000000	2303.500000	2.000000	2.000000	1129.500000	
max	38.000000	38.000000	38.000000	3420.000000	23.000000	14.000000	3214.000000	

Checking the number of null values

In [5]:
```
epl_df.isna().sum()
```

Out[5]:
```
Name                       0
Club                       0
Nationality                0
Position                   0
Age                        0
Matches                    0
Starts                     0
Mins                       0
Goals                      0
Assists                    0
Passes_Attempted           0
Perc_Passes_Completed      0
Penalty_Goals              0
Penalty_Attempted          0
xG                         0
xA                         0
Yellow_Cards               0
Red_Cards                  0
dtype: int64
```

Creating two new columns named Mins_Per_Match and Goals_Per_Match

In [6]:
```
epl_df["Mins_Per_Match"] = (epl_df["Mins"] / epl_df["Matches"]).astype(int)
epl_df["Goals_Per_Match"] = (epl_df["Goals"] / epl_df["Matches"]).astype(float)
```

In [7]:
```
epl_df[["Name","Mins_Per_Match","Goals_Per_Match"]].head()
```

Out[7]:

	Name	Mins_Per_Match	Goals_Per_Match
0	Mason Mount	80	0.166667
1	Edouard Mendy	88	0.000000
2	Timo Werner	74	0.171429
3	Ben Chilwell	84	0.111111
4	Reece James	74	0.031250

Total Goals in the season

In [8]:
```
Total_Goals = epl_df["Goals"].sum()
print(f"The total goals scored in EPL 2020-21 season : {Total_Goals}")
```

```
The total goals scored in EPL 2020-21 season : 986
```

Total Penalty Goals

In [9]:
```
Total_Penalty_Goals = epl_df["Penalty_Goals"].sum()
print(f"The total penalty goals scored in EPL 2020-21 season : {Total_Penalty_Goals}")
```

```
The total penalty goals scored in EPL 2020-21 season : 102
```

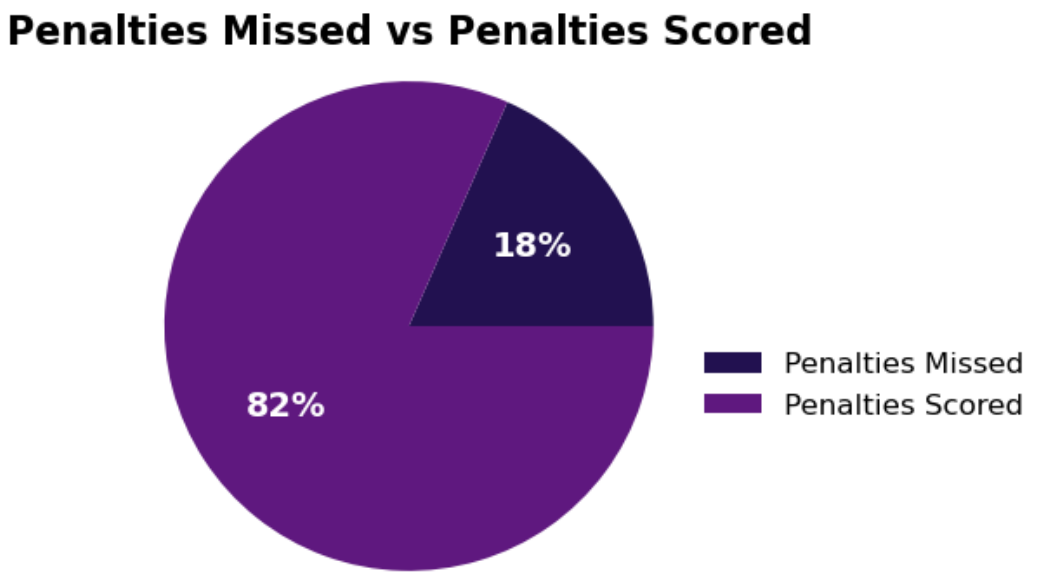
Total Penalty Attempts

In [10]:
```
Total_Penalty_Attempts = epl_df["Penalty_Attempted"].sum()
print(f"The total penalty attempts in EPL 2020-21 season : {Total_Penalty_Attempts}")
```

```
The total penalty attempts in EPL 2020-21 season : 125
```

Distribution of penalties missed vs penalties scored

In [11]:
```
penalty_not_scored = epl_df["Penalty_Attempted"].sum() - Total_Penalty_Goals

data = [penalty_not_scored, Total_Penalty_Goals]
labels = ["Penalties Missed", "Penalties Scored"]

plt.figure(figsize=(4, 4))
color_palette = sns.color_palette("magma")
plt.pie(data, labels=labels, colors=color_palette, autopct="%.0f%%",
        textprops={'fontsize': 14, 'fontweight': 'bold','color':'white'})
plt.title("Penalties Missed vs Penalties Scored", fontsize=16, fontweight='bold')
plt.legend(loc='best', labels=labels, fontsize=12, frameon=False, bbox_to_anchor=(1, 0.5
plt.axis('equal')

plt.show()
```



Total number of unique positions

In [12]:
```
unique_positions = epl_df["Position"].unique()
print(f"The list of unique positions in this season : {unique_positions}")
```

```
The list of unique positions in this season : ['MF,FW' 'GK' 'FW' 'DF' 'MF' 'FW,MF' 'FW,D
F' 'DF,MF' 'MF,DF' 'DF,FW']
```

Total FW players

In [13]:
```
epl_df[(epl_df["Position"] == "FW") | (epl_df["Position"] == "MF,FW") | (epl_df["Positic
       (epl_df["Position"] == "FW,DF") | (epl_df["Position"] == "DF,FW")]
```

Out[13]:

