Introduction

Founded in 1904 to provide unity among national soccer associations, the Federation of Internationale de Football Association (FIFA) boasts 209 members, rivaling that of the United Nations, and is arguably the most prestigious sports organization in the world.

In this Data Science Project we will do some analysis on the matches and records of FIFA with Python.

Importing the Libraries ¶

In [1]:

```
## Importing the necessary Libraries
import numpy as np
import pandas as pd

## for visualizations
import matplotlib.pyplot as plt
import seaborn as sns
sns.set()
```

Importing the DataSet and Viewing it

In [2]:

```
## Importing the dataSet
data = pd.read_csv(r"C:\Users\lenovo\Desktop\Fifa Data.csv")
data.head()
```

Out[2]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cd

5 rows × 89 columns

Let’s Eye (Check) on Indian Footballers

In [3]:

```
## Defining the function
def country(x):
    return data[data['Nationality'] == x][['Name','Overall','Potential','Position

# let's check the Indian Players
country('India')
```

Out[3]:

	Name	Overall	Potential	Position
8605	S. Chhetri	67	67	LS
10011	S. Jhingan	65	71	RCB
12598	J. Lalpekhlua	63	64	RS
12811	G. Singh Sandhu	63	68	GK
13508	A. Edathodika	62	62	LCB
14054	P. Halder	61	67	RCM
14199	P. Kotal	61	66	RB
14218	L. Ralte	61	62	LW
14705	N. Das	60	65	LB
14786	U. Singh	60	67	RM
14915	H. Narzary	60	66	LM
15356	R. Singh	59	59	ST
15643	S. Singh	59	65	CB
15652	A. Thapa	59	71	LCM
15855	M. Rafique	58	61	CM
15864	A. Singh	58	62	GK
15884	B. Singh	58	58	ST
16135	S. Bose	58	66	LB
16265	R. Borges	58	60	CDM
16450	S. Paul	57	57	NaN
16499	A. Mondal	57	57	CB
16539	L. Lalruatthara	57	63	NaN
16793	E. Lyngdoh	56	56	NaN
16903	J. Lalrinzuala	56	64	LB
16976	A. Kuruniyan	56	70	LW
17129	J. Singh	55	58	NaN
17197	V. Kaith	55	64	GK
17339	S. Passi	54	63	NaN
17436	D. Lalhlimpuia	54	67	NaN
17539	C. Singh	53	62	NaN

Analyzing the Club Data (Manchester United)

In [4]:

```
## Definimng the function
def club(x):
    return data[data['Club'] == x][['Name','Jersey Number','Position','Overall','
                                    'Value','Contract Valid Until']]

club('Manchester United')
```

Out[4]:

	Name	Jersey Number	Position	Overall	Nationality	Age	Wage	Value	Contract Valid Until
3	De Gea	1.0	GK	91	Spain	27	€260K	€72M	2020
45	P. Pogba	6.0	RDM	87	France	25	€210K	€64M	2021
47	R. Lukaku	9.0	ST	87	Belgium	25	€230K	€62.5M	2022
93	A. Sánchez	7.0	RW	85	Chile	29	€215K	€37.5M	2022
116	A. Martial	11.0	LW	84	France	22	€165K	€42.5M	2019
132	N. Matić	31.0	CDM	84	Serbia	29	€165K	€24M	2020
211	Juan Mata	8.0	RM	83	Spain	30	€160K	€24.5M	2019
250	Fred	17.0	CM	82	Brazil	25	€140K	€26.5M	2023
254	J. Lingard	7.0	CAM	82	England	25	€140K	€26.5M	2021
319	M. Rashford	11.0	LW	81	England	20	€110K	€27M	2020
327	E. Bailly	2.0	CB	81	Ivory Coast	24	€105K	€21M	2020
374	Ander Herrera	21.0	CM	81	Spain	28	€140K	€17.5M	2019
377	C. Smalling	12.0	RCB	81	England	28	€130K	€16M	2019
399	A. Valencia	16.0	RM	81	Ecuador	32	€120K	€10M	2019
454	L. Shaw	23.0	LB	80	England	22	€96K	€16.5M	2023
526	S. Romero	1.0	GK	80	Argentina	31	€91K	€9M	2021
584	V. Lindelöf	3.0	CB	79	Sweden	23	€91K	€14.5M	2021
629	M. Rojo	16.0	CB	79	Argentina	28	€115K	€10M	2021
654	P. Jones	4.0	CB	79	England	26	€110K	€12M	2019
700	M. Fellaini	8.0	CM	79	Belgium	30	€120K	€11.5M	2020
717	A. Young	18.0	LB	79	England	32	€110K	€7M	2019
807	Andreas Pereira	15.0	CM	78	Brazil	22	€91K	€14M	2019
1313	M. Darmian	36.0	LB	76	Italy	28	€88K	€6M	2019
2561	L. Grant	13.0	GK	74	England	35	€39K	€1.3M	2020
3451	Diogo Dalot	20.0	RB	72	Portugal	19	€26K	€4.7M	2023
4513	S. McTominay	17.0	CM	71	Scotland	21	€43K	€3.8M	2021
8191	A. Gomes	47.0	CAM	67	England	17	€15K	€1.5M	2021
10087	T. Chong	44.0	RW	65	Netherlands	18	€13K	€1.1M	2019
10457	E. Hamilton	48.0	CM	65	Scotland	19	€11K	€1M	2020
10461	C. Gribbin	42.0	CAM	65	England	19	€11K	€1.2M	2019
11081	R. Poole	50.0	CB	64	Wales	20	€13K	€675K	2019
11422	R. Williams	52.0	CB	64	England	19	€8K	€875K	2019
12545	J. Bohui	46.0	ST	63	England	19	€10K	€700K	2019

In [5]:

```
## Checking for the shape and size of the data ( Manchester United )
x = club('Manchester United')
x.shape
```

Out[5]:

```
(33, 9)
```

Describing the data

In [6]:

```
## Checking for the values in the dataset
data.describe()
```

Out[6]:

	Unnamed: 0	ID	Age	Overall	Potential	Special	Interr Rep
count	18207.000000	18207.000000	18207.000000	18207.000000	18207.000000	18207.000000	18159.
mean	9103.000000	214298.338606	25.122206	66.238699	71.307299	1597.809908	1.
std	5256.052511	29965.244204	4.669943	6.908930	6.136496	272.586016	0.
min	0.000000	16.000000	16.000000	46.000000	48.000000	731.000000	1.
25%	4551.500000	200315.500000	21.000000	62.000000	67.000000	1457.000000	1.
50%	9103.000000	221759.000000	25.000000	66.000000	71.000000	1635.000000	1.
75%	13654.500000	236529.500000	28.000000	71.000000	75.000000	1787.000000	1.
max	18206.000000	246620.000000	45.000000	94.000000	95.000000	2346.000000	5.

8 rows × 44 columns

Checking for null values in dataset

In [7]:

```
data.isnull().sum()
```

Out[7]:

```
Unnamed: 0          0
ID                  0
Name                0
Age                 0
Photo               0
                 ...
GKHandling         48
GKKicking          48
GKPositioning      48
GKReflexes         48
Release Clause   1564
Length: 89, dtype: int64
```

Filling the missing value for the continuous variables for proper data visualization

In [8]:

```python
## Filling the null values
data['ShortPassing'].fillna(data['ShortPassing'].mean(), inplace = True)
data['Volleys'].fillna(data['Volleys'].mean(), inplace = True)
data['Dribbling'].fillna(data['Dribbling'].mean(), inplace = True)
data['Curve'].fillna(data['Curve'].mean(), inplace = True)
data['FKAccuracy'].fillna(data['FKAccuracy'], inplace = True)
data['LongPassing'].fillna(data['LongPassing'].mean(), inplace = True)
data['BallControl'].fillna(data['BallControl'].mean(), inplace = True)
data['HeadingAccuracy'].fillna(data['HeadingAccuracy'].mean(), inplace = True)
data['Finishing'].fillna(data['Finishing'].mean(), inplace = True)
data['Crossing'].fillna(data['Crossing'].mean(), inplace = True)
data['Weight'].fillna('200lbs', inplace = True)
data['Contract Valid Until'].fillna(2019, inplace = True)
data['Height'].fillna("5'11", inplace = True)
data['Loaned From'].fillna('None', inplace = True)
data['Joined'].fillna('Jul 1, 2018', inplace = True)
data['Jersey Number'].fillna(8, inplace = True)
data['Body Type'].fillna('Normal', inplace = True)
data['Position'].fillna('ST', inplace = True)
data['Club'].fillna('No Club', inplace = True)
data['Work Rate'].fillna('Medium/ Medium', inplace = True)
data['Skill Moves'].fillna(data['Skill Moves'].median(), inplace = True)
data['Weak Foot'].fillna(3, inplace = True)
data['Preferred Foot'].fillna('Right', inplace = True)
data['International Reputation'].fillna(1, inplace = True)
data['Wage'].fillna('€200K', inplace = True)
data.fillna(0, inplace = True)
```

In [9]:

```
## Defining the required functions and calling it
def defending(data):
    return int(round((data[['Marking', 'StandingTackle', 
                               'SlidingTackle']].mean()).mean()))

def general(data):
    return int(round((data[['HeadingAccuracy', 'Dribbling', 'Curve', 
                               'BallControl']].mean()).mean()))

def mental(data):
    return int(round((data[['Aggression', 'Interceptions', 'Positioning', 
                               'Vision','Composure']].mean()).mean()))

def passing(data):
    return int(round((data[['Crossing', 'ShortPassing', 
                               'LongPassing']].mean()).mean()))

def mobility(data):
    return int(round((data[['Acceleration', 'SprintSpeed', 
                               'Agility','Reactions']].mean()).mean()))
def power(data):
    return int(round((data[['Balance', 'Jumping', 'Stamina', 
                               'Strength']].mean()).mean()))

def rating(data):
    return int(round((data[['Potential', 'Overall']].mean()).mean()))

def shooting(data):
    return int(round((data[['Finishing', 'Volleys', 'FKAccuracy', 
                               'ShotPower','LongShots', 'Penalties']].mean()).mea
```

Renaming the columns

In [10]:

```
## Renaming the column
data.rename(columns={'Club Logo':'Club_Logo'}, inplace=True)

## Adding these categories to the data

data['Defending'] = data.apply(defending, axis = 1)
data['General'] = data.apply(general, axis = 1)
data['Mental'] = data.apply(mental, axis = 1)
data['Passing'] = data.apply(passing, axis = 1)
data['Mobility'] = data.apply(mobility, axis = 1)
data['Power'] = data.apply(power, axis = 1)
data['Rating'] = data.apply(rating, axis = 1)
data['Shooting'] = data.apply(shooting, axis = 1)
```

In [11]:

```
## Fitting the data & Checking only the required columns
players = data[['Name','Defending','General','Mental','Passing',
                'Mobility','Power','Rating','Shooting','Flag','Age',
                'Nationality', 'Photo', 'Club_Logo', 'Club']]

players.head()
```

Out[11]:

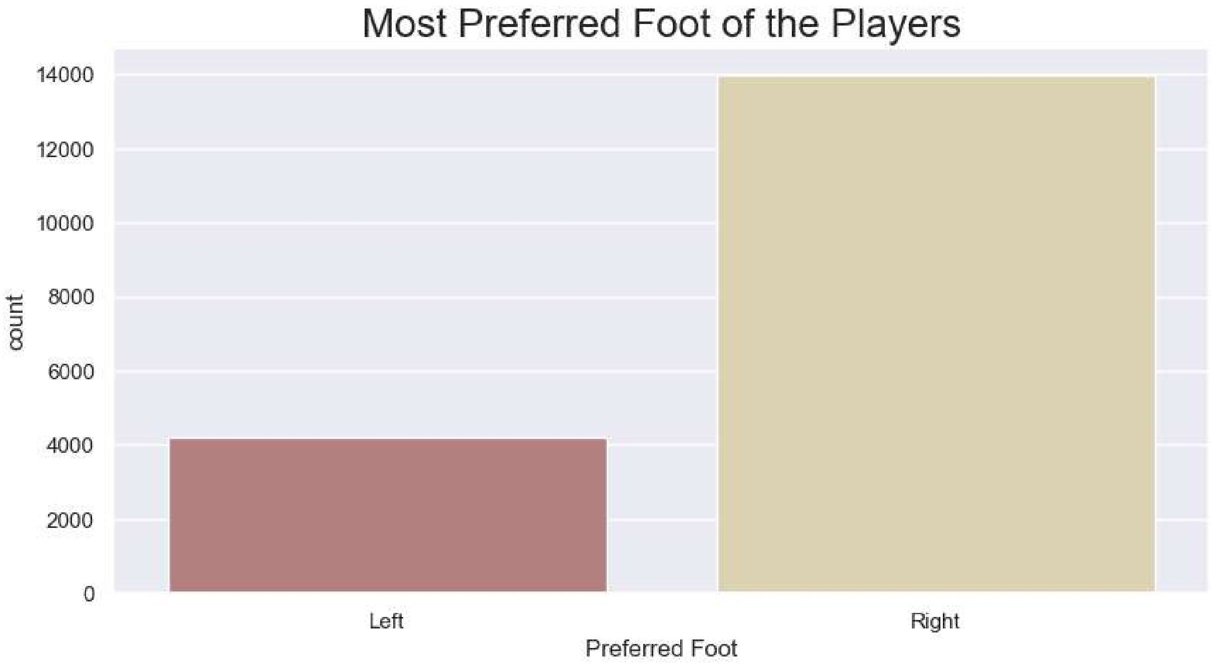
	Name	Defending	General	Mental	Passing	Mobility	Power	Rating	Shooting	
0	L. Messi	29	89	71	87	91	74	94	88	https://cdn.sofifa
1	Cristiano Ronaldo	27	88	73	81	91	83	94	88	https://cdn.sofifa
2	Neymar Jr	28	85	72	80	94	69	92	84	https://cdn.sofifa
3	De Gea	16	26	43	39	66	54	92	21	https://cdn.sofifa
4	K. De Bruyne	59	79	81	92	81	76	92	85	https://cdn.sofif

Data Visualization

Comparison of preferred foot over the different players

In [12]:
```
plt.rcParams['figure.figsize'] = (10, 5)
sns.countplot(data['Preferred Foot'], palette = 'pink')
plt.title('Most Preferred Foot of the Players', fontsize = 20)
plt.show()
```