	Name	Club	Nationality	Position	Age	Matches	Starts	Mins	Goals	Assists	Passes_Atten
0	Mason Mount	Chelsea	ENG	MF,FW	21	36	32	2890	6	5	
2	Timo Werner	Chelsea	GER	FW	24	35	29	2602	6	8	
12	Christian Pulisic	Chelsea	USA	FW,MF	21	27	18	1738	4	2	
13	Kai Havertz	Chelsea	GER	MF,FW	21	27	18	1520	4	3	
15	Hakim Ziyech	Chelsea	MAR	FW,MF	27	23	15	1172	2	3	
...	...	...	...	...	...	...	...	...	...	...	
519	Rhian Brewster	Sheffield United	ENG	FW	20	27	12	1128	0	0	
523	Billy Sharp	Sheffield United	ENG	FW	34	16	7	735	3	0	
526	Daniel Jebbison	Sheffield United	ENG	FW	17	4	3	284	1	0	
527	Lys Mousset	Sheffield United	FRA	FW,MF	24	11	2	296	0	0	
530	Antwoine Hackford	Sheffield United	ENG	DF,FW	16	1	0	11	0	0	

176 rows × 20 columns

176 players played in FW position

In [14]:
```
epl_df[(epl_df["Position"] == "FW")]
```

Out[14]:

	Name	Club	Nationality	Position	Age	Matches	Starts	Mins	Goals	Assists	Passes_Att
2	Timo Werner	Chelsea	GER	FW	24	35	29	2602	6	8	
16	Tammy Abraham	Chelsea	ENG	FW	22	22	12	1040	6	1	
19	Olivier Giroud	Chelsea	FRA	FW	33	17	8	748	4	0	
23	Ruben Loftus-Cheek	Chelsea	ENG	FW	24	1	1	60	0	0	
30	Raheem Sterling	Manchester City	ENG	FW	25	31	28	2536	10	7	
...	...	...	...	...	...	...	...	...	...	...	
516	Oliver Burke	Sheffield United	SCO	FW	23	25	14	1269	1	1	
518	Oliver McBurnie	Sheffield United	SCO	FW	24	23	12	1324	1	0	
519	Rhian Brewster	Sheffield United	ENG	FW	20	27	12	1128	0	0	
523	Billy Sharp	Sheffield United	ENG	FW	34	16	7	735	3	0	
526	Daniel Jebbison	Sheffield United	ENG	FW	17	4	3	284	1	0	

81 rows × 20 columns

81 players played in only in FW position

Nations from which different players belong to.

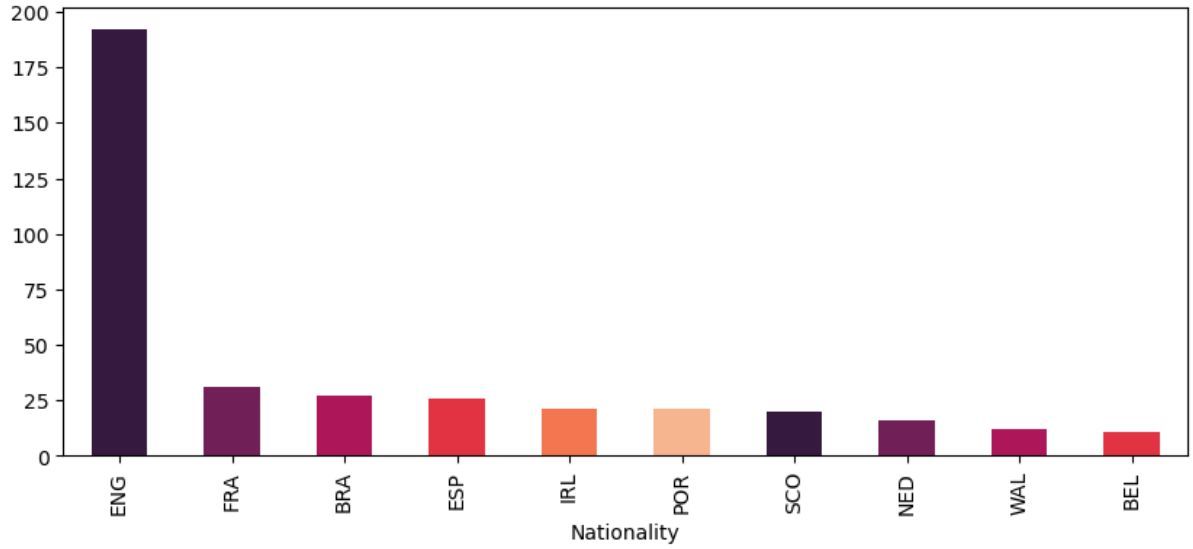
In [15]:
```
nation_count = np.size(epl_df["Nationality"].unique())
```

In [16]:
```
print(f"Number of nations from which the players come from : {nation_count}")
```

```
Number of nations from which the players come from : 59
```

Analysing from which country the maximum players come from

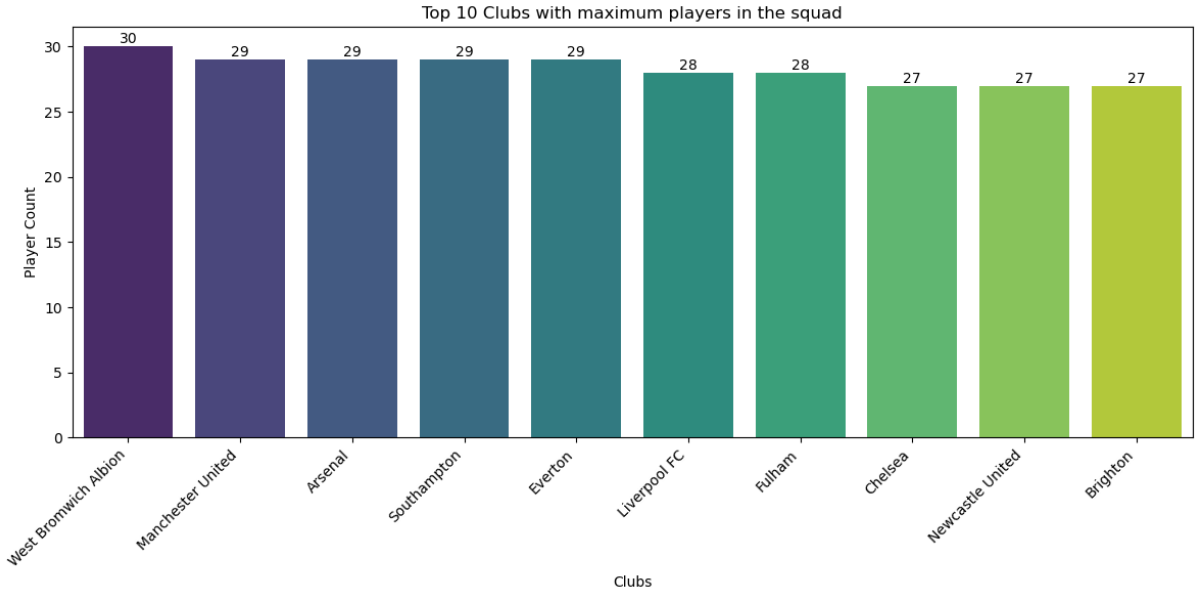
In [17]:
```
nationality = epl_df.groupby("Nationality").size().sort_values(ascending = False)
nationality.head(10).plot(kind = "bar", figsize = (10,4), color = sns.color_palette("roc
plt.show()
```