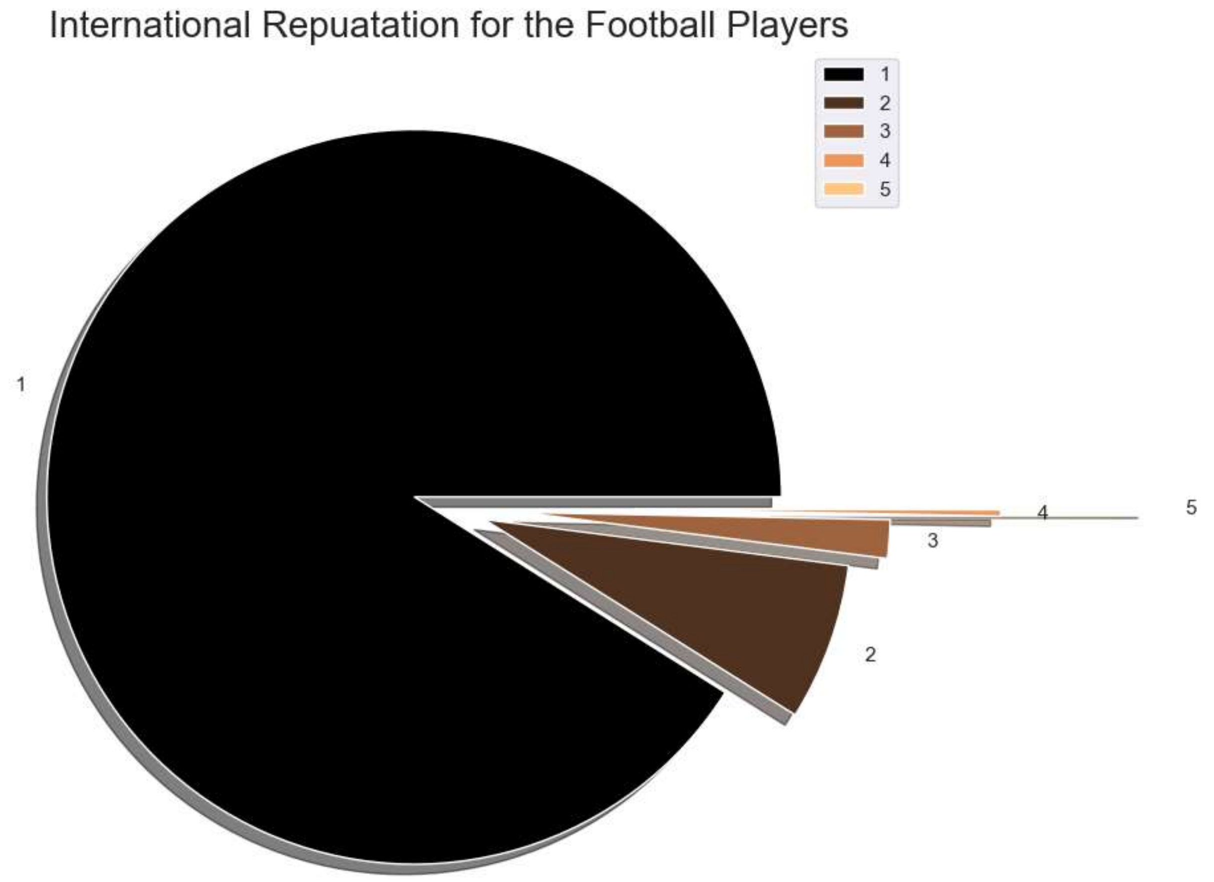
```
C:\Users\lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments without
an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



Plotting a pie chart to represent share of international reputation

In [13]:

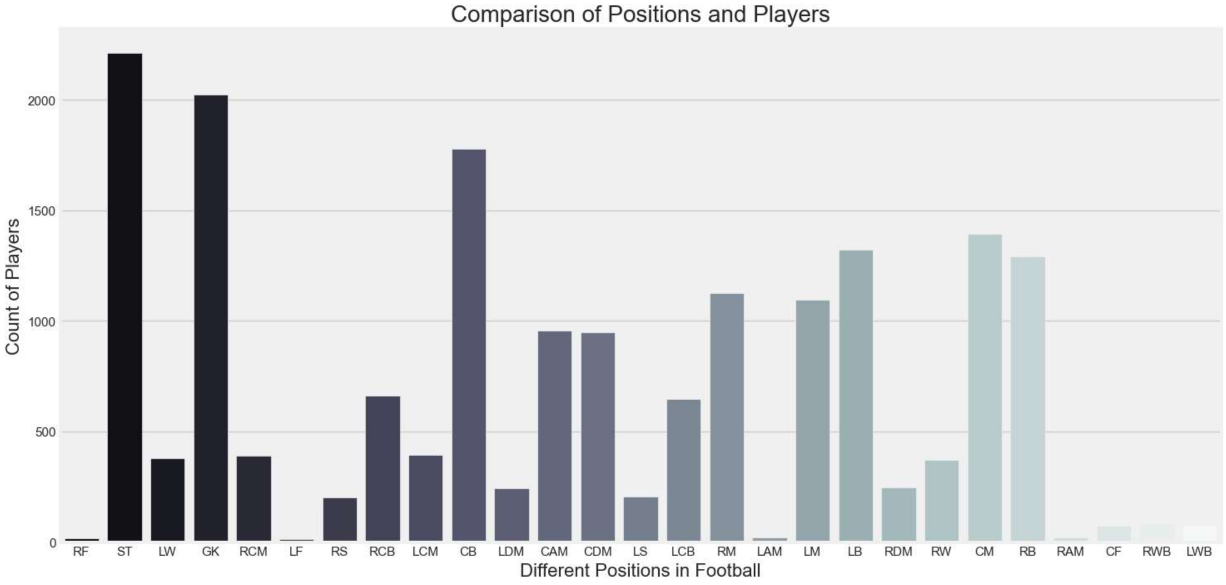
```
## Plotting pie Chart
labels = ['1', '2', '3', '4', '5']
sizes = data['International Reputation'].value_counts()
colors = plt.cm.copper(np.linspace(0, 1, 5))
explode = [0.1, 0.1, 0.2, 0.5, 0.9]

plt.rcParams['figure.figsize'] = (9, 9)
plt.pie(sizes, labels = labels, colors = colors, explode = explode, shadow = True
plt.title('International Repuatation for the Football Players', fontsize = 20)
plt.legend()
plt.show()
```



Different positions acquired by the players

In [14]:
```
## Plotting Bar graph for position acquired by players
plt.figure(figsize = (18, 8))
plt.style.use('fivethirtyeight')
ax = sns.countplot('Position', data = data, palette = 'bone')
ax.set_xlabel(xlabel = 'Different Positions in Football', fontsize = 16)
ax.set_ylabel(ylabel = 'Count of Players', fontsize = 16)
ax.set_title(label = 'Comparison of Positions and Players', fontsize = 20)
plt.show()
```

```
C:\Users\lenovo\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the on
ly valid positional argument will be `data`, and passing other arguments without
an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



Defining a function for cleaning the Weight data

In [15]:
```
## Defining the Function
def extract_value_from(value):
    out = value.replace('lbs', '')
    return float(out)

## Applying the function to weight column
## Data['value'] = data['value'].apply(lambda x: extract_value_from(x))
data['Weight'] = data['Weight'].apply(lambda x : extract_value_from(x))

data['Weight'].head()
```

Out[15]:
```
0    159.0
1    183.0
2    150.0
3    168.0
4    154.0
Name: Weight, dtype: float64
```

Defining a function for cleaning the wage column

In [16]:

```
## Defining the function
def extract_value_from(Value):
    out = Value.replace('€', '')
    if 'M' in out:
        out = float(out.replace('M', ''))*1000000
    elif 'K' in Value:
        out = float(out.replace('K', ''))*1000
    return float(out)
```

Applying the function to the wage column

In [17]:

```
## Applying the above defined function
data['Value'] = data['Value'].apply(lambda x: extract_value_from(x))
data['Wage'] = data['Wage'].apply(lambda x: extract_value_from(x))

data['Wage'].head()
```