Clubs with maximum players in their squad

In [18]:
```
top_10_clubs = epl_df["Club"].value_counts().nlargest(10)
```

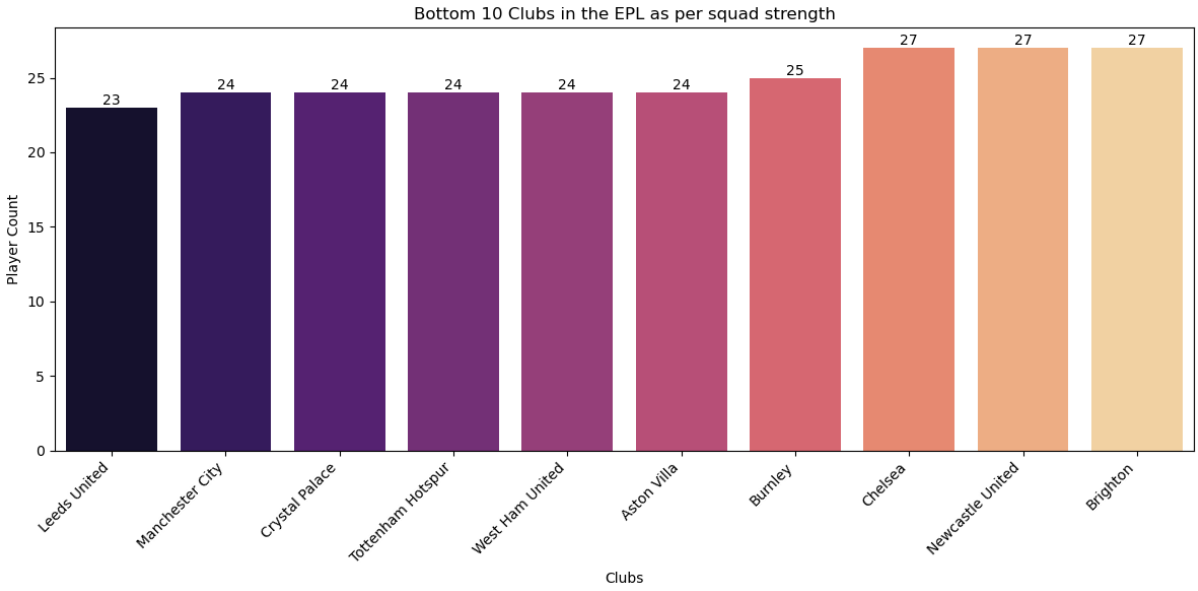
In [19]:
```
plt.figure(figsize=(12, 6))
ax = sns.barplot(x=top_10_clubs.index, y=top_10_clubs, palette="viridis")

# Adding data labels
for p in ax.patches:
    ax.annotate(f'{p.get_height():.0f}', (p.get_x() + p.get_width() / 2., p.get_height()

plt.xticks(rotation=45, ha='right')

plt.xlabel("Clubs")
plt.ylabel("Player Count")
plt.title("Top 10 Clubs with maximum players in the squad")

plt.tight_layout()  # Adjust the layout to prevent overlapping labels
plt.show()
```



Clubs with minimum players in their squad

In [20]:
```
lowest_10_clubs = epl_df["Club"].value_counts().nsmallest(10)
```

In [21]:
```
plt.figure(figsize=(12, 6))
ax = sns.barplot(x=lowest_10_clubs.index, y=lowest_10_clubs, palette="magma")

# Adding data labels
for p in ax.patches:
    ax.annotate(f'{p.get_height():.0f}', (p.get_x() + p.get_width() / 2., p.get_height()

# Tilting x-axis labels by 45 degrees
plt.xticks(rotation=45, ha='right')

# Adding labels and title
plt.xlabel("Clubs")
plt.ylabel("Player Count")
plt.title("Bottom 10 Clubs in the EPL as per squad strength")

plt.tight_layout()  # Adjust the layout to prevent overlapping labels
plt.show()
```