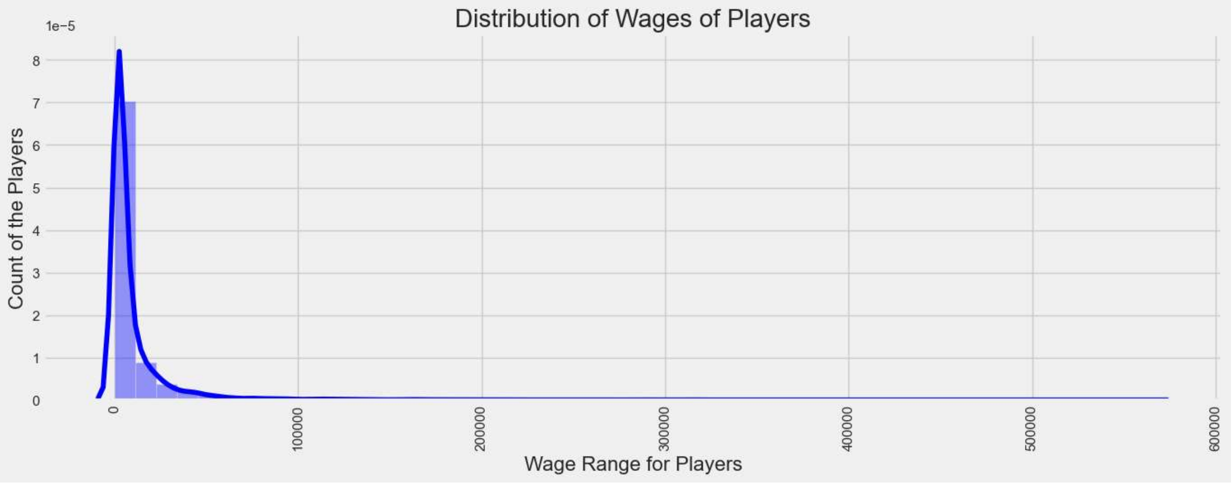
Out[17]:

```
0    565000.0
1    405000.0
2    290000.0
3    260000.0
4    355000.0
Name: Wage, dtype: float64
```

Comparing the players’ Wages

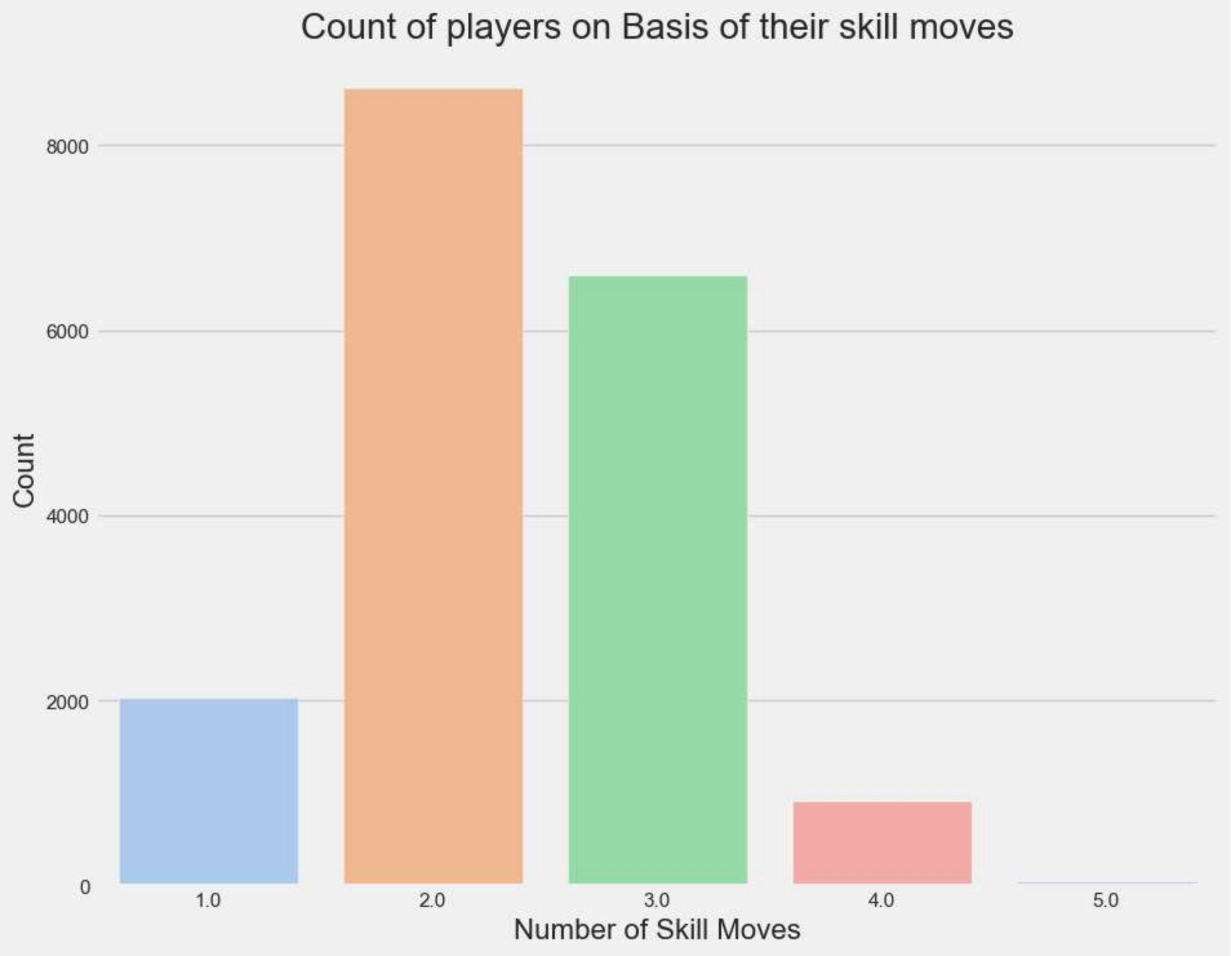
In [18]:

```
## Importing the library
import warnings
warnings.filterwarnings('ignore')

## Plotting the Graph for checking distribution of Player Wages
plt.rcParams['figure.figsize'] = (15, 5)
sns.distplot(data['Wage'], color = 'blue')
plt.xlabel('Wage Range for Players', fontsize = 16)
plt.ylabel('Count of the Players', fontsize = 16)
plt.title('Distribution of Wages of Players', fontsize = 20)
plt.xticks(rotation = 90)
plt.show()
```



Skill Moves of Players

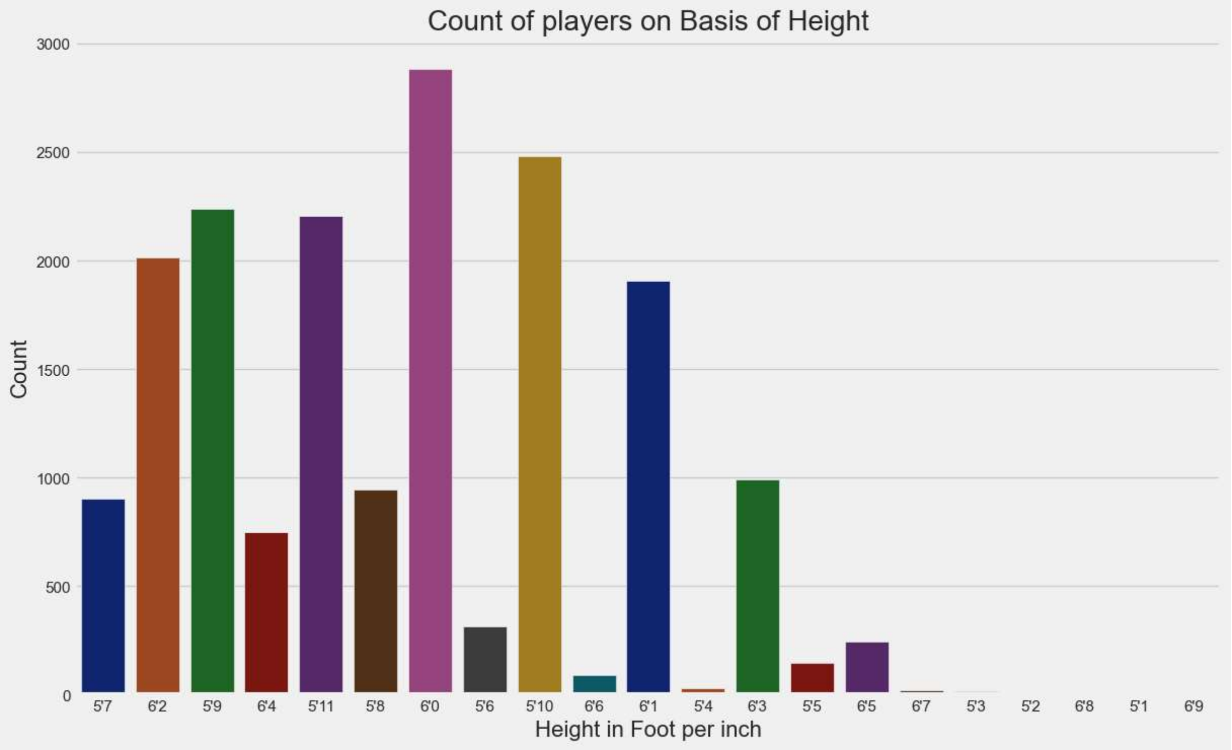
```
In [19]: ## Plotting graph for skilled Moves of Players
plt.figure(figsize = (10, 8))
ax = sns.countplot(x = 'Skill Moves', data = data, palette = 'pastel')
ax.set_title(label = 'Count of players on Basis of their skill moves', fontsize =
ax.set_xlabel(xlabel = 'Number of Skill Moves', fontsize = 16)
ax.set_ylabel(ylabel = 'Count', fontsize = 16)
plt.show()
```