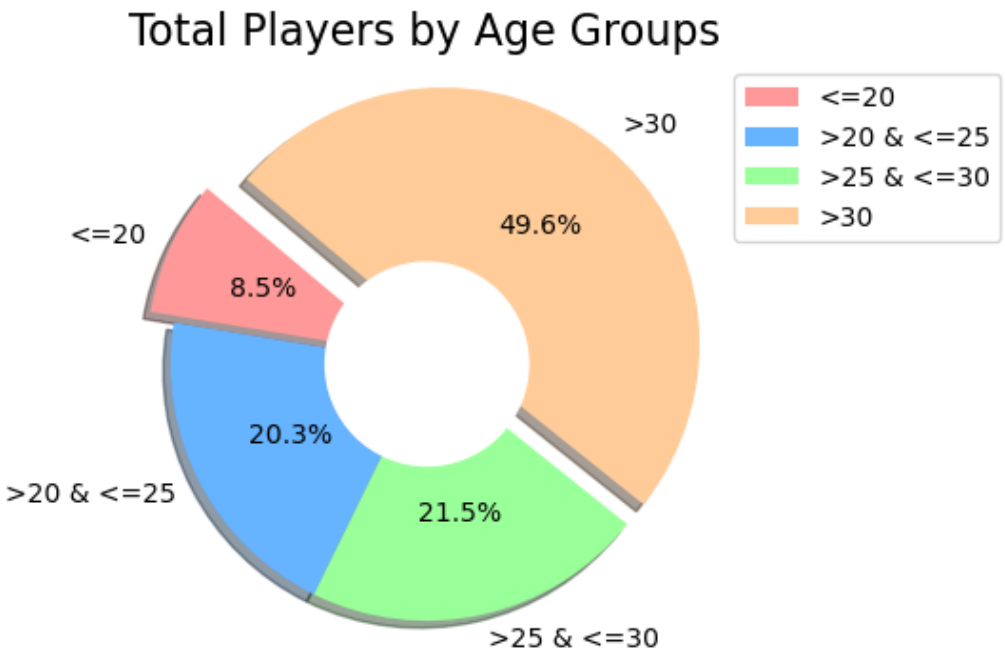


Players analysis as per their age

In [22]:
```
under20 = epl_df[epl_df["Age"] <= 20]
age20_25 = epl_df[(epl_df["Age"] > 20) & (epl_df["Age"] <= 25)]
age25_30 = epl_df[(epl_df["Age"] > 25) & (epl_df["Age"] <= 30)]
above30 = epl_df[epl_df["Age"] > 20]
```

In [23]:
```
x = np.array([under20["Name"].count(), age20_25["Name"].count(), age25_30["Name"].count(
mylabels = ["<=20", ">20 & <=25", ">25 & <=30", ">30"]
colors = ["#ff9999", "#66b3ff", "#99ff99", "#ffcc99"]
explode = [0.1, 0, 0, 0.1]       # Explode the first and last slices for emphasis

plt.figure(figsize=(4,4))
plt.title("Total Players by Age Groups", fontsize=16)

# Creating the pie chart
plt.pie(x, labels=mylabels, colors=colors, autopct="%.1f%%", explode=explode, shadow=Tru
```

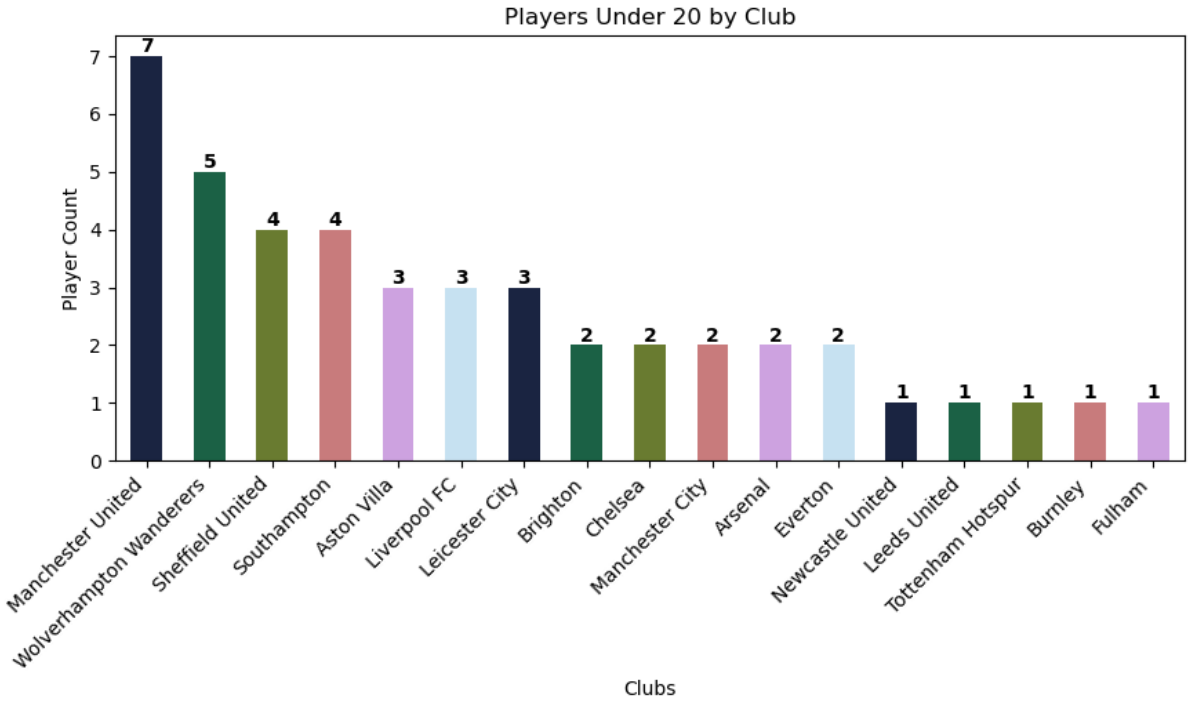
```
# Drawing a circle in the center to make it a donut chart
centre_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

# Adding a legend
plt.legend(labels=mylabels, loc=9, bbox_to_anchor=(1, 0, 0.5, 1))

plt.axis("equal")            # Equal aspect ratio ensures the pie chart is drawn as a ci

plt.show()
```



Total under 20 players in each club

In [24]:
```
players_under_20 = epl_df[epl_df["Age"] < 20]
```

In [25]:
```
plt.figure(figsize=(10, 4))
ax = players_under_20["Club"].value_counts().plot(kind="bar", color=sns.color_palette("c

plt.bar_label(ax.containers[0], label_type='edge', fontsize=10, fontweight='bold')

plt.xlabel("Clubs")
plt.ylabel("Player Count")
plt.title("Players Under 20 by Club")
plt.xticks(rotation=45, ha='right')

plt.show()
```



Under 20 players in ManU

In [26]:
```
players_under_20[players_under_20["Club"] == "Manchester United"]
```

Out[26]:

	Name	Club	Nationality	Position	Age	Matches	Starts	Mins	Goals	Assists	Passes_At
61	Mason Greenwood	Manchester United	ENG	FW	18	31	21	1822	7	2	
72	Brandon Williams	Manchester United	ENG	DF	19	4	2	188	0	0	
73	Amad Diallo	Manchester United	CIV	FW	18	3	2	166	0	1	
74	Anthony Elanga	Manchester United	SWE	FW	18	2	2	155	1	0	
76	Shola Shoretire	Manchester United	ENG	FW	16	2	0	11	0	0	
78	Hannibal Mejbri	Manchester United	FRA	MF	17	1	0	9	0	0	
79	William Thomas Fish	Manchester United	ENG	DF	17	1	0	1	0	0	

Under 20 players in Arsenal

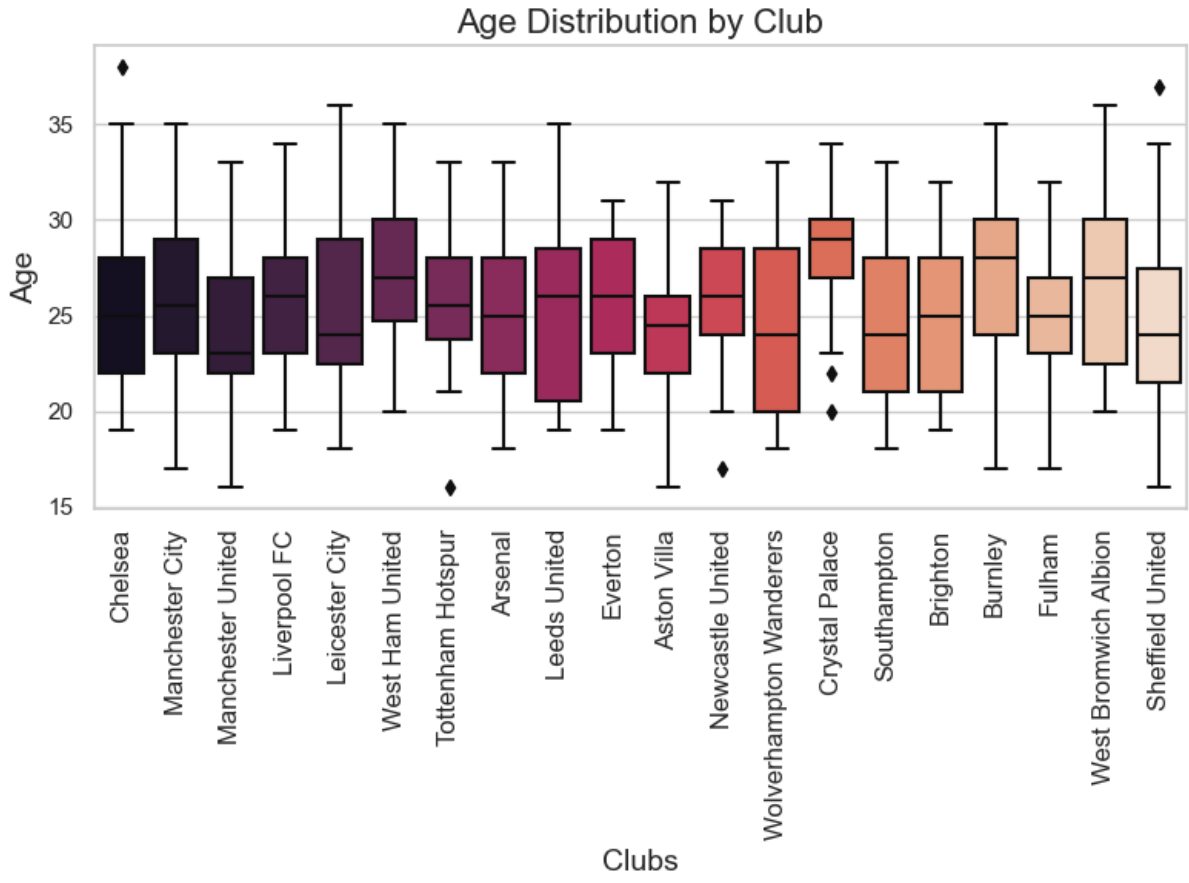
In [27]:
```
players_under_20[players_under_20["Club"] == "Arsenal"]
```

Out[27]:

	Name	Club	Nationality	Position	Age	Matches	Starts	Mins	Goals	Assists	Passes_Attemp
184	Bukayo Saka	Arsenal	ENG	FW,DF	18	32	30	2553	5	3	1
203	Martinelli	Arsenal	BRA	FW	19	14	7	589	2	1	

Let's view the average age of players in each club

In [28]:
```
plt.figure(figsize=(8, 6))
sns.set_theme(style="whitegrid")
color_palette = sns.color_palette("rocket", n_colors=len(epl_df["Club"].unique()))
sns.boxplot(x="Club", y="Age", data=epl_df, palette=color_palette, orient="v")
plt.title("Age Distribution by Club", fontsize=16)
plt.xlabel("Clubs", fontsize=14)
plt.ylabel("Age", fontsize=14)
plt.xticks(rotation = 90, fontsize=12)
plt.tight_layout()
plt.show()
```



Total assists from each club

In [29]:
```
assists_by_clubs = pd.DataFrame(epl_df.groupby("Club", as_index = False)["Assists"].sum(
```