Height of Players

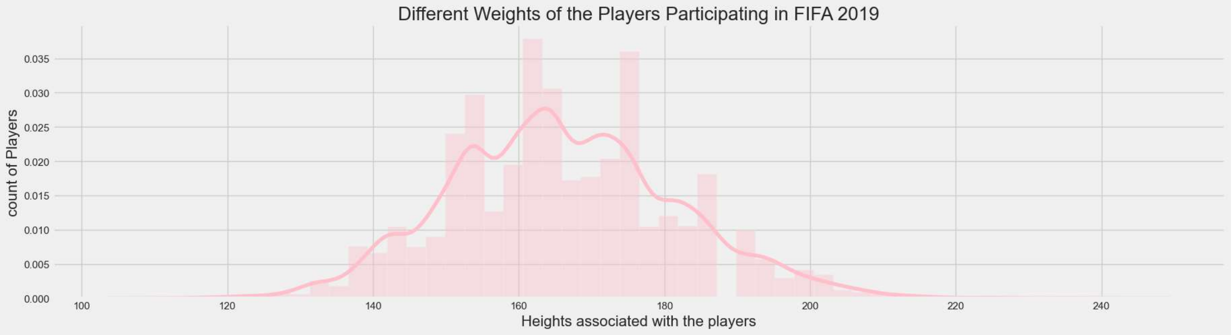
In [20]:

```
## Plotting the bar graph on the basic of their height
plt.figure(figsize = (13, 8))
ax = sns.countplot(x = 'Height', data = data, palette = 'dark')
ax.set_title(label = 'Count of players on Basis of Height', fontsize = 20)
ax.set_xlabel(xlabel = 'Height in Foot per inch', fontsize = 16)
ax.set_ylabel(ylabel = 'Count', fontsize = 16)
plt.show()
```



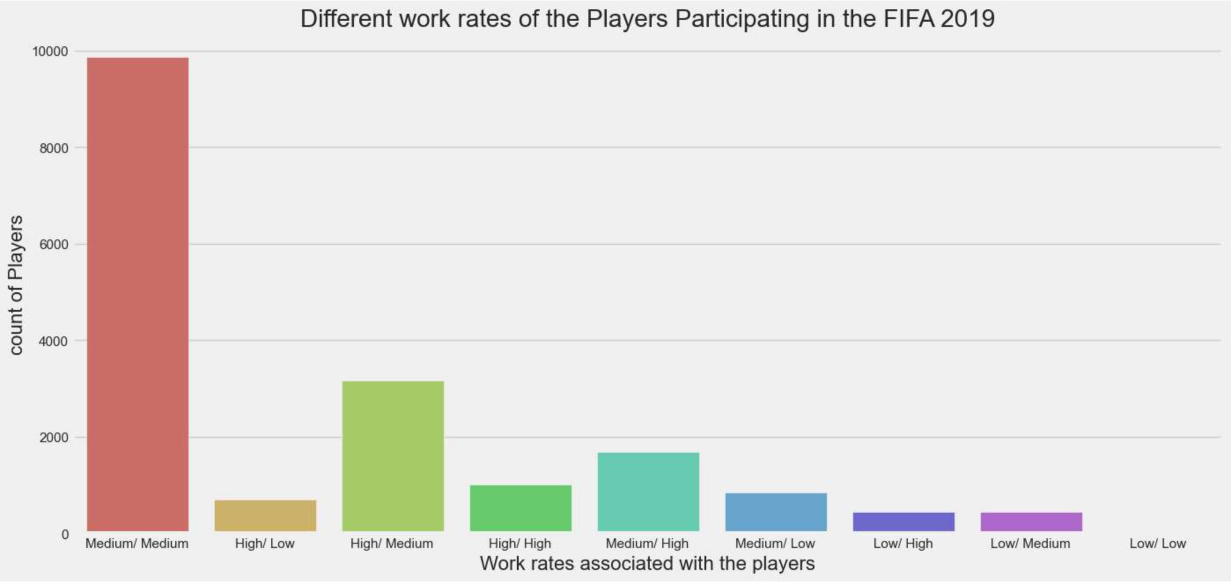
To show Different body weight of the players participating in the FIFA 2019

In [21]:
```
## Plotting the graph on the basic of body weight of the Players
plt.figure(figsize = (20, 5))
sns.distplot(data['Weight'], color = 'pink')
plt.title('Different Weights of the Players Participating in FIFA 2019', fontsize
plt.xlabel('Heights associated with the players', fontsize = 16)
plt.ylabel('count of Players', fontsize = 16)
plt.show()
```



To show Different Work rate of the players participating in the FIFA 2019

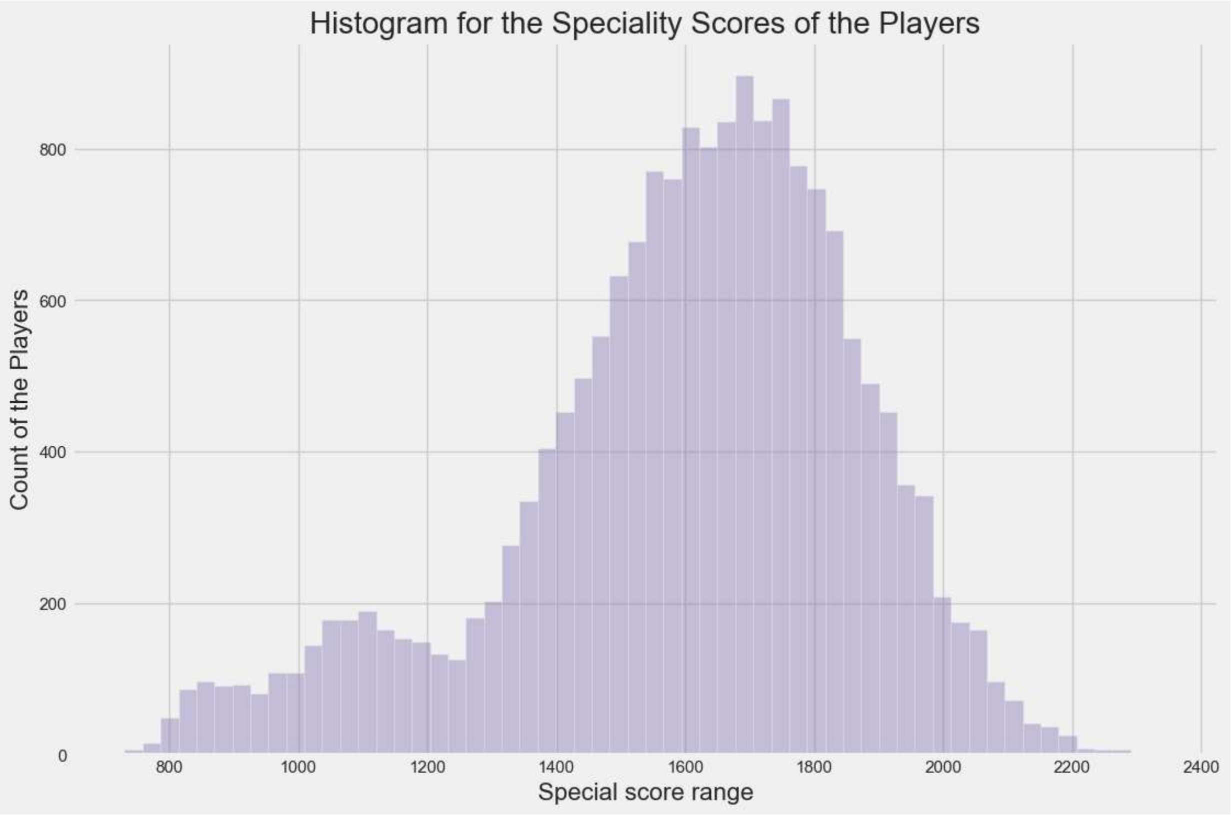
In [22]:
```
## Plotting the Graph on the basic of Work Rate of players
plt.figure(figsize = (15, 7))

sns.countplot(x = 'Work Rate', data = data, palette = 'hls')
plt.title('Different work rates of the Players Participating in the FIFA 2019', f
plt.xlabel('Work rates associated with the players', fontsize = 16)
plt.ylabel('count of Players', fontsize = 16)
plt.show()
```



To show Different Speciality Score of the players participating in the FIFA 2019

In [23]:

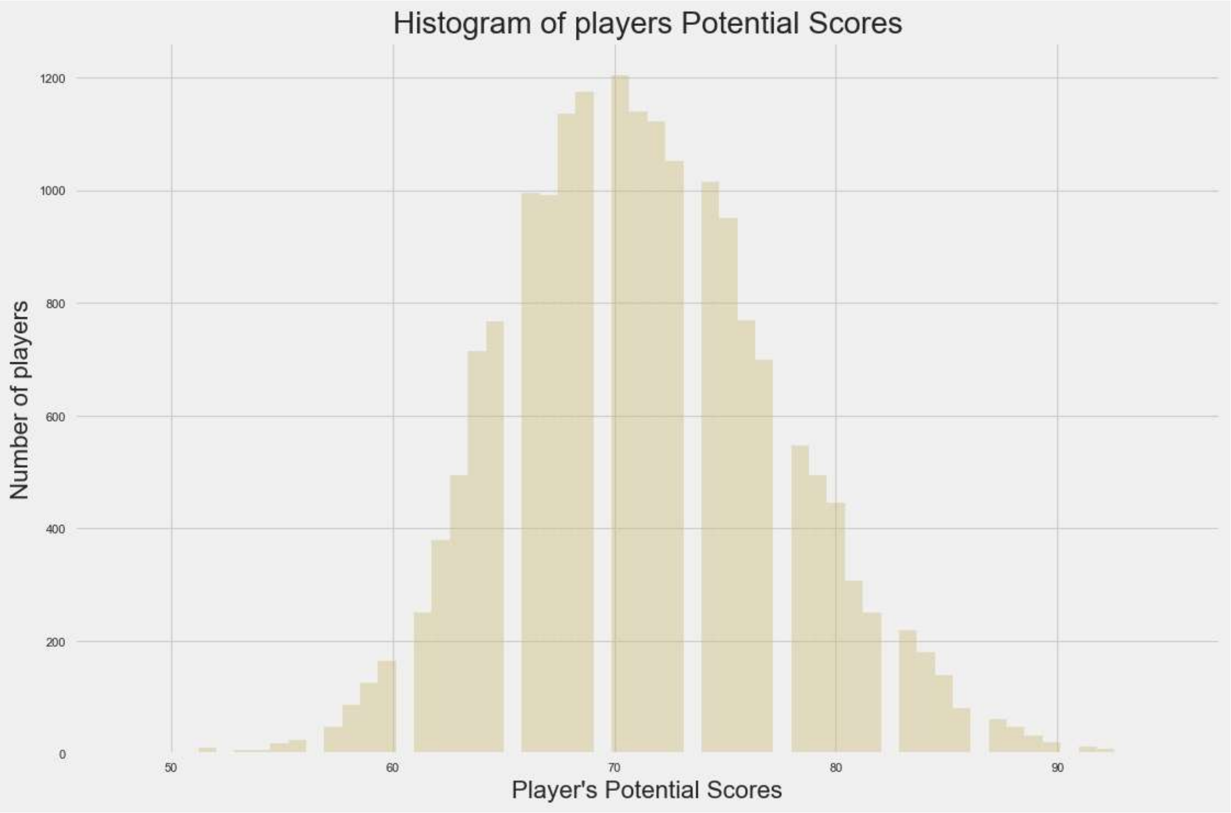
```
## Plotting of Histogram on the basic of Different Speciality Score
x = data.Special
plt.figure(figsize = (12, 8))
plt.style.use('tableau-colorblind10')

ax = sns.distplot(x, bins = 58, kde = False, color = 'm')
ax.set_xlabel(xlabel = 'Special score range', fontsize = 16)
ax.set_ylabel(ylabel = 'Count of the Players',fontsize = 16)
ax.set_title(label = 'Histogram for the Speciality Scores of the Players', fontsi
plt.show()
```



To show Different potential scores of the players participating in the FIFA 2019

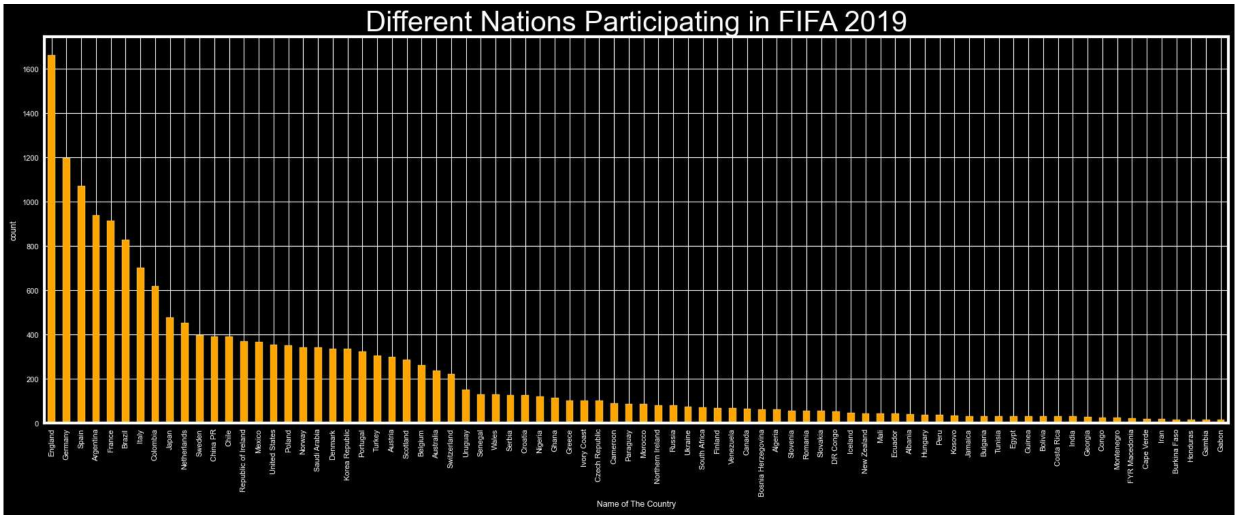
In [24]:

```
## Plotting Histogram for Different Potential Scores of Players
x = data.Potential
plt.figure(figsize=(12,8))
plt.style.use('seaborn-paper')

ax = sns.distplot(x, bins = 58, kde = False, color = 'y')
ax.set_xlabel(xlabel = "Player\'s Potential Scores", fontsize = 16)
ax.set_ylabel(ylabel = 'Number of players', fontsize = 16)
ax.set_title(label = 'Histogram of players Potential Scores', fontsize = 20)
plt.show()
```