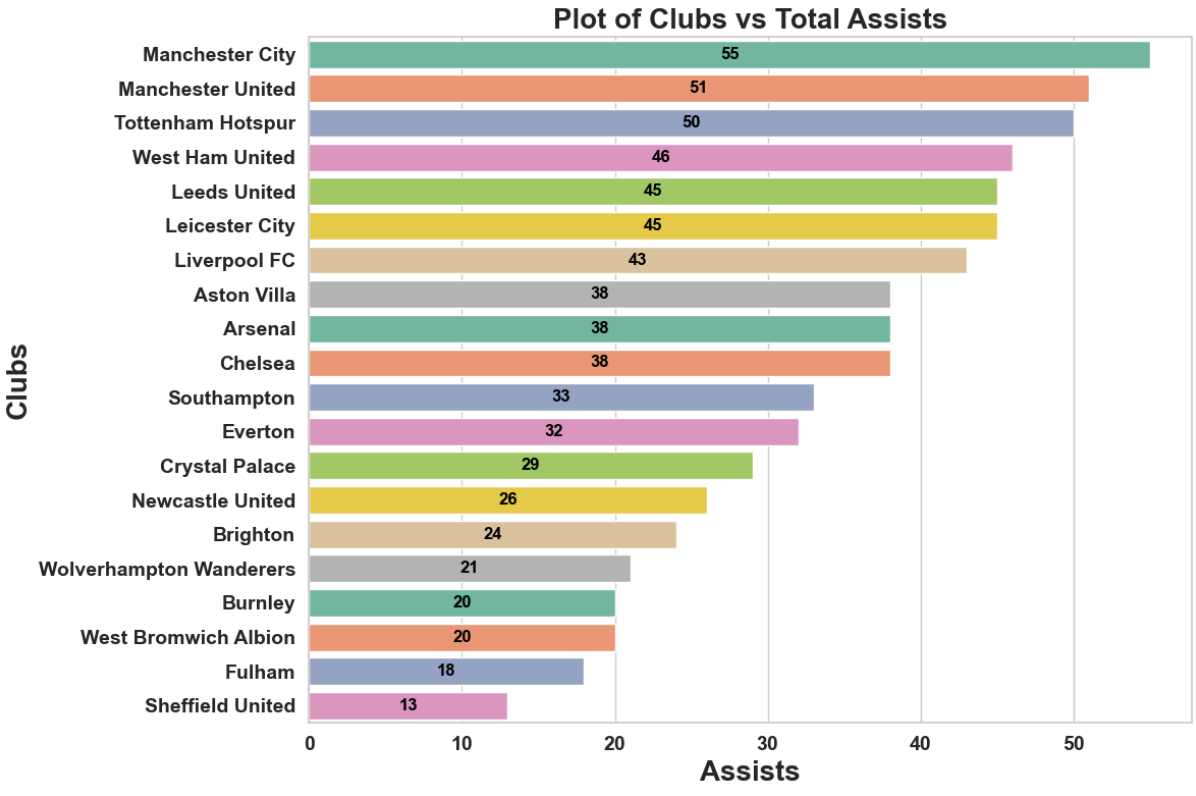
In [30]:
```
assists_by_clubs.head()
```

Out[30]:

	Club	Assists
0	Arsenal	38
1	Aston Villa	38
2	Brighton	24
3	Burnley	20
4	Chelsea	38

In [31]:

```
plt.figure(figsize=(12, 8))
sns.set_theme(style="whitegrid", color_codes=True)

# Sortinge data by assists in descending order
assists_by_clubs_sorted = assists_by_clubs.sort_values(by="Assists", ascending=False)

color_palette = sns.color_palette("Set2", n_colors=len(assists_by_clubs_sorted))
ax = sns.barplot(x="Assists", y="Club", data=assists_by_clubs_sorted, palette=color_pale

# Add data labels inside the bars
for p, label in zip(ax.patches, assists_by_clubs_sorted["Assists"]):
    ax.annotate(f'{label}', (p.get_width() / 2, p.get_y() + p.get_height() / 2),
                ha='center', va='center', fontsize=12, fontweight='bold', color='black')

ax.set_xlabel("Assists", fontsize=20, fontweight='bold')
ax.set_ylabel("Clubs", fontsize=20, fontweight='bold')
plt.xticks(fontsize=14, fontweight='bold')
plt.yticks(fontsize=14, fontweight='bold')

plt.title("Plot of Clubs vs Total Assists", fontsize=20, fontweight='bold')

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



Top 10 assists

In [32]:
```
top_10_assists = epl_df[["Name","Club","Assists","Matches"]].nlargest(n = 10, columns =
top_10_assists
```

Out[32]:

	Name	Club	Assists	Matches
162	Harry Kane	Tottenham Hotspur	14	35
34	Kevin De Bruyne	Manchester City	12	25
51	Bruno Fernandes	Manchester United	12	37
161	Son Heung-min	Tottenham Hotspur	10	37
273	Jack Grealish	Aston Villa	10	26
54	Marcus Rashford	Manchester United	9	37
110	Jamie Vardy	Leicester City	9	34
220	Raphael Dias Belloli	Leeds United	9	30
2	Timo Werner	Chelsea	8	35
136	Aaron Cresswell	West Ham United	8	36

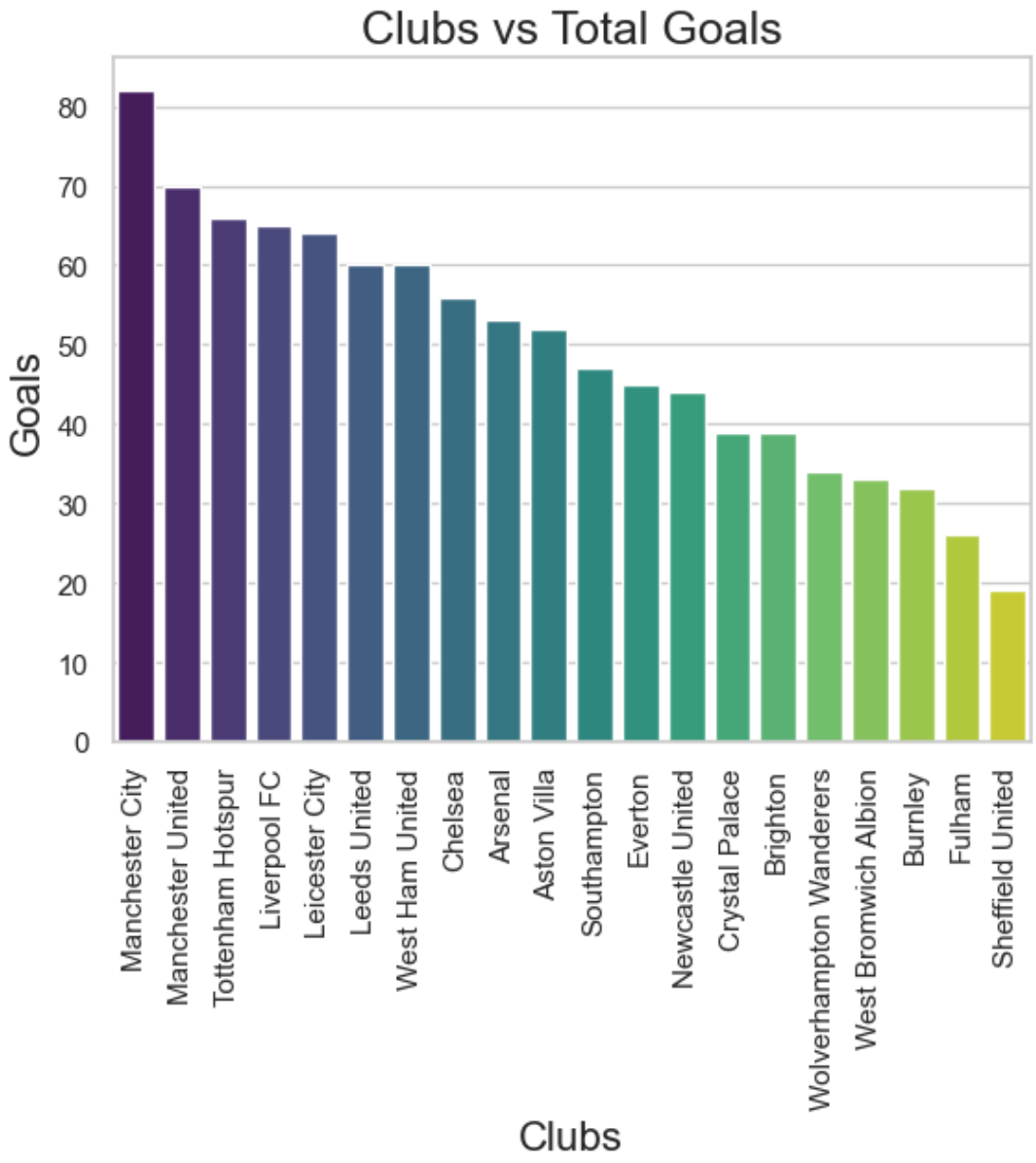
Goals by Clubs

In [33]:
```
goals_by_club = pd.DataFrame(epl_df.groupby("Club", as_index = False)["Goals"].sum())
goals_by_club.head(3)
```

Out[33]:

	Club	Goals
0	Arsenal	53
1	Aston Villa	52
2	Brighton	39

In [34]:
```
sns.set_theme(style = "whitegrid", color_codes = True)
ax =sns.barplot(x = "Club", y ="Goals", data = goals_by_club.sort_values(by = "Goals", a
ax.set_xlabel("Clubs", fontsize = 16)
ax.set_ylabel("Goals", fontsize = 16)
plt.xticks(rotation = 90, fontsize = 11)
plt.rcParams["figure.figsize"] = (10,8)
plt.title("Clubs vs Total Goals", fontsize = 18)
plt.show()
```



Most goals by players

In [35]:
```
top_10_goals = epl_df[["Name","Club","Goals","Matches"]].nlargest(n = 10, columns = "Goa
top_10_goals
```

Out[35]:

	Name	Club	Goals	Matches
162	Harry Kane	Tottenham Hotspur	23	35
81	Mohamed Salah	Liverpool FC	22	37
51	Bruno Fernandes	Manchester United	18	37
161	Son Heung-min	Tottenham Hotspur	17	37
214	Patrick Bamford	Leeds United	17	38
237	Dominic Calvert-Lewin	Everton	16	33
110	Jamie Vardy	Leicester City	15	34
267	Ollie Watkins	Aston Villa	14	37
33	İlkay Gündoğan	Manchester City	13	28
191	Alexandre Lacazette	Arsenal	13	31

Goals per Match

In [36]:
```
top_10_goals_per_match = epl_df[["Name","Goals_Per_Match","Matches","Goals"]].nlargest(n
top_10_goals_per_match
```

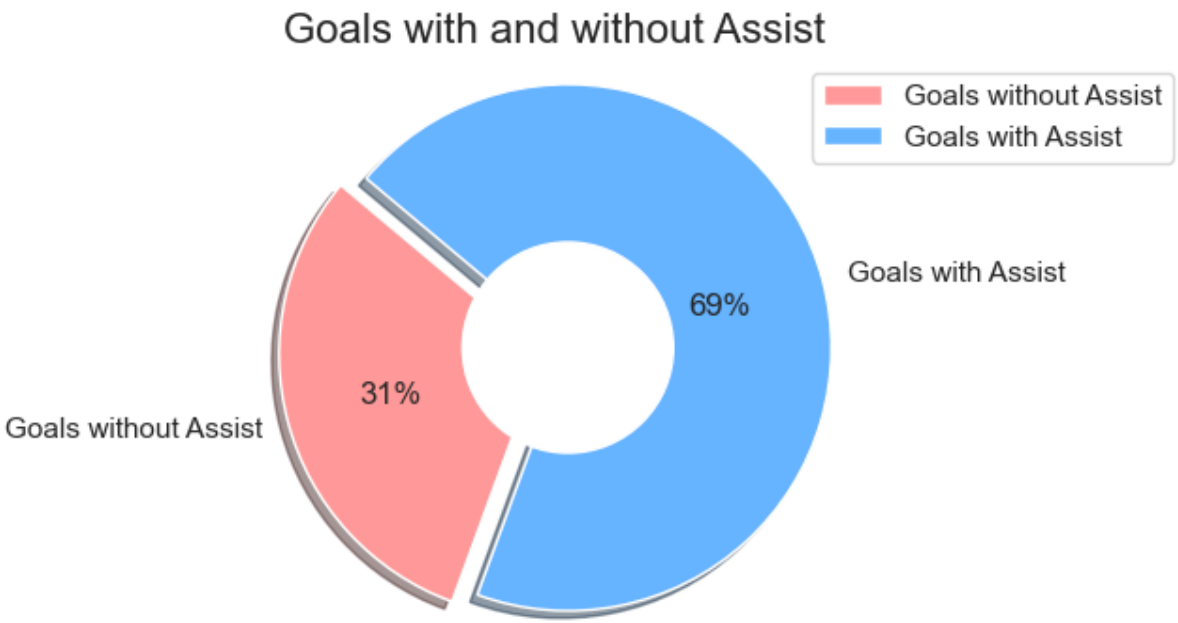
Out[36]:

	Name	Goals_Per_Match	Matches	Goals
162	Harry Kane	0.657143	35	23
81	Mohamed Salah	0.594595	37	22
307	Joe Willock	0.571429	14	8
145	Jesse Lingard	0.562500	16	9
175	Gareth Bale	0.550000	20	11
74	Anthony Elanga	0.500000	2	1
51	Bruno Fernandes	0.486486	37	18
237	Dominic Calvert-Lewin	0.484848	33	16
120	Kelechi Iheanacho	0.480000	25	12
92	Diogo Jota	0.473684	19	9

Goals with assist and without assist

In [37]:
```
plt.figure(figsize = (4, 4))
plt.title("Goals with and without Assist", fontsize=16)

assists = epl_df["Assists"].sum()
data = [Total_Goals - assists, assists]
labels = ["Goals without Assist", "Goals with Assist"]
colors = ["#ff9999", "#66b3ff"]
explode = [0.1, 0]          # Exploding the first slice for emphasis

plt.pie(data, labels=labels, colors=colors, autopct="%.0f%%", explode=explode, shadow=Tr
```

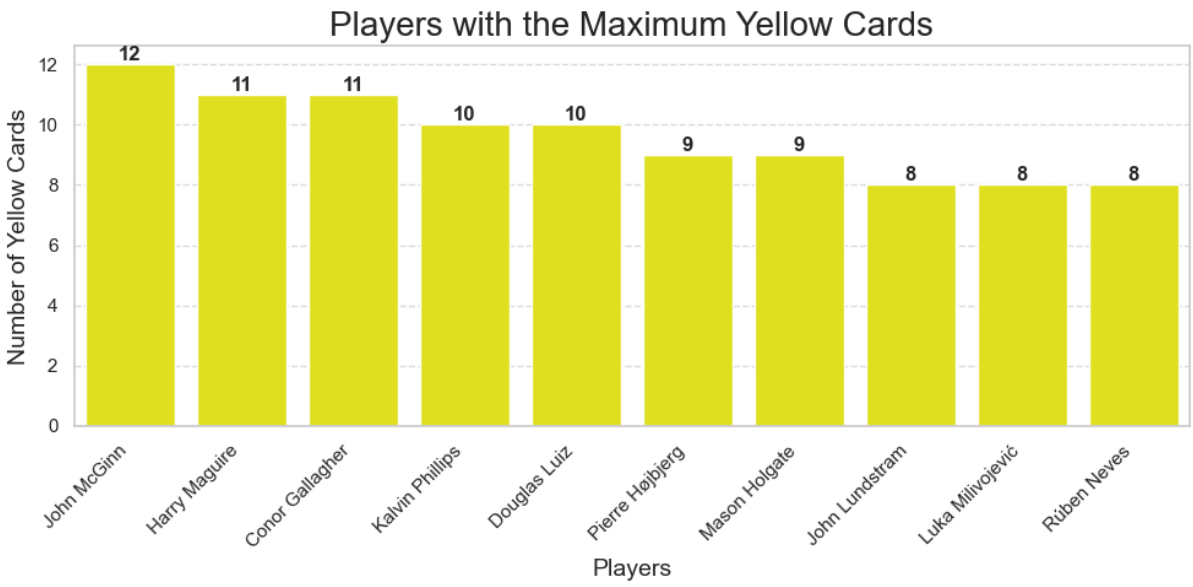
```
# Drawing a circle in the center to make it look like a donut chart
centre_circle = plt.Circle((0, 0), 0.4, fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

# Adding legend
plt.legend(labels = labels, loc =9 , bbox_to_anchor=(1, 0, 0.5, 1))

plt.axis("equal")  # Equal aspect ratio ensures the pie chart is drawn as a circle.

plt.show()
```



Top 10 players with maximum yellow cards

In [38]:
```
epl_yellow = epl_df.sort_values(by = "Yellow_Cards", ascending = False)[ :10]
```

In [39]:
```
plt.figure(figsize=(10, 5))
plt.title("Players with the Maximum Yellow Cards", fontsize=20)
barplot_yellow = sns.barplot(x = epl_yellow["Name"], y = epl_yellow["Yellow_Cards"], col

for p in barplot_yellow.patches:
    barplot_yellow.annotate(f'{p.get_height():.0f}', (p.get_x() + p.get_width() / 2., p.
                            ha='center', va='bottom', fontsize=12, fontweight='bold')

plt.ylabel("Number of Yellow Cards", fontsize=14)
plt.xlabel("Players", fontsize=14)
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



Top 10 players with maximum red cards

In [40]:
```
epl_red = epl_df.sort_values(by = "Red_Cards", ascending = False)[ :10]
```

In [41]:
```
plt.figure(figsize=(10, 5))
plt.title("Players with the Maximum Red Cards", fontsize=20)
barplot_red = sns.barplot(x = epl_red["Name"], y = epl_red["Red_Cards"], color = "red")

for p in barplot_yellow.patches:
    barplot_red.annotate(f'{p.get_height():.0f}', (p.get_x() + p.get_width() / 2., p.get
                         ha='center', va='bottom', fontsize=12, fontweight='bold')

plt.ylabel("Number of Red Cards", fontsize=14)
plt.xlabel("Players", fontsize=14)
plt.xticks(rotation=45, ha='right')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```