To show Different nations participating in the FIFA 2019

In [25]:
```
## Bar Graph showing the Nations participated in FIFA 2019
plt.style.use('dark_background')
data['Nationality'].value_counts().head(80).plot.bar(color = 'orange', figsize =
plt.title('Different Nations Participating in FIFA 2019', fontsize = 30, fontweigl
plt.xlabel('Name of The Country')
plt.ylabel('count')
plt.show()
```



Countries with Most Players

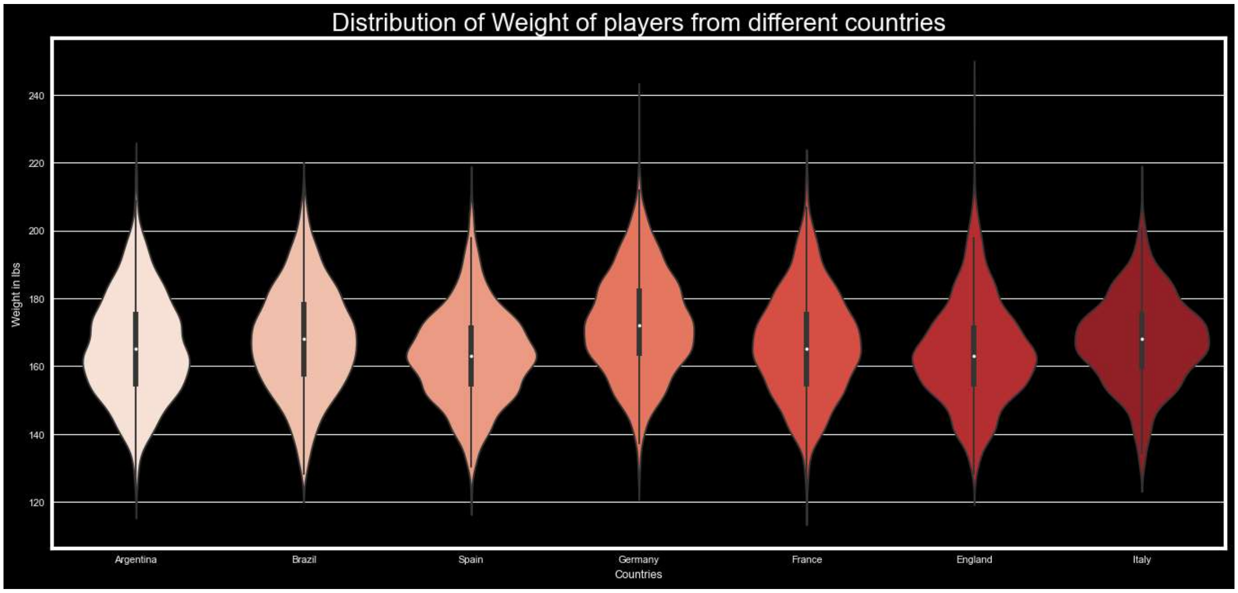
Picking up the countries with highest number of players to compare their overall scores

In [26]:
```
## Counting the Number of players
data['Nationality'].value_counts().head(8)
```

Out[26]:
```
England      1662
Germany      1198
Spain        1072
Argentina     937
France        914
Brazil        827
Italy         702
Colombia      618
Name: Nationality, dtype: int64
```

Every Nations’ Player and their Weights

In [27]:
```
## Graph showing the player's Nation and Weight
some_countries = ('England', 'Germany', 'Spain', 'Argentina', 'France', 'Brazil',
data_countries = data.loc[data['Nationality'].isin(some_countries) & data['Weight

plt.rcParams['figure.figsize'] = (15, 7)
ax = sns.violinplot(x = data_countries['Nationality'], y = data_countries['Weight
ax.set_xlabel(xlabel = 'Countries', fontsize = 9)
ax.set_ylabel(ylabel = 'Weight in lbs', fontsize = 9)
ax.set_title(label = 'Distribution of Weight of players from different countries'
plt.show()
```



Finding out the popular clubs around the globe

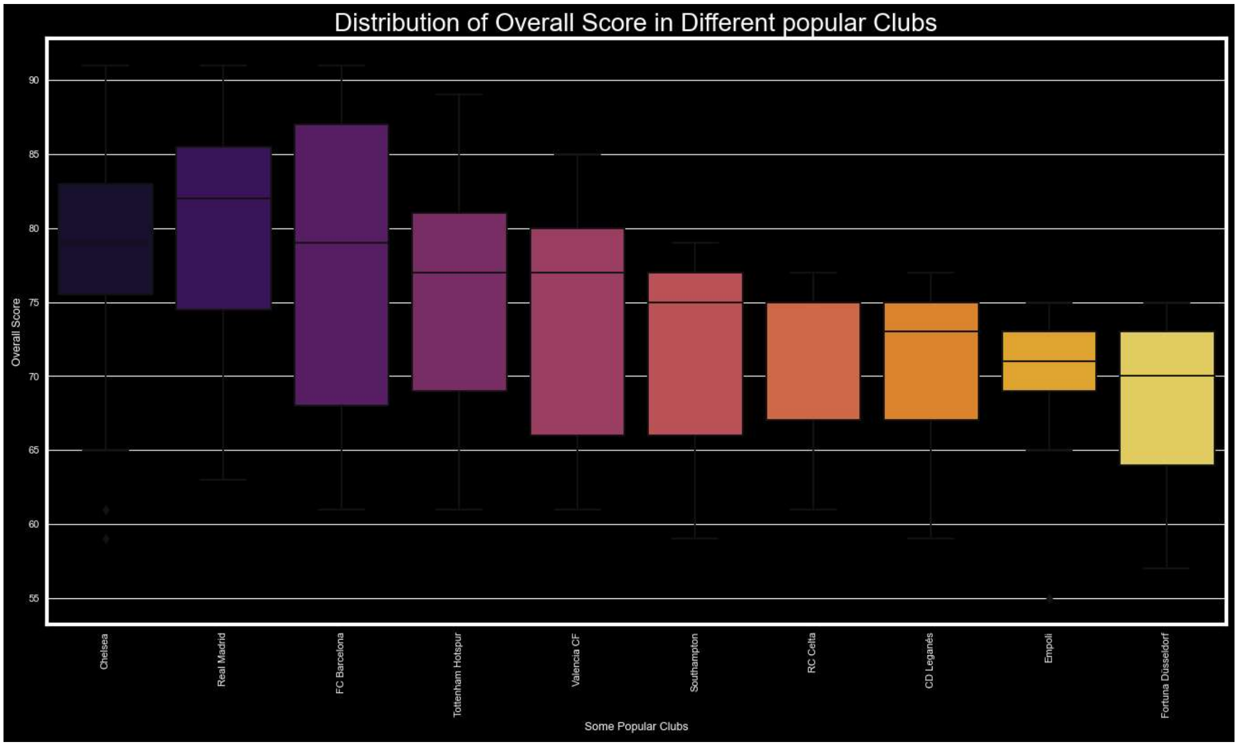
In [28]:
```
## Most Popular club of all
data['Club'].value_counts().head(10)
```

Out[28]:
```
No Club                    241
FC Barcelona                33
Burnley                     33
AS Monaco                   33
Everton                     33
TSG 1899 Hoffenheim         33
Wolverhampton Wanderers     33
Eintracht Frankfurt         33
Southampton                 33
Valencia CF                 33
Name: Club, dtype: int64
```

Distribution of Overall Score in Different Popular Clubs

In [29]:

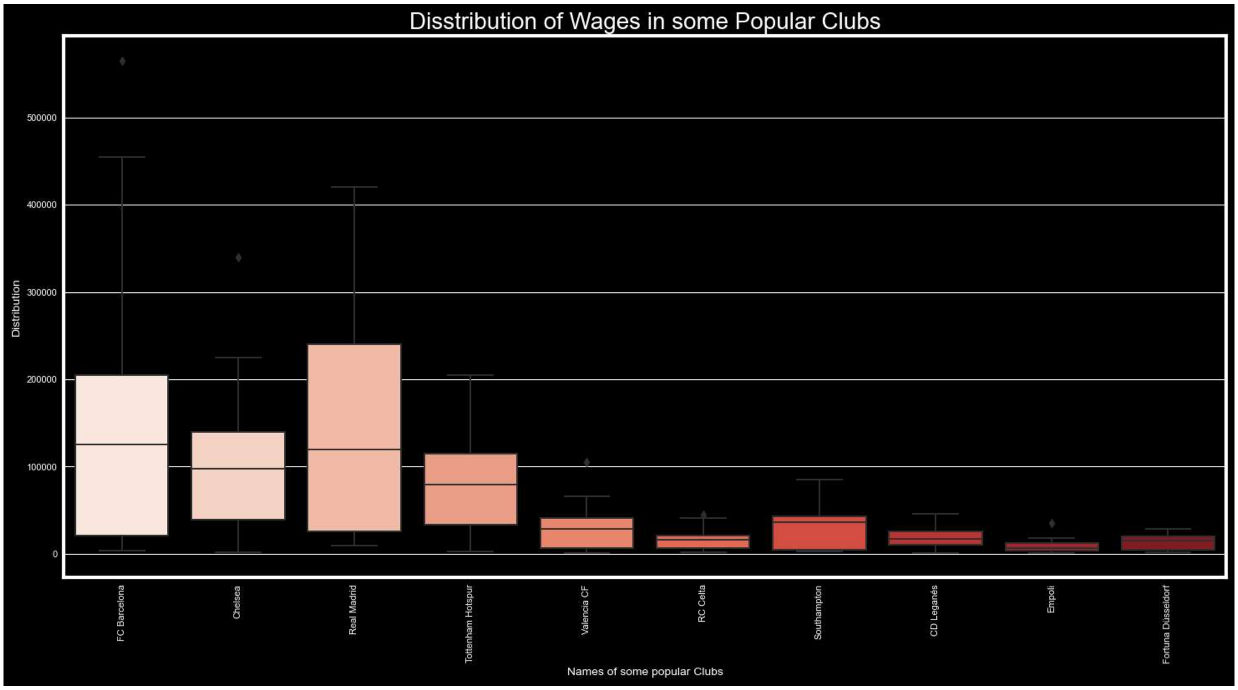
```
## Graph showing the Distribution of overall score of Different Popular Clubs
some_clubs = ('CD Leganés', 'Southampton', 'RC Celta', 'Empoli', 'Fortuna Düsseld
             'Tottenham Hotspur', 'FC Barcelona', 'Valencia CF', 'Chelsea', 'Real

data_clubs = data.loc[data['Club'].isin(some_clubs) & data['Overall']]

plt.rcParams['figure.figsize'] = (15, 8)
ax = sns.boxplot(x = data_clubs['Club'], y = data_clubs['Overall'], palette = 'in
ax.set_xlabel(xlabel = 'Some Popular Clubs', fontsize = 9)
ax.set_ylabel(ylabel = 'Overall Score', fontsize = 9)
ax.set_title(label = 'Distribution of Overall Score in Different popular Clubs', 
plt.xticks(rotation = 90)
plt.show()
```



Distribution of Wages in some Popular clubs

In [30]:

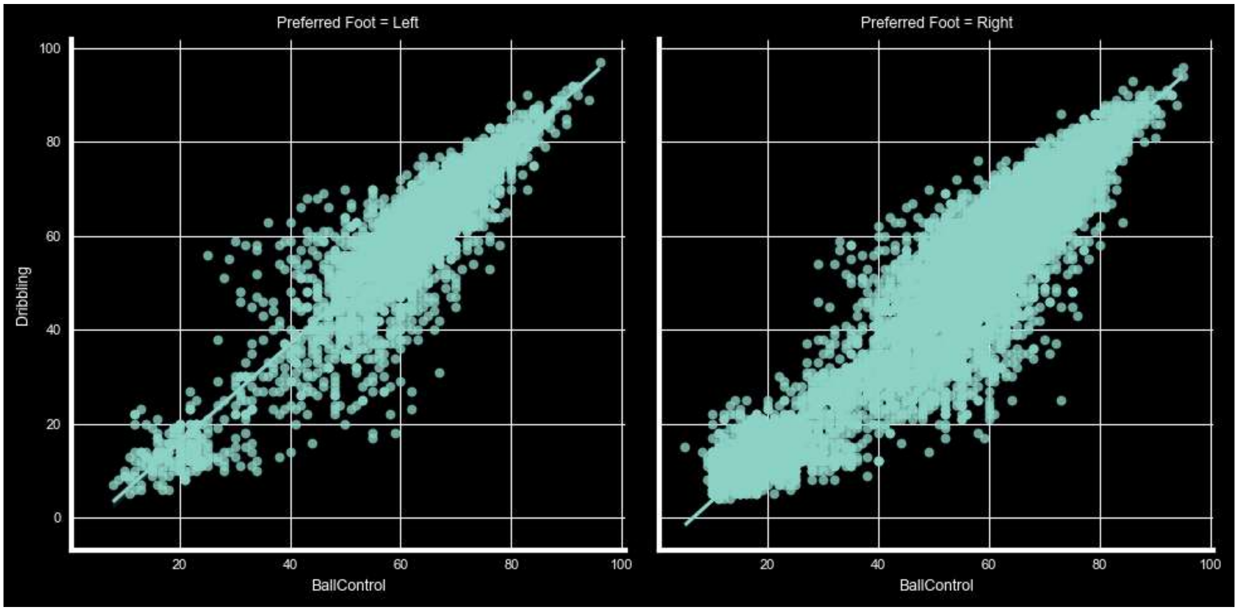
```
## Plotting the Distribution of Wages according to different Clubs
some_clubs = ('CD Leganés', 'Southampton', 'RC Celta', 'Empoli', 'Fortuna Düsseld
             'Tottenham Hotspur', 'FC Barcelona', 'Valencia CF', 'Chelsea', 'Real

data_club = data.loc[data['Club'].isin(some_clubs) & data['Wage']]

plt.rcParams['figure.figsize'] = (16, 8)
ax = sns.boxplot(x = 'Club', y = 'Wage', data = data_club, palette = 'Reds')
ax.set_xlabel(xlabel = 'Names of some popular Clubs', fontsize = 10)
ax.set_ylabel(ylabel = 'Distribution', fontsize = 10)
ax.set_title(label = 'Disstribution of Wages in some Popular Clubs', fontsize = 2
plt.xticks(rotation = 90)
plt.show()
```



Comparing the performance of left-footed and right-footed footballers

In [31]:

```
## Ballcontrol vs Dribbing

sns.lmplot(x = 'BallControl', y = 'Dribbling', data = data, col = 'Preferred Foot
plt.show()
```



- - - - - - - - X X X X X X X X - - - - - - - -

In [ ]: