FIFA 2020 Players Analytics

In [2]:

```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

In [3]:

```
df = pd.read_csv('./players_20.csv')
df.head()
```

Out[3]:

	sofifa_id	player_url	short_name	long_name	age	dob	height_cm	weight_kg	nationality
0	158023	https://sofifa.com/player/158023/lionel-messi/...	L. Messi	Lionel Andrés Messi Cuccittini	32	1987-06-24	170	72	Argentina
1	20801	https://sofifa.com/player/20801/c-ronaldo-dos-...	Cristiano Ronaldo	Cristiano Ronaldo dos Santos Aveiro	34	1985-02-05	187	83	Portugal
2	190871	https://sofifa.com/player/190871/neymar-da-sil...	Neymar Jr	Neymar da Silva Santos Junior	27	1992-02-05	175	68	Brazil
3	200389	https://sofifa.com/player/200389/jan-oblak/20/...	J. Oblak	Jan Oblak	26	1993-01-07	188	87	Slovenia
4	183277	https://sofifa.com/player/183277/eden-hazard/2...	E. Hazard	Eden Hazard	28	1991-01-07	175	74	Belgium

5 rows × 104 columns

In [4]:

```
cols = []
for col in df.columns:
    cols.append(col)
print(cols)
```

```
['sofifa_id', 'player_url', 'short_name', 'long_name', 'age', 'dob', 'height_cm', 'weight
_kg', 'nationality', 'club', 'overall', 'potential', 'value_eur', 'wage_eur', 'player_pos
itions', 'preferred_foot', 'international_reputation', 'weak_foot', 'skill_moves', 'work_
rate', 'body_type', 'real_face', 'release_clause_eur', 'player_tags', 'team_position', 't
eam_jersey_number', 'loaned_from', 'joined', 'contract_valid_until', 'nation_position', '
nation_jersey_number', 'pace', 'shooting', 'passing', 'dribbling', 'defending', 'physic',
'gk_diving', 'gk_handling', 'gk_kicking', 'gk_reflexes', 'gk_speed', 'gk_positioning', 'p
layer_traits', 'attacking_crossing', 'attacking_finishing', 'attacking_heading_accuracy',
'attacking_short_passing', 'attacking_volleys', 'skill_dribbling', 'skill_curve', 'skill_
fk_accuracy', 'skill_long_passing', 'skill_ball_control', 'movement_acceleration', 'movem
ent_sprint_speed', 'movement_agility', 'movement_reactions', 'movement_balance', 'power_s
hot_power', 'power_jumping', 'power_stamina', 'power_strength', 'power_long_shots', 'ment
ality_aggression', 'mentality_interceptions', 'mentality_positioning', 'mentality_vision'
, 'mentality_penalties', 'mentality_composure', 'defending_marking', 'defending_standing_
tackle', 'defending_sliding_tackle', 'goalkeeping_diving', 'goalkeeping_handling', 'goalk
eeping_kicking', 'goalkeeping_positioning', 'goalkeeping_reflexes', 'ls', 'st', 'rs', 'lw
', 'lf', 'cf', 'rf', 'rw', 'lam', 'cam', 'ram', 'lm', 'lcm', 'cm', 'rcm', 'rm', 'lwb', 'l
dm', 'cdm', 'rdm', 'rwb', 'lb', 'lcb', 'cb', 'rcb', 'rb']
```

Columns of No use:

'sofifa_id', 'player_url', 'long_name', 'real_face','goalkeeping_diving', 'goalkeeping_handling', 'goalkeeping_kicking', 'goalkeeping_positioning', 'goalkeeping_reflexes', 'ls', 'st', 'rs', 'lw', 'lf', 'cf', 'rf', 'rw', 'lam', 'cam', 'ram', 'lm', 'lcm', 'cm', 'rcm', 'rm', 'lwb', 'ldm', 'cdm', 'rdm', 'rwb', 'lb', 'lcb', 'cb', 'rcb', 'rb'

Removing Unnecessary Columns

In [5]:

```
toDrop = ['sofifa_id', 'player_url', 'long_name', 'real_face','goalkeeping_diving', 'goa
lkeeping_handling', 'goalkeeping_kicking', 'goalkeeping_positioning', 'goalkeeping_reflex
es', 'ls', 'st', 'rs', 'lw', 'lf', 'cf', 'rf', 'rw', 'lam', 'cam', 'ram', 'lm', 'lcm', '
cm', 'rcm', 'rm', 'lwb', 'ldm', 'cdm', 'rdm', 'rwb', 'lb', 'lcb', 'cb', 'rcb', 'rb']

df.drop(toDrop, axis='columns', inplace=True)
df.columns
```

Out[5]:

```
Index(['short_name', 'age', 'dob', 'height_cm', 'weight_kg', 'nationality',
       'club', 'overall', 'potential', 'value_eur', 'wage_eur',
       'player_positions', 'preferred_foot', 'international_reputation',
       'weak_foot', 'skill_moves', 'work_rate', 'body_type',
       'release_clause_eur', 'player_tags', 'team_position',
       'team_jersey_number', 'loaned_from', 'joined', 'contract_valid_until',
       'nation_position', 'nation_jersey_number', 'pace', 'shooting',
       'passing', 'dribbling', 'defending', 'physic', 'gk_diving',
       'gk_handling', 'gk_kicking', 'gk_reflexes', 'gk_speed',
       'gk_positioning', 'player_traits', 'attacking_crossing',
       'attacking_finishing', 'attacking_heading_accuracy',
       'attacking_short_passing', 'attacking_volleys', 'skill_dribbling',
       'skill_curve', 'skill_fk_accuracy', 'skill_long_passing',
       'skill_ball_control', 'movement_acceleration', 'movement_sprint_speed',
       'movement_agility', 'movement_reactions', 'movement_balance',
       'power_shot_power', 'power_jumping', 'power_stamina', 'power_strength',
       'power_long_shots', 'mentality_aggression', 'mentality_interceptions',
       'mentality_positioning', 'mentality_vision', 'mentality_penalties',
       'mentality_composure', 'defending_marking', 'defending_standing_tackle',
       'defending_sliding_tackle'],
      dtype='object')
```

Checking for null values in the data

In [6]:

```
df.isnull().sum()
```

Out[6]:

```
short_name                   0
age                          0
dob                          0
height_cm                    0
weight_kg                    0
                            ..
mentality_penalties          0
mentality_composure          0
defending_marking            0
defending_standing_tackle    0
defending_sliding_tackle     0
Length: 69, dtype: int64
```

Describing Data

In [7]:

```
df.describe()
```

Out[7]:

	age	height_cm	weight_kg	overall	potential	value_eur	wage_eur	international_repu
	age	height_cm	weight_kg	overall	potential	value_eur	wage_eur	international_repu
count	18278.000000	18278.000000	18278.000000	18278.000000	18278.000000	1.827800e+04	18278.000000	18278.(
mean	25.283291	181.362184	75.276343	66.244994	71.546887	2.484038e+06	9456.942773	1.1
std	4.656964	6.756961	7.047744	6.949953	6.139669	5.585481e+06	21351.714095	0.3
min	16.000000	156.000000	50.000000	48.000000	49.000000	0.000000e+00	0.000000	1.(
25%	22.000000	177.000000	70.000000	62.000000	67.000000	3.250000e+05	1000.000000	1.(
50%	25.000000	181.000000	75.000000	66.000000	71.000000	7.000000e+05	3000.000000	1.(
75%	29.000000	186.000000	80.000000	71.000000	75.000000	2.100000e+06	8000.000000	1.(
max	42.000000	205.000000	110.000000	94.000000	95.000000	1.055000e+08	565000.000000	5.(

8 rows × 55 columns

In [8]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18278 entries, 0 to 18277
Data columns (total 69 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   short_name                  18278 non-null  object 
 1   age                         18278 non-null  int64  
 2   dob                         18278 non-null  object 
 3   height_cm                   18278 non-null  int64  
 4   weight_kg                   18278 non-null  int64  
 5   nationality                 18278 non-null  object 
 6   club                        18278 non-null  object 
 7   overall                     18278 non-null  int64  
 8   potential                   18278 non-null  int64  
 9   value_eur                   18278 non-null  int64  
 10  wage_eur                    18278 non-null  int64  
 11  player_positions            18278 non-null  object 
 12  preferred_foot              18278 non-null  object 
 13  international_reputation    18278 non-null  int64  
 14  weak_foot                   18278 non-null  int64  
 15  skill_moves                 18278 non-null  int64  
 16  work_rate                   18278 non-null  object 
 17  body_type                   18278 non-null  object 
 18  release_clause_eur          16980 non-null  float64
 19  player_tags                 1499 non-null   object 
 20  team_position               18038 non-null  object 
 21  team_jersey_number          18038 non-null  float64
 22  loaned_from                 1048 non-null   object 
 23  joined                      16990 non-null  object 
 24  contract_valid_until        18038 non-null  float64
 25  nation_position             1126 non-null   object 
 26  nation_jersey_number        1126 non-null   float64
 27  pace                        16242 non-null  float64
 28  shooting                    16242 non-null  float64
 29  passing                     16242 non-null  float64
 30  dribbling                   16242 non-null  float64
 31  defending                   16242 non-null  float64
 32  physic                      16242 non-null  float64
 33  gk_diving                   2036 non-null   float64
 34  gk_handling                 2036 non-null   float64
 35  gk_kicking                  2036 non-null   float64
 36  gk_reflexes                 2036 non-null   float64
 37  gk_speed                    2036 non-null   float64
 38  gk_positioning              2036 non-null   float64
 39  player_traits               7566 non-null   object 
 40  attacking_crossing          18278 non-null  int64  
 41  attacking_finishing         18278 non-null  int64  
 42  attacking_heading_accuracy  18278 non-null  int64  
 43  attacking_short_passing     18278 non-null  int64  
 44  attacking_volleys           18278 non-null  int64  
 45  skill_dribbling             18278 non-null  int64
 46  skill_curve                 18278 non-null  int64
 47  skill_fk_accuracy           18278 non-null  int64
 48  skill_long_passing          18278 non-null  int64
 49  skill_ball_control          18278 non-null  int64
 50  movement_acceleration       18278 non-null  int64
 51  movement_sprint_speed       18278 non-null  int64
 52  movement_agility            18278 non-null  int64
 53  movement_reactions          18278 non-null  int64
 54  movement_balance            18278 non-null  int64
 55  power_shot_power            18278 non-null  int64
 56  power_jumping               18278 non-null  int64
 57  power_stamina               18278 non-null  int64
 58  power_strength              18278 non-null  int64
 59  power_long_shots            18278 non-null  int64
 60  mentality_aggression        18278 non-null  int64
 61  mentality_interceptions     18278 non-null  int64
 62  mentality_positioning       18278 non-null  int64
 63  mentality_vision            18278 non-null  int64
 64  mentality_penalties         18278 non-null  int64
 65  mentality_composure         18278 non-null  int64
 66  defending_marking           18278 non-null  int64
 67  defending_standing_tackle   18278 non-null  int64
 68  defending_sliding_tackle    18278 non-null  int64
dtypes: float64(16), int64(39), object(14)
memory usage: 9.6+ MB
```

In [9]:

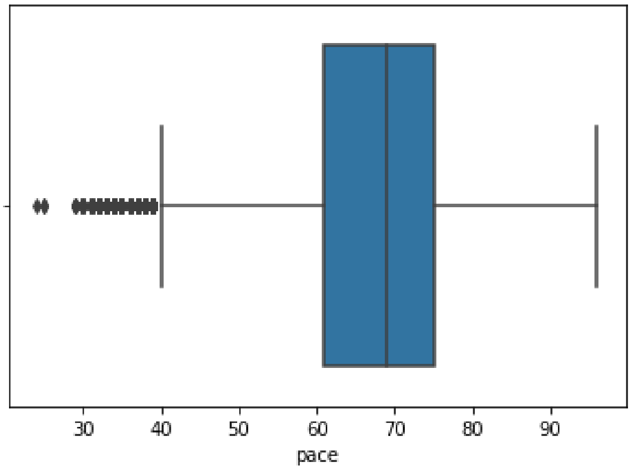
```
#boxplot of pace without cleaning
import seaborn as sns

sns.boxplot(df['pace'])
```

```
C:\Users\1992729\AppData\Roaming\Python\Python39\site-packages\seaborn\_decorators.py:36:
FutureWarning: 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 expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[9]:

```
<AxesSubplot:xlabel='pace'>
```



Filling Null Values

In [10]:

```
#there are multiple null values in pace column so filling those null values with Mean
df['pace'] = df['pace'].fillna(df['pace'].mean())
```

In [11]:

```
df['pace'].info()
```

```
<class 'pandas.core.series.Series'>
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 18278 entries, 0 to 18277
Series name: pace
Non-Null Count  Dtype
--------------  -----
18278 non-null  float64
dtypes: float64(1)
memory usage: 142.9 KB
```

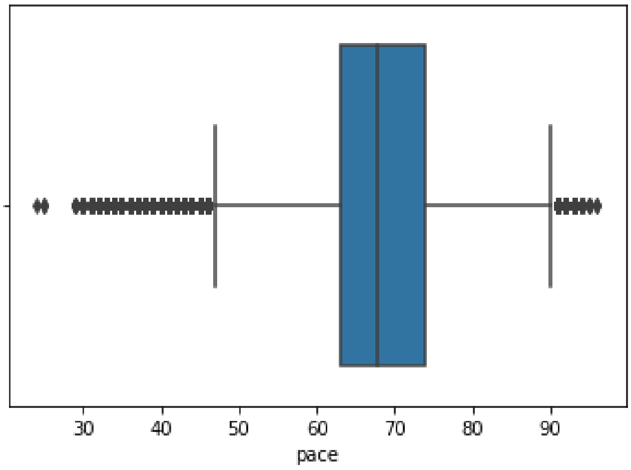
In [12]:

```
#boxplot of pace after cleaning
sns.boxplot(df['pace'])
```

```
C:\Users\1992729\AppData\Roaming\Python\Python39\site-packages\seaborn\_decorators.py:36:
FutureWarning: 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 expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[12]:

```
<AxesSubplot:xlabel='pace'>
```



In [13]:

```
#filling null values in shooting with average shooting ratings
df['shooting'] = df['shooting'].fillna(df['shooting'].mean())
df['shooting'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 18278 entries, 0 to 18277
Series name: shooting
Non-Null Count  Dtype
--------------  -----
18278 non-null  float64
dtypes: float64(1)
memory usage: 142.9 KB
```

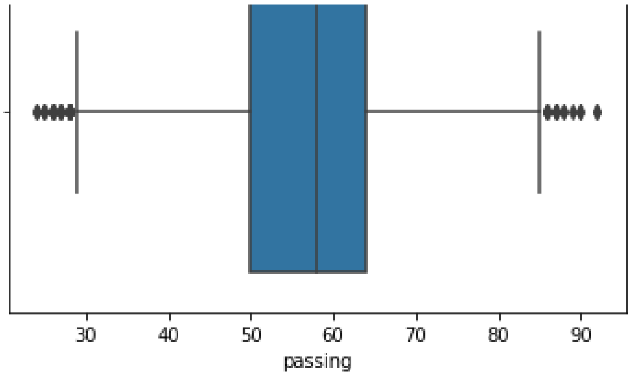
In [14]:

```
#box plot of passing before cleaning
sns.boxplot(df['passing'])
```

```
C:\Users\1992729\AppData\Roaming\Python\Python39\site-packages\seaborn\_decorators.py:36:
FutureWarning: 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 expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[14]:

```
<AxesSubplot:xlabel='passing'>
```



In [15]:

```
#replacing null values with median in passing column
df['passing'] = df['passing'].fillna(df['passing'].median())
df['passing'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 18278 entries, 0 to 18277
Series name: passing
Non-Null Count  Dtype
--------------  -----
18278 non-null  float64
dtypes: float64(1)
memory usage: 142.9 KB
```

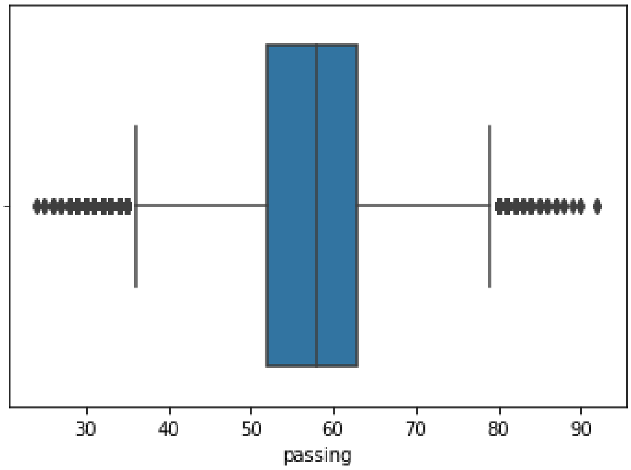
In [16]:

```
#boxplot after cleaning data
sns.boxplot(df['passing'])
```

```
C:\Users\1992729\AppData\Roaming\Python\Python39\site-packages\seaborn\_decorators.py:36:
FutureWarning: 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 expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[16]:

```
<AxesSubplot:xlabel='passing'>
```



In [17]:

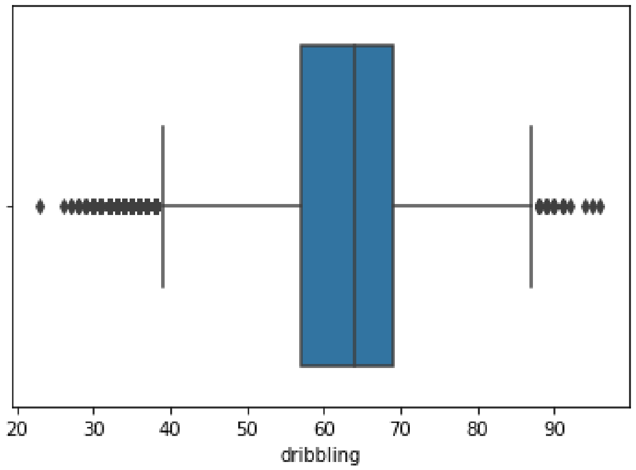
```
#boxplot of dribbling before cleaning
sns.boxplot(df['dribbling'])
```

```
C:\Users\1992729\AppData\Roaming\Python\Python39\site-packages\seaborn\_decorators.py:36:
FutureWarning: 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 expli
cit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[17]:

```
<AxesSubplot:xlabel='dribbling'>
```

```
<AxesSubplot:xlabel='dribbling'>
```



In [18]:

```
#box plot of dribbling after cleaning
df['dribbling'] = df['dribbling'].fillna(df['dribbling'].mean())
df['dribbling'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 18278 entries, 0 to 18277
Series name: dribbling
Non-Null Count  Dtype  
--------------  -----  
18278 non-null  float64
dtypes: float64(1)
memory usage: 142.9 KB
```

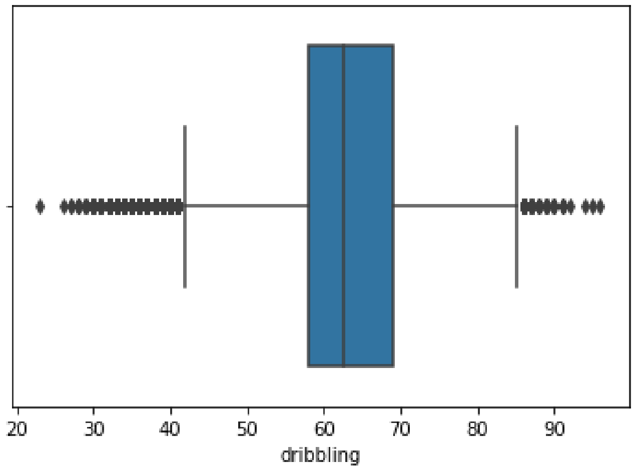
In [19]:

```
#boxplot after cleaning
sns.boxplot(df['dribbling'])
```

```
C:\Users\1992729\AppData\Roaming\Python\Python39\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[19]:

```
<AxesSubplot:xlabel='dribbling'>
```



In [20]:

```
#filling na values in physic
df['physic'] = df['physic'].fillna(df['physic'].mean())
df['physic'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 18278 entries, 0 to 18277
Series name: physic
Series name: physic
Non-Null Count  Dtype
--------------  -----
18278 non-null  float64
dtypes: float64(1)
memory usage: 142.9 KB
```

In [21]:

```
#filling na values in defending
df['defending'] = df['defending'].fillna(df['defending'].mean())
```

Top 5 Players Based On Overall Ratings

In [22]:

```
df.nlargest(5, 'overall')
```

Out[22]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots
0	L. Messi	32	1987-06-24	170	72	Argentina	FC Barcelona	94	94	95500000	...	94
1	Cristiano Ronaldo	34	1985-02-05	187	83	Portugal	Juventus	93	93	58500000	...	93
2	Neymar Jr	27	1992-02-05	175	68	Brazil	Paris Saint-Germain	92	92	105500000	...	84
3	J. Oblak	26	1993-01-07	188	87	Slovenia	Atlético Madrid	91	93	77500000	...	12
4	E. Hazard	28	1991-01-07	175	74	Belgium	Real Madrid	91	91	90000000	...	80

5 rows × 69 columns

Top 5 Players Based On Potential

In [23]:

```
df.nlargest(5, 'potential')
```

Out[23]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots
10	K. Mbappé	20	1998-12-20	178	73	France	Paris Saint-Germain	89	95	93500000	...	79
0	L. Messi	32	1987-06-24	170	72	Argentina	FC Barcelona	94	94	95500000	...	94
1	Cristiano Ronaldo	34	1985-02-05	187	83	Portugal	Juventus	93	93	58500000	...	93
3	J. Oblak	26	1993-01-07	188	87	Slovenia	Atlético Madrid	91	93	77500000	...	12
6	M. ter Stegen	27	1992-04-30	187	85	Germany	FC Barcelona	90	93	67500000	...	10

5 rows × 69 columns

In [24]:

```
df['dob'].dtype
```

Out[24]:

```
dtype('O')
```

Data type of Date of Birth is object, so we need to change it to date

In [25]:

```
df['dob'] = pd.to_datetime(df['dob'])
df['dob'].dtype
```

Out[25]:

```
dtype('<M8[ns]')
```

Top 5 Players Based On Dribbling Skills

In [26]:

```
df.nlargest(5, 'dribbling')
```

Out[26]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_sho
0	L. Messi	32	1987-06-24	170	72	Argentina	FC Barcelona	94	94	95500000	...	9
2	Neymar Jr	27	1992-02-05	175	68	Brazil	Paris Saint-Germain	92	92	105500000	...	8
4	E. Hazard	28	1991-01-07	175	74	Belgium	Real Madrid	91	91	90000000	...	8
41	Bernardo Silva	24	1994-08-10	173	64	Portugal	Manchester City	87	90	64000000	...	7
48	D. Mertens	32	1987-05-06	169	61	Belgium	Napoli	87	87	40000000	...	8

5 rows × 69 columns

Top 5 Defenders

In [27]:

```
df.nlargest(5, 'defending')
```

Out[27]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shot
7	V. van Dijk	27	1991-07-08	193	92	Netherlands	Liverpool	90	91	78000000	...	6
16	G. Chiellini	34	1984-08-14	187	85	Italy	Juventus	89	89	24500000	...	4
11	K. Koulibaly	28	1991-06-20	187	89	Senegal	Napoli	89	91	67500000	...	1
35	D. Godín	33	1986-02-16	187	78	Uruguay	Inter	88	88	28000000	...	4
49	M. Hummels	30	1988-12-16	191	94	Germany	Borussia Dortmund	87	87	41000000	...	5

5 rows × 69 columns

5 Highest Paid Players

In [28]:

```
df.nlargest(5, 'value_eur')
```

Out[28]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_sho
2	Neymar Jr	27	1992-02-05	175	68	Brazil	Paris Saint-Germain	92	92	105500000	...	8
0	L. Messi	32	1987-06-24	170	72	Argentina	FC Barcelona	94	94	95500000	...	9
10	K. Mbappé	20	1998-12-20	178	73	France	Paris Saint-Germain	89	95	93500000	...	7
4	E. Hazard	28	1991-01-07	175	74	Belgium	Real Madrid	91	91	90000000	...	8
5	K. De Bruyne	28	1991-06-28	181	70	Belgium	Manchester City	91	91	90000000	...	9

5 rows × 69 columns

What is the average age of players in dataset?

In [29]:

```
print("Average age of players: ", round(df['age'].mean(),2))
```

```
Average age of players:  25.28
```

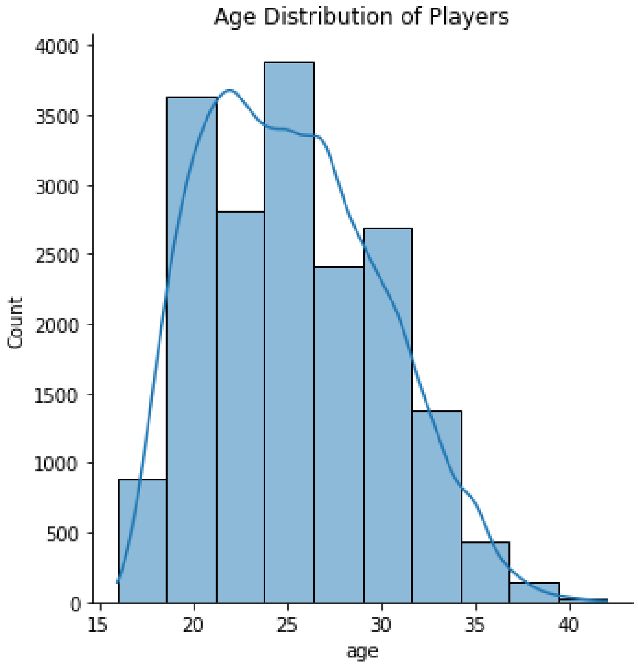
Distplot For Age

In [31]:

```
sns.displot(df['age'], bins=10, kde=True)
plt.title("Age Distribution of Players")
```

Out[31]:

```
Text(0.5, 1.0, 'Age Distribution of Players')
```



Players with maximum age

In [32]:

```
df[df['age'] == df['age'].max()]
```

Out[32]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_s
11832	C. Muñoz	42	1977-07-01	177	73	Argentina	CD Universidad de Concepción	64	64	50000	...	
13003	H. Sulaimani	42	1977-01-21	173	70	Saudi Arabia	Al Ahli	63	63	0	...	

2 rows × 69 columns

What is the minimum age of players?

In [33]:

```
print("Minimum age of players: ", df['age'].min())
```

```
Minimum age of players:  16
```

Players with minimum age

In [34]:

```
df[df['age'] == df['age'].min()][:5]
```

Out[34]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long
4764	A. Hložek	16	2002-07-25	185	80	Czech Republic	Sparta Praha	70	86	3500000	...	
6630	Fábio Silva	16	2002-07-19	185	75	Portugal	FC Porto	68	85	1800000	...	
12158	E. Millot	16	2002-07-17	175	65	France	AS Monaco	63	86	800000	...	
12160	S. Esposito	16	2002-07-02	186	75	Italy	Inter	63	85	825000	...	
14626	A. Velasco	16	2002-07-27	167	63	Argentina	Independiente	60	83	450000	...	

5 rows × 69 columns

What is the average height of players?

In [35]:

```
print("Average Height Of Players: ", round(df['height_cm'].mean(),2))
```

```
Average Height Of Players:  181.36
```

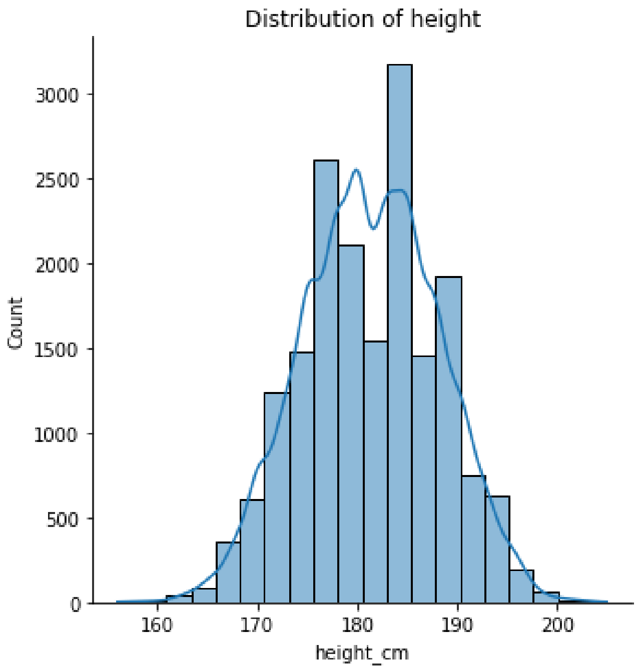
Distplot for height

In [36]:

```
sns.displot(df['height_cm'], kde=True, bins=20)
plt.title("Distribution of height")
```

Out[36]:

```
Text(0.5, 1.0, 'Distribution of height')
```



What is the average weight of players?

In [37]:

```
print("Average Weight Of Players: ", round(df['weight_kg'].mean(),2))
```

```
Average Weight Of Players:  75.28
```

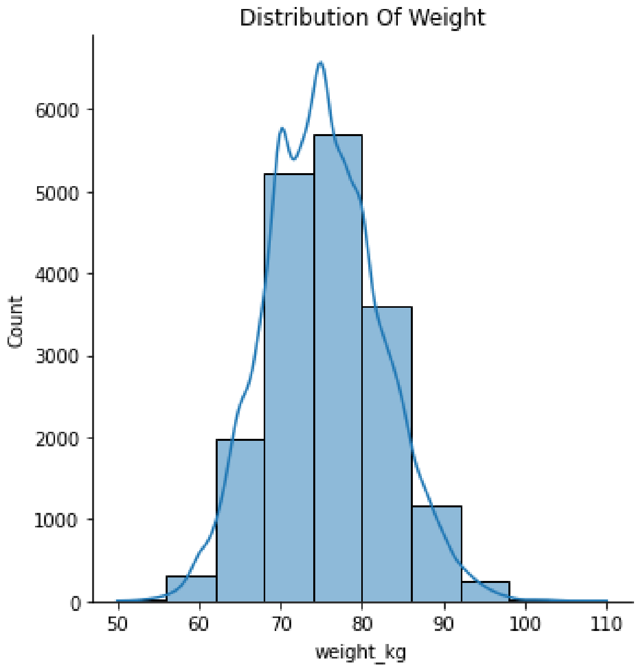
Distplot for weight

In [38]:

```
sns.displot(df['weight_kg'], kde=True, bins=10)
plt.title("Distribution Of Weight")
```

Out[38]:

```
Text(0.5, 1.0, 'Distribution Of Weight')
```



Top 5 Nationalities

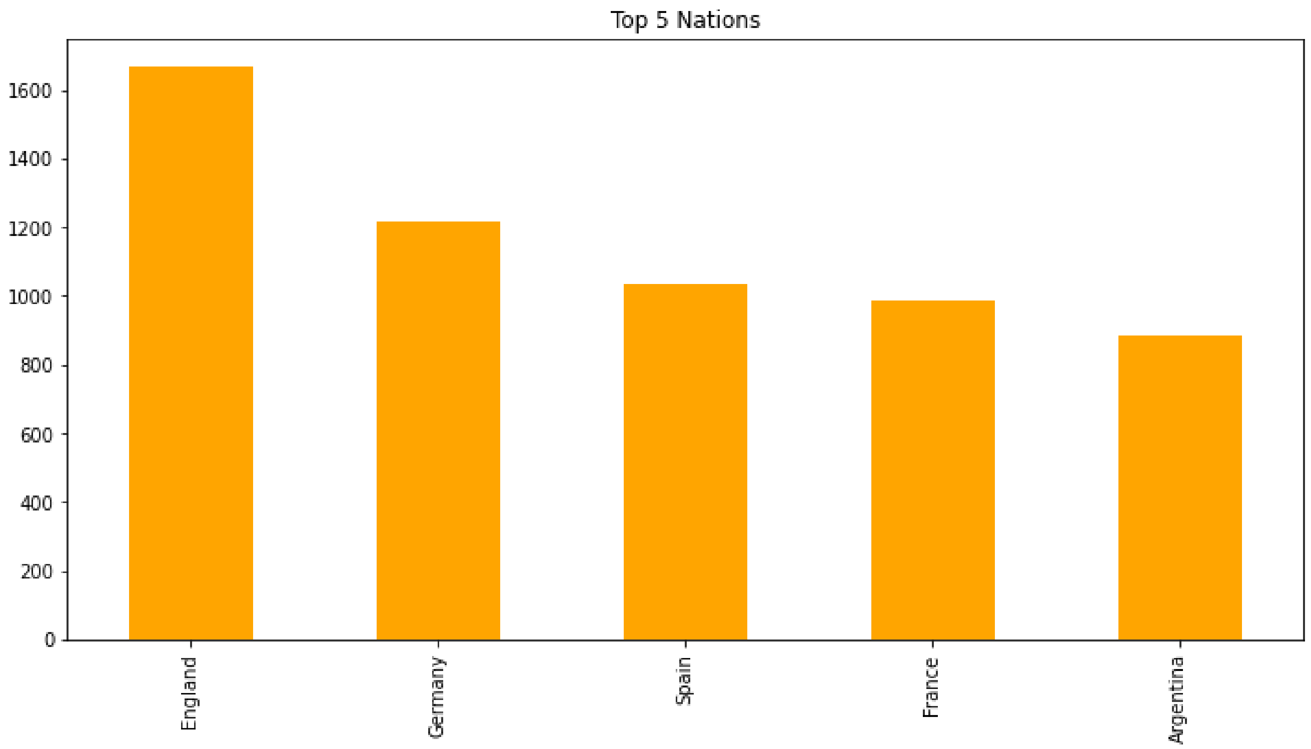
In [39]:

```
df['nationality'].value_counts()[:5].plot(kind='bar', figsize=(12,6), color = 'orange')
```

```
df['nationality'].value_counts()[:5].plot(kind='bar', figsize=(12,6), color = 'orange')
plt.title("Top 5 Nations")
```

Out[39]:

```
Text(0.5, 1.0, 'Top 5 Nations')
```



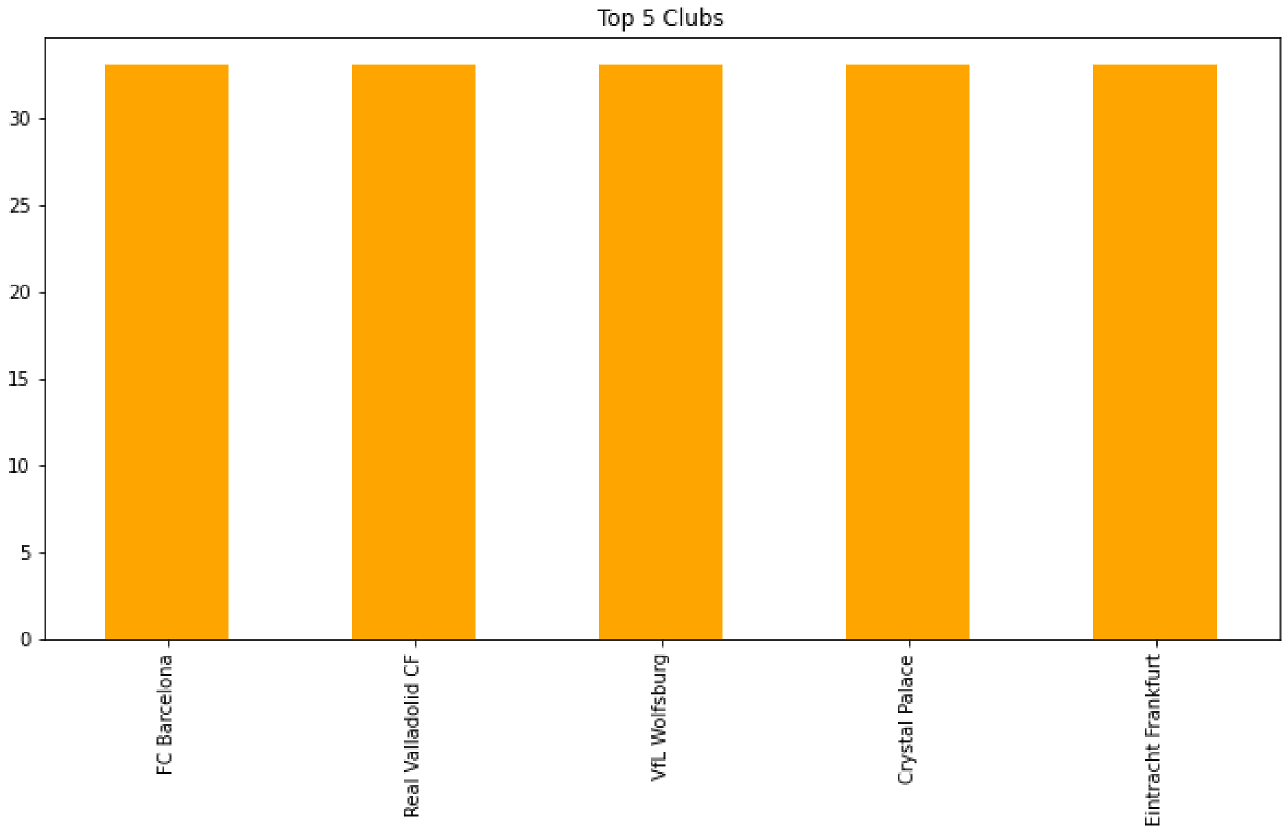
Top 5 Clubs

In [40]:

```
df['club'].value_counts()[:5].plot(kind='bar', figsize=(12,6), color = 'orange')
plt.title("Top 5 Clubs")
```

Out[40]:

```
Text(0.5, 1.0, 'Top 5 Clubs')
```



In [41]:

```
df.columns
```

Out[41]:

```
Index(['short_name', 'age', 'dob', 'height_cm', 'weight_kg', 'nationality',
       'club', 'overall', 'potential', 'value_eur', 'wage_eur',
       'player_positions', 'preferred_foot', 'international_reputation',
       'weak_foot', 'skill_moves', 'work_rate', 'body_type',
       'release_clause_eur', 'player_tags', 'team_position',
       'team_jersey_number', 'loaned_from', 'joined', 'contract_valid_until',
       'nation_position', 'nation_jersey_number', 'pace', 'shooting',
       'passing', 'dribbling', 'defending', 'physic', 'gk_diving',
       'gk_handling', 'gk_kicking', 'gk_reflexes', 'gk_speed',
       'gk_positioning', 'player_traits', 'attacking_crossing',
       'attacking_finishing', 'attacking_heading_accuracy',
       'attacking_short_passing', 'attacking_volleys', 'skill_dribbling',
       'skill_curve', 'skill_fk_accuracy', 'skill_long_passing',
       'skill_ball_control', 'movement_acceleration', 'movement_sprint_speed',
       'movement_agility', 'movement_reactions', 'movement_balance',
       'power_shot_power', 'power_jumping', 'power_stamina', 'power_strength',
       'power_long_shots', 'mentality_aggression', 'mentality_interceptions',
       'mentality_positioning', 'mentality_vision', 'mentality_penalties',
       'mentality_composure', 'defending_marking', 'defending_standing_tackle',
       'defending_sliding_tackle'],
      dtype='object')
```

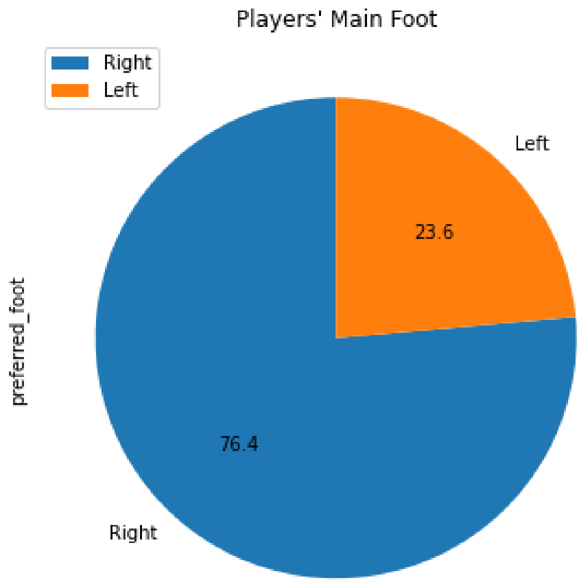
Which foot is preferred more?

In [42]:

```
df['preferred_foot'].value_counts().plot(kind='pie', figsize=(10,6), autopct="%.1f", startangle=90)
plt.title("Players' Main Foot")
plt.legend()
```

Out[42]:

```
<matplotlib.legend.Legend at 0x2921d242250>
```



Who is the best right foot player in the dataset?

In [43]:

```
df[df['preferred_foot'] == 'Right'].sort_values(by='overall', ascending=False)[:1]
```

Out[43]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots	n

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots	m
1	Cristiano Ronaldo	34	1985-02-05	187	83	Portugal	Juventus	93	93	58500000	...	93	

1 rows × 69 columns

Who is the best right foot defender in the dataset?

In [112]:

```
df[(df['preferred_foot'] == 'Right') & (df['player_positions'] == 'CB')].sort_values(by=
'overall', ascending=False)[:1]
```

Out[112]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots
7	V. van Dijk	27	1991-07-08	193	92	Netherlands	Liverpool	90	91	78000000	...	64

1 rows × 69 columns

Who is the best right foot goal keeper in the dataset?

In [44]:

```
df[(df['preferred_foot'] == 'Right') & (df['player_positions'] == 'GK')].sort_values(by=
'overall', ascending=False)[:1]
```

Out[44]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots	me
3	J. Oblak	26	1993-01-07	188	87	Slovenia	Atlético Madrid	91	93	77500000	...	12	

1 rows × 69 columns

Who is the best left foot player in the dataset?

In [108]:

```
df[df['preferred_foot'] == 'Left'].sort_values(by='overall', ascending=False)[:1]
```

Out[108]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots
0	L. Messi	32	1987-06-24	170	72	Argentina	FC Barcelona	94	94	95500000	...	94

1 rows × 69 columns

Who is the best left foot defender in the dataset?

In [46]:

```
df[(df['preferred_foot'] == 'Left') & (df['player_positions'] == 'CB')].sort_values(by='
overall', ascending=False)[:1]
```

Out[46]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots
16	G. Chiellini	34	1984-08-14	187	85	Italy	Juventus	89	89	24500000	...	49

1 rows × 69 columns

Who is the best left foot GK in the dataset?

In [47]:

```
df[(df['preferred_foot'] == 'Left') & (df['player_positions'] == 'GK')].sort_values(by='
overall', ascending=False)[:1]
```

Out[47]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shot
25	Ederson	25	1993-08-17	188	86	Brazil	Manchester City	88	91	54500000	...	1

1 rows × 69 columns

Are there any indian players in the dataset?

In [109]:

```
df[df['nationality'] == 'India']
```

Out[109]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots
12155	G. Chatterjee	33	1985-07-04	195	84	India	India	64	64	0	...	13
13102	P. Bhatt	34	1985-04-07	182	78	India	India	63	63	0	...	58
13103	B. Raj	31	1988-03-25	175	69	India	India	63	63	0	...	57
13970	A. Chakraborty	33	1985-10-27	189	79	India	India	62	62	0	...	42
13971	H. Bhandari	30	1989-06-22	178	74	India	India	62	62	0	...	49
13972	A. Swaminathan	27	1991-10-29	173	65	India	India	62	62	0	...	51
14625	D. Pillai	31	1988-06-22	178	70	India	India	61	61	0	...	53
15204	A. Ginti	25	1993-08-10	170	69	India	India	60	62	0	...	38
15205	A. Khurana	26	1993-04-21	186	81	India	India	60	62	0	...	37
15321	A. Deshpande	38	1980-12-17	180	76	India	India	60	60	0	...	57
15322	T. Atwal	35	1983-09-28	181	77	India	India	60	60	0	...	51
15323	B. Sidhu	31	1987-08-20	171	68	India	India	60	60	0	...	55
15324	R. Nadkarni	33	1985-10-31	180	70	India	India	60	60	0	...	51
15809	V. Boral	29	1989-07-14	183	77	India	India	59	60	0	...	12
15866	D. Sundaram	30	1989-	186	81	India	India	59	59	0	...	30

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shots
15867	R. Jayaraman	34	1984-11-20	180	82	India	India	59	59	0	...	57
15868	D. Bajwa	38	1981-03-05	182	72	India	India	59	59	0	...	33
16300	O. Patla	26	1992-10-13	167	69	India	India	58	61	0	...	49
16303	P. Nagarajan	29	1989-10-12	189	87	India	India	58	60	0	...	13
16352	T. Agarwal	33	1986-02-15	173	68	India	India	58	58	0	...	52
16353	A. Varkay	33	1985-08-18	179	71	India	India	58	58	0	...	17
16354	C. Palan	28	1991-06-20	181	69	India	India	58	58	0	...	51
16714	D. Singhal	34	1984-11-09	177	71	India	India	57	57	0	...	58

23 rows × 69 columns

In [48]:

```
groupedDf = df.groupby(by = ['age', 'overall'])['age'].count().unstack()
groupedDf.fillna(df['overall'].mean())
```

Out[48]:

overall	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**...**	
age												
16	2.000000	1.000000	66.244994	1.000000	2.000000	1.000000	66.244994	66.244994	66.244994	66.244994	...	66.
17	8.000000	13.000000	13.000000	16.000000	26.000000	17.000000	17.000000	14.000000	10.000000	14.000000	...	66.
18	14.000000	18.000000	28.000000	41.000000	42.000000	46.000000	45.000000	41.000000	40.000000	18.000000	...	66.
19	9.000000	10.000000	18.000000	32.000000	33.000000	42.000000	65.000000	53.000000	65.000000	55.000000	...	1.
20	3.000000	7.000000	16.000000	12.000000	28.000000	33.000000	34.000000	55.000000	79.000000	77.000000	...	1.
21	4.000000	8.000000	7.000000	10.000000	12.000000	23.000000	27.000000	41.000000	42.000000	56.000000	...	66.
22	2.000000	3.000000	11.000000	1.000000	14.000000	16.000000	18.000000	28.000000	43.000000	49.000000	...	1.
23	66.244994	3.000000	3.000000	1.000000	3.000000	5.000000	10.000000	15.000000	18.000000	31.000000	...	2.
24	66.244994	66.244994	2.000000	66.244994	3.000000	5.000000	6.000000	8.000000	27.000000	17.000000	...	5.
25	66.244994	1.000000	1.000000	1.000000	2.000000	1.000000	2.000000	3.000000	6.000000	11.000000	...	2.
26	1.000000	66.244994	66.244994	66.244994	1.000000	3.000000	4.000000	8.000000	9.000000	4.000000	...	4.
27	66.244994	66.244994	1.000000	2.000000	66.244994	2.000000	1.000000	1.000000	10.000000	6.000000	...	4.
28	66.244994	1.000000	2.000000	66.244994	66.244994	2.000000	1.000000	1.000000	2.000000	7.000000	...	5.
29	66.244994	66.244994	66.244994	66.244994	1.000000	66.244994	66.244994	2.000000	2.000000	5.000000	...	3.
30	66.244994	66.244994	1.000000	66.244994	66.244994	66.244994	66.244994	2.000000	1.000000	1.000000	...	1.
31	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	2.000000	66.244994	66.244994	2.000000	...	4.
32	66.244994	66.244994	66.244994	66.244994	66.244994	1.000000	66.244994	66.244994	1.000000	5.000000	...	1.
33	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	...	66.
34	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	1.000000	1.000000	...	66.
35	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	1.000000	1.000000	66.244994	...	66.
36	66.244994	66.244994	66.244994	66.244994	2.000000	66.244994	66.244994	1.000000	1.000000	66.244994	...	66.
37	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	2.000000	66.244994	...	1.
38	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	...	66.

overall	48	49	50	51	52	53	54	55	56	57	...	
age												
38	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	...	66.
39	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	1.000000	...	66.
40	66.244994	1.000000	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	...	66.
41	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	...	66.
42	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	66.244994	...	66.

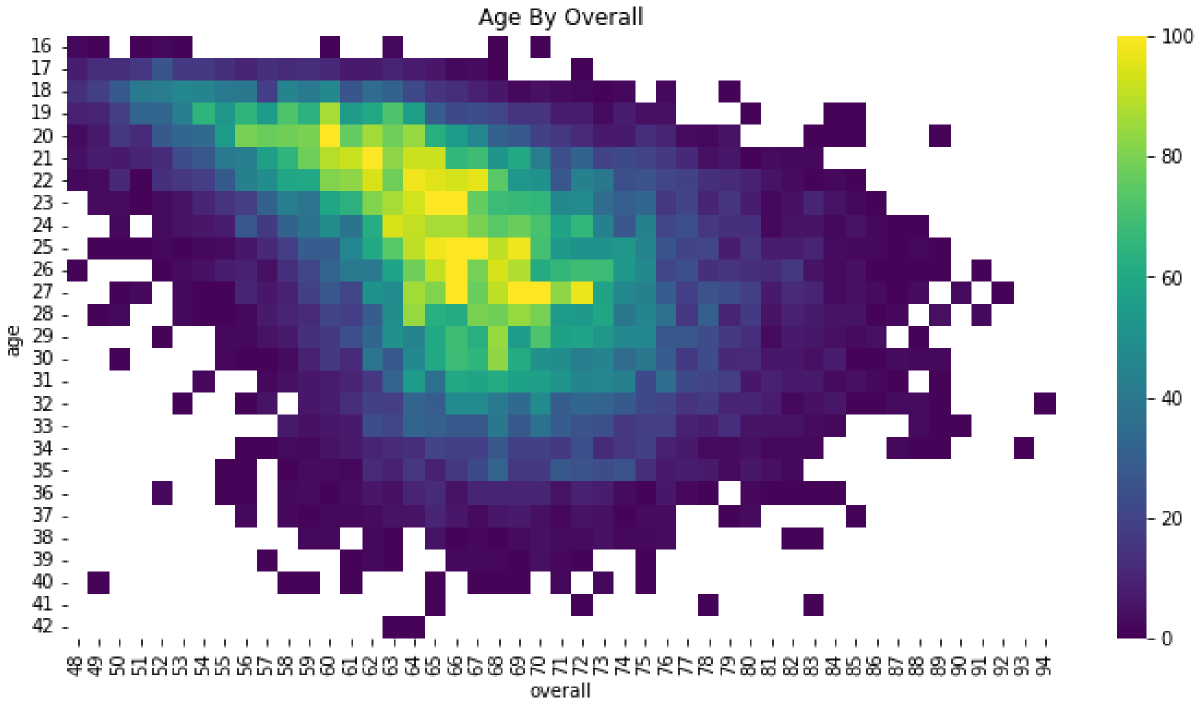
27 rows × 47 columns

In [49]:

```
plt.figure(figsize=(12,6))
sns.heatmap(groupedDf, cmap='viridis', vmin=0, vmax=100)
plt.title("Age By Overall")
```

Out[49]:

```
Text(0.5, 1.0, 'Age By Overall')
```



Checking correlations between weight and other parameters

In [51]:

```
print("Correlation between weight and agility: " , round(df['weight_kg'].corr(df['moveme
nt_agility']),2))

print("Correlation between weight and sprint speed: " , round(df['weight_kg'].corr(df['m
ovement_sprint_speed']),2))

print("Correlation between weight and Dribbling: " , round(df['weight_kg'].corr(df['drib
bling']),2))

print("Correlation between weight and Defending: " , round(df['weight_kg'].corr(df['defe
nding']),2))

print("Correlation between weight and pace: " , round(df['weight_kg'].corr(df['pace']),2
))
```

```
Correlation between weight and agility:  -0.55
Correlation between weight and sprint speed:  -0.42
Correlation between weight and Dribbling:  -0.27
Correlation between weight and Defending:  0.2
Correlation between weight and pace:  -0.35
```

In [54]:

```
print("Correlation between physic and defending: ", round(df['physic'].corr(df['defending']),2))
#print("Correlation between physic and pace: ", round(df['physic'].corr(df['pace']),2))
```

```
Correlation between physic and defending:  0.55
```

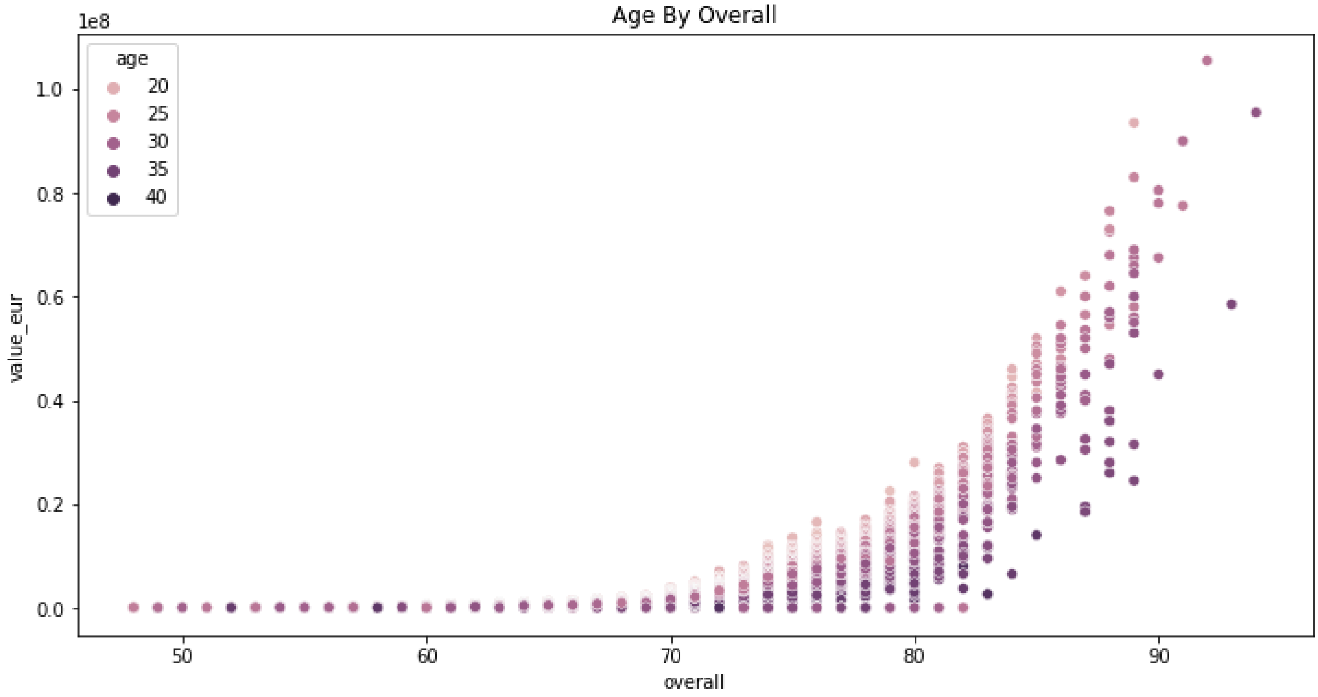
Scatterplot on Value By Overall based on Age

In [111]:

```
plt.figure(figsize=(12,6))
sns.scatterplot(x='overall', y='value_eur', data=df, hue='age')
plt.title("Age By Overall")
```

Out[111]:

```
Text(0.5, 1.0, 'Age By Overall')
```



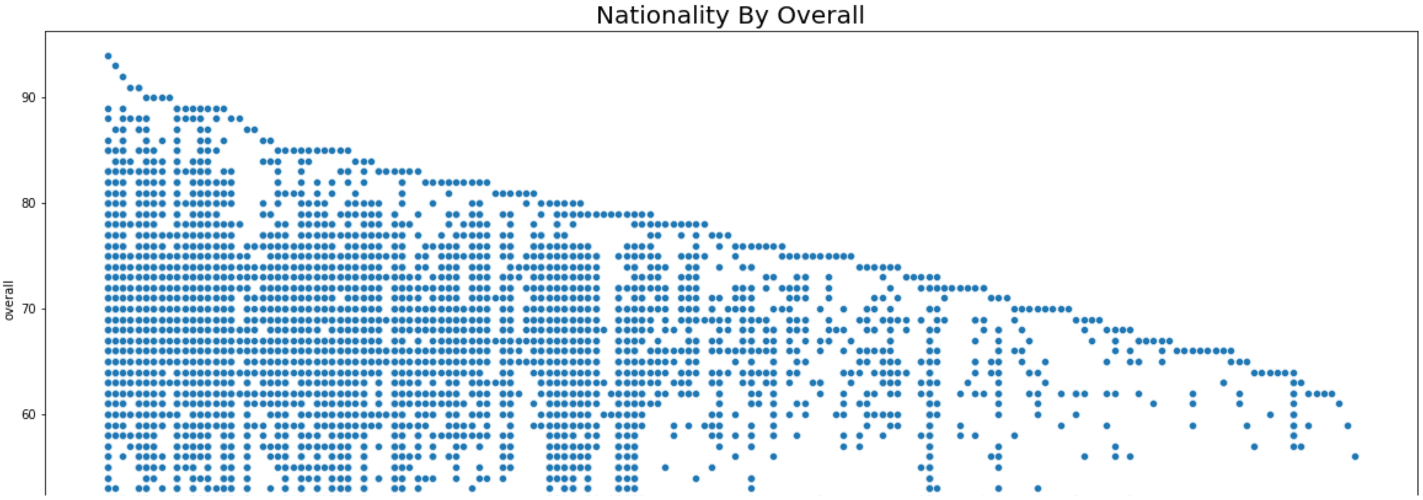
Scatterplot on Nationality and Overall

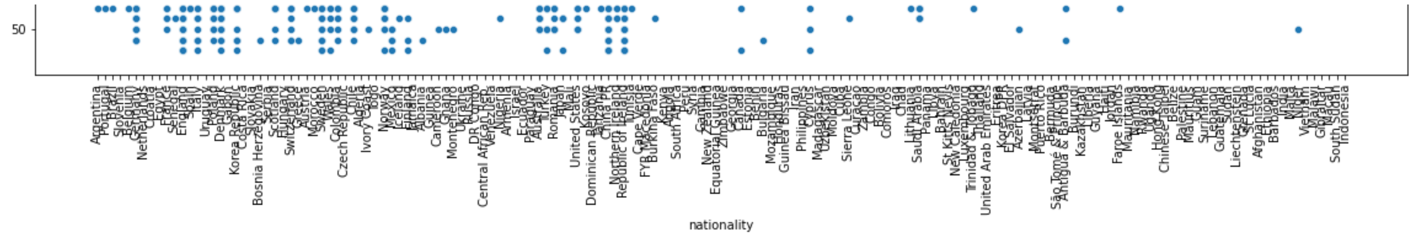
In [129]:

```
plt.figure(figsize=(20,8))
sns.scatterplot(x='nationality', y='overall', data=df)
plt.xticks(rotation=90)
plt.title("Nationality By Overall", fontsize=20)
```

Out[129]:

```
Text(0.5, 1.0, 'Nationality By Overall')
```



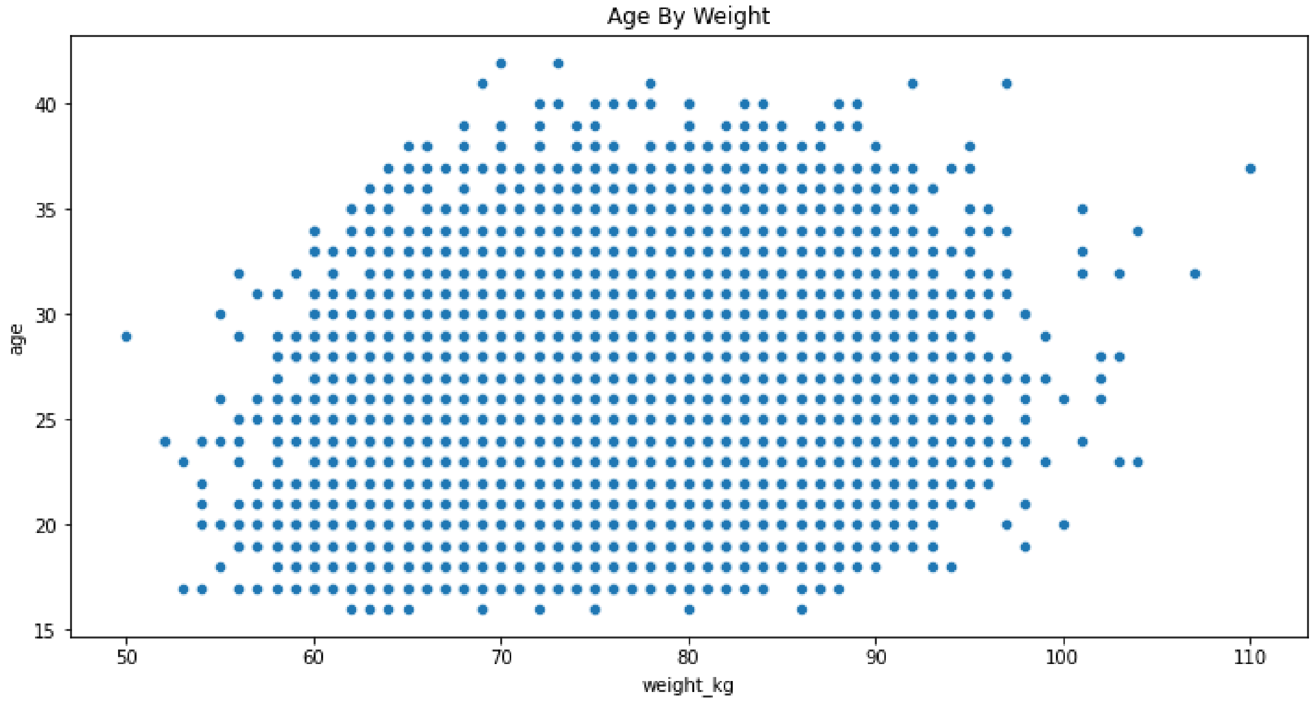


In [64]:

```
plt.figure(figsize=(12,6))
sns.scatterplot(x='weight_kg', y='age', data=df)
plt.title("Age By Weight")
```

Out[64]:

```
Text(0.5, 1.0, 'Age By Weight')
```



Changing data type of contract valid until column to integer

In [65]:

```
df['contract_valid_until'] = df['contract_valid_until'].fillna(float(2021))
df['contract_valid_until'] = df['contract_valid_until'].astype("int")
```

FC Barcelona players with contract expiring in 2021

In [66]:

```
year = 2021

df[(df['club'] == 'FC Barcelona') & (df['contract_valid_until'] == 2021)]
```

Out[66]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_sh
0	L. Messi	32	1987-06-24	170	72	Argentina	FC Barcelona	94	94	95500000	...	
19	L. Suárez	32	1987-01-24	182	86	Uruguay	FC Barcelona	89	89	53000000	...	
64	I. Rakitić	31	1988-03-10	184	78	Croatia	FC Barcelona	86	86	38000000	...	
142	A. Vidal	32	1987-05-22	180	75	Chile	FC Barcelona	84	84	23500000	...	

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_sh
4042	Riqui Puig	19	1999-08-13	169	56	Spain	FC Barcelona	71	87	5000000	...	
6634	Abel Ruiz	19	2000-01-28	182	73	Spain	FC Barcelona	68	84	1900000	...	
7711	Oriol Busquets	20	1999-06-20	185	77	Spain	FC Barcelona	67	82	1600000	...	
7713	Miranda	19	2000-01-19	185	76	Spain	FC Barcelona	67	82	1500000	...	
9938	Álex Collado	20	1999-04-22	177	66	Spain	FC Barcelona	65	80	1200000	...	
9970	Jorge Cuenca	19	1999-11-17	189	76	Spain	FC Barcelona	65	78	950000	...	
11042	Iñaki Peña	20	1999-03-02	184	78	Spain	FC Barcelona	64	81	850000	...	

11 rows × 69 columns

Youngest Players in FC Barcelona

In [67]:

```
df[(df['club'] == 'FC Barcelona')].sort_values(by='age', ascending=True)[:5]
```

Out[67]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_shot
4045	J. Todibo	19	1999-12-30	190	81	France	FC Barcelona	71	86	4800000	...	4
9970	Jorge Cuenca	19	1999-11-17	189	76	Spain	FC Barcelona	65	78	950000	...	3
7713	Miranda	19	2000-01-19	185	76	Spain	FC Barcelona	67	82	1500000	...	3
4042	Riqui Puig	19	1999-08-13	169	56	Spain	FC Barcelona	71	87	5000000	...	5
6634	Abel Ruiz	19	2000-01-28	182	73	Spain	FC Barcelona	68	84	1900000	...	6

5 rows × 69 columns

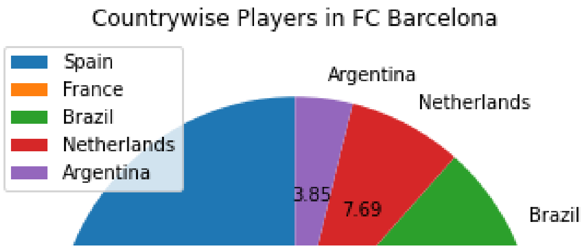
Players By Nationality In FC Barcelona

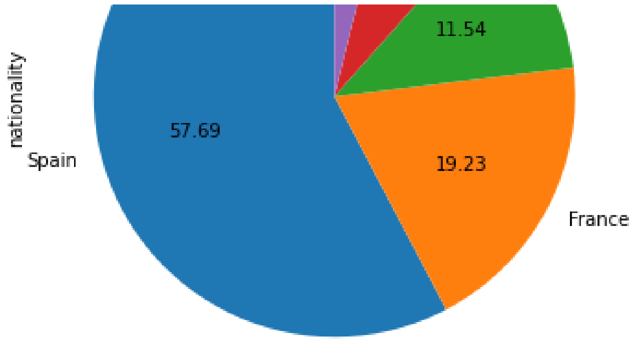
In [68]:

```
plt.figure(figsize=(10,6))
df[(df['club'] == 'FC Barcelona')]['nationality'].value_counts()[:5].plot(kind='pie', au
topct="%1.2f", startangle=90)
plt.title("Countrywise Players in FC Barcelona")
plt.legend()
```

Out[68]:

```
<matplotlib.legend.Legend at 0x2921f650e20>
```





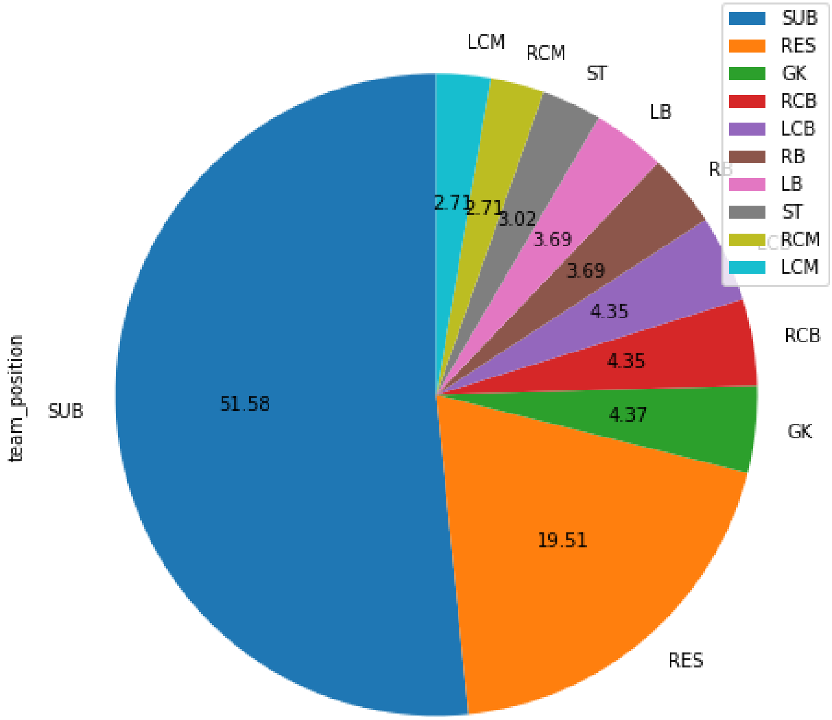
Pie Plot on player positions

In [133]:

```
df['team_position'].value_counts()[:10].plot(kind='pie', autopct='%1.2f', figsize=(20,8)
, startangle=90)
plt.legend()
```

Out[133]:

```
<matplotlib.legend.Legend at 0x292353dbbe0>
```



Top 10 Players With Contracts Expiring In 2021

In [71]:

```
df[df['contract_valid_until'] == 2021].sort_values(by='overall', ascending=False)[:10]
```

Out[71]:

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_sh
0	L. Messi	32	1987-06-24	170	72	Argentina	FC Barcelona	94	94	95500000	...	
17	S. Agüero	31	1988-06-02	173	70	Argentina	Manchester City	89	89	60000000	...	
19	L. Suárez	32	1987-01-24	182	86	Uruguay	FC Barcelona	89	89	53000000	...	

	short_name	age	dob	height_cm	weight_kg	nationality	club	overall	potential	value_eur	...	power_long_sho
20	R. Lewandowski	30	1988-08-21	184	80	Poland	FC Bayern München	89	89	64500000	...	
11	K. Koulibaly	28	1991-06-20	187	89	Senegal	Napoli	89	91	67500000	...	
24	P. Pogba	26	1993-03-15	191	84	France	Manchester United	88	91	72500000	...	
30	S. Handanovič	34	1984-07-14	193	92	Slovenia	Inter	88	88	26000000	...	
31	M. Neuer	33	1986-03-27	193	92	Germany	FC Bayern München	88	88	32000000	...	
38	P. Aubameyang	30	1989-06-18	187	80	Gabon	Arsenal	88	88	57000000	...	
52	Thiago	28	1991-04-11	174	70	Spain	FC Bayern München	87	87	50000000	...	

10 rows × 69 columns

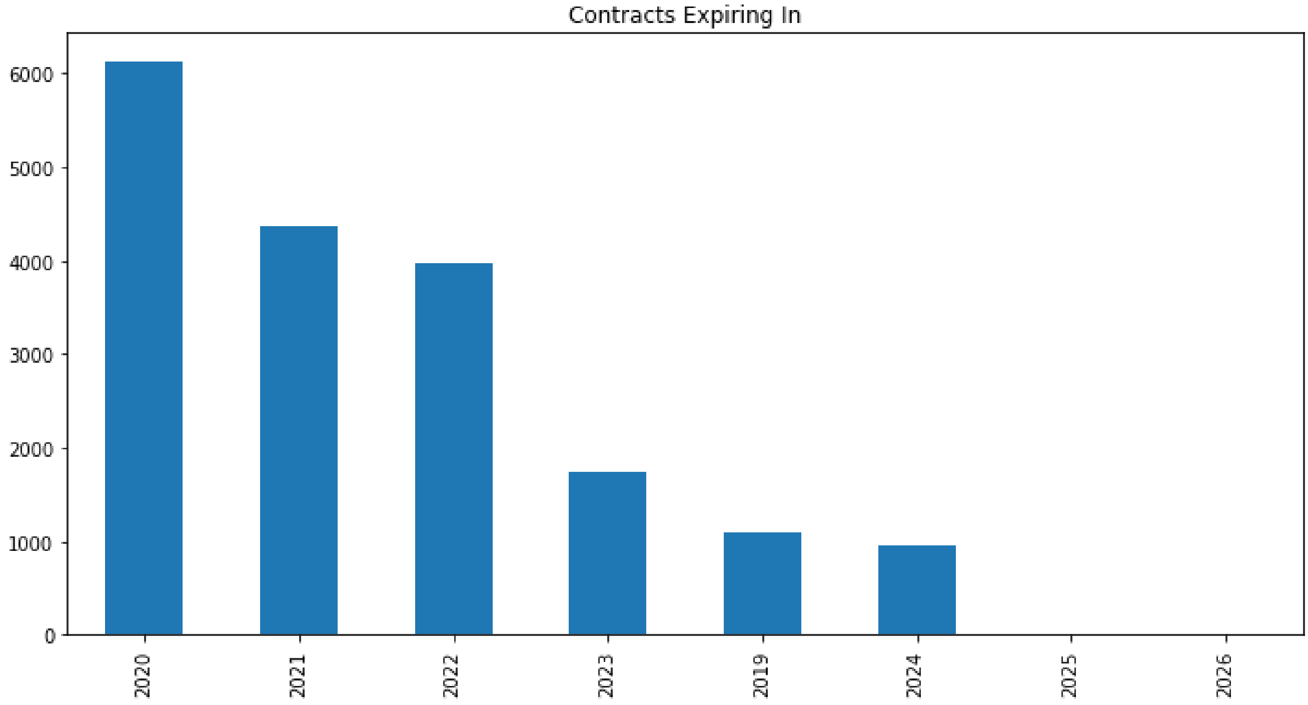
When are most contracts expiring?

In [80]:

```
df['contract_valid_until'].value_counts().plot(kind='bar', figsize=(12,6))
plt.title("Contracts Expiring In")
```

Out[80]:

```
Text(0.5, 1.0, 'Contracts Expiring In')
```



Which country has the most players in top 50 based on overall ratings?

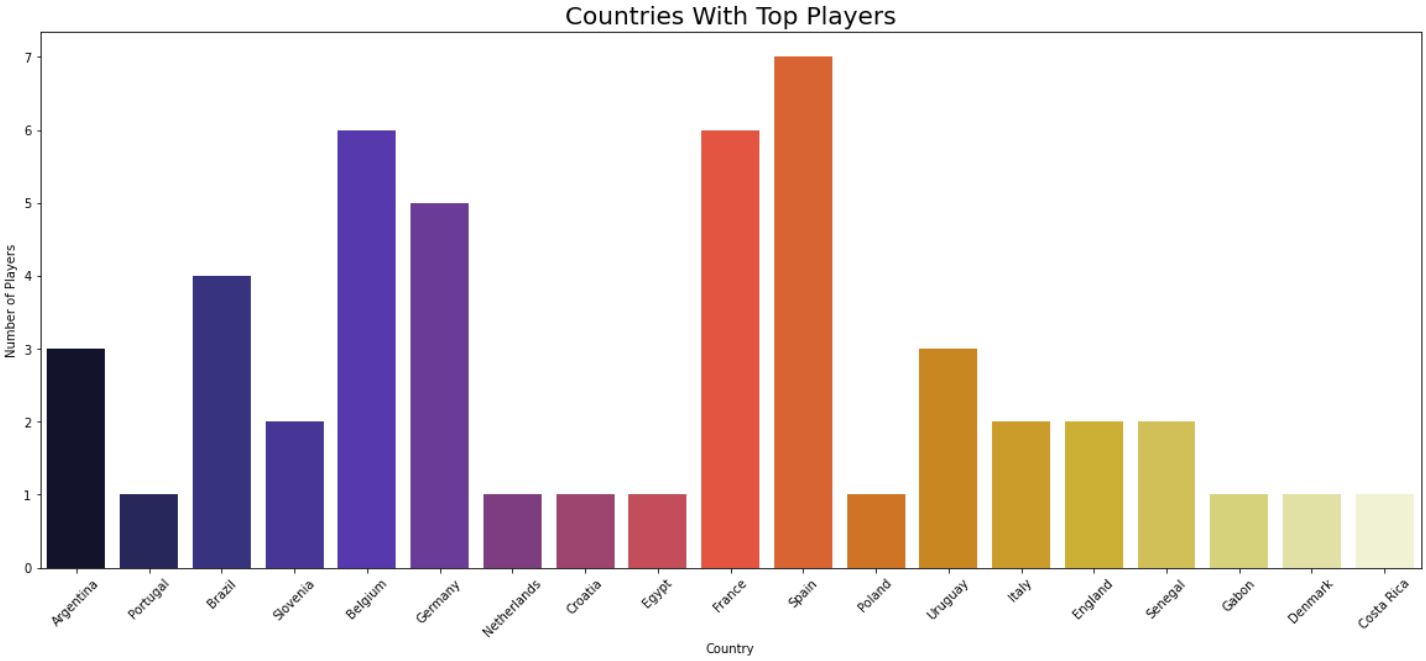
In [106]:

```
newdf = df.sort_values(by='overall', ascending=False)[:50]

plt.figure(figsize=(20,8))
sns.countplot(x='nationality', data=newdf, palette='CMRmap')
plt.xlabel("Country")
plt.xticks(rotation=45)
plt.ylabel("Number of Players")
plt.title("Countries With Top Players", fontsize=20)
```

Out[106]:

```
Text(0.5, 1.0, 'Countries With Top Players')
```



Which club has the most players in top 50?

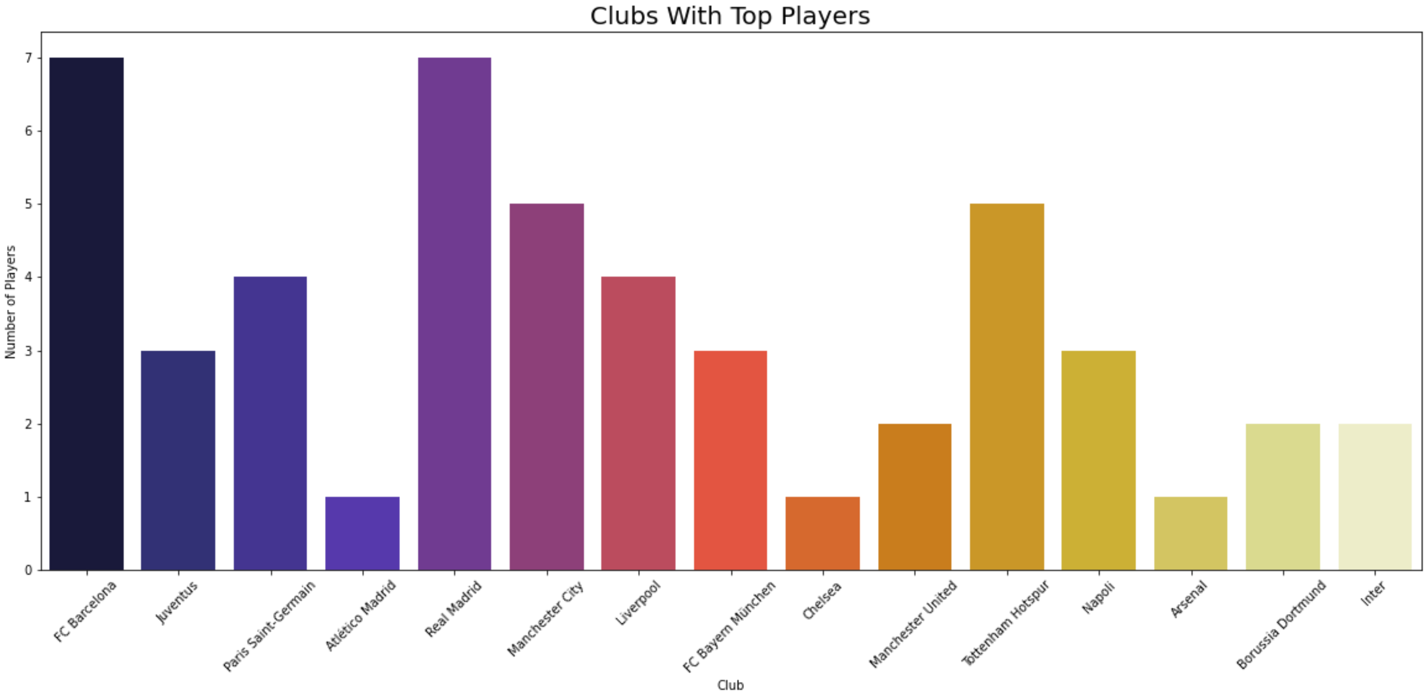
In [105]:

```
newdf = df.sort_values(by='overall', ascending=False)[:50]

plt.figure(figsize=(20,8))
sns.countplot(x='club', data=newdf, palette='CMRmap')
plt.xlabel("Club")
plt.xticks(rotation=45)
plt.ylabel("Number of Players")
plt.title("Clubs With Top Players", fontsize=20)
```

Out[105]:

```
Text(0.5, 1.0, 'Clubs With Top Players')
```



Which country has the highest paid players?

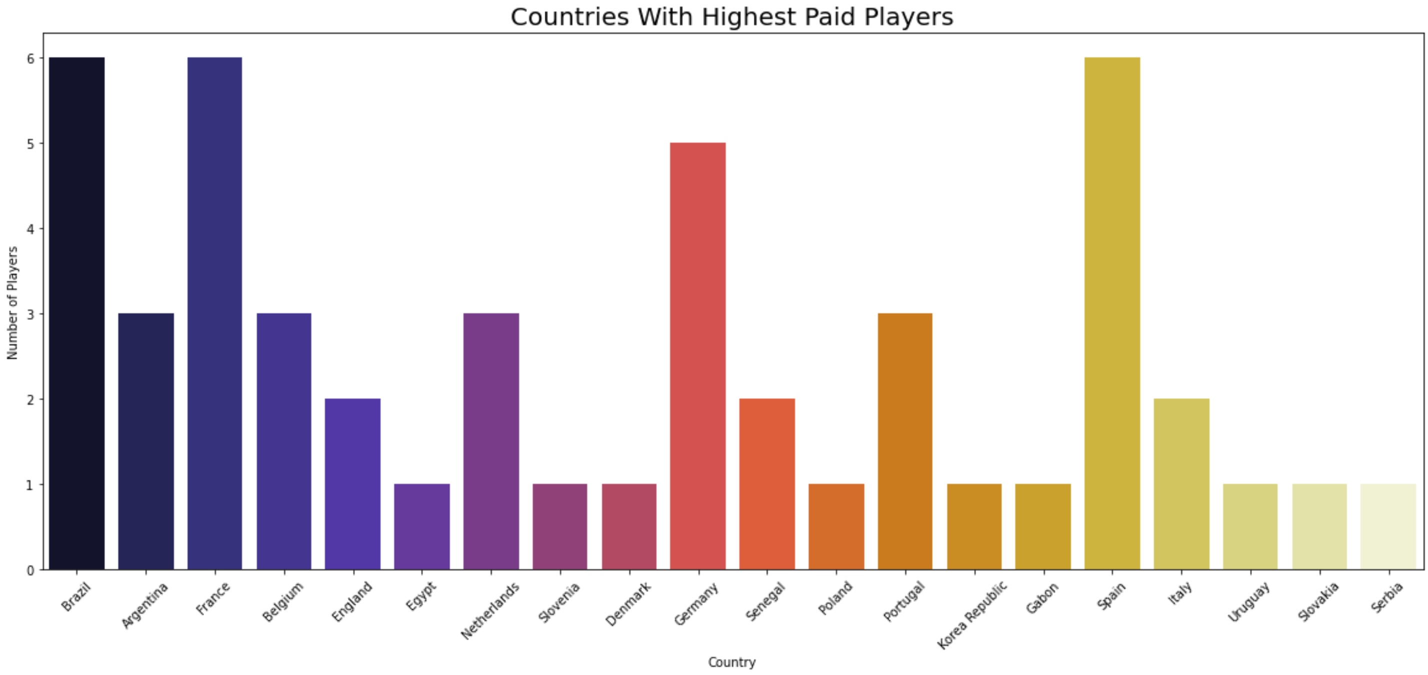
In [104]:

```
newdf = df.sort_values(by='value_eur', ascending=False)[:50]
```

```
plt.figure(figsize=(20,8))
sns.countplot(x='nationality', data=newdf, palette='CMRmap')
plt.xlabel("Country")
plt.xticks(rotation=45)
plt.ylabel("Number of Players")
plt.title("Countries With Highest Paid Players", fontsize=20)
```

Out[104]:

```
Text(0.5, 1.0, 'Countries With Highest Paid Players')
```



Which club pays more?

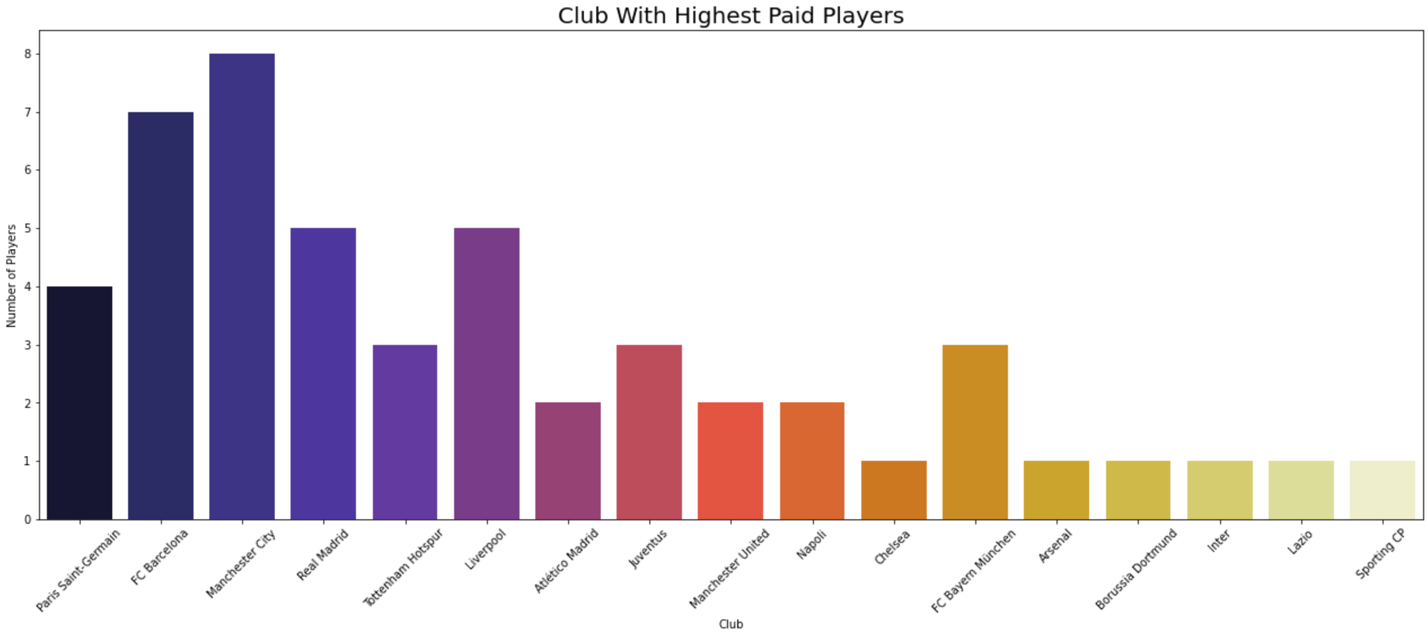
In [103]:

```
newdf = df.sort_values(by='value_eur', ascending=False)[:50]

plt.figure(figsize=(22,8))
sns.countplot(x='club', data=newdf, palette='CMRmap')
plt.xlabel("Club")
plt.xticks(rotation=45)
plt.ylabel("Number of Players")
plt.title("Club With Highest Paid Players", fontsize=20)
```

Out[103]:

```
Text(0.5, 1.0, 'Club With Highest Paid Players')
```



Which type of player comes more in top 50?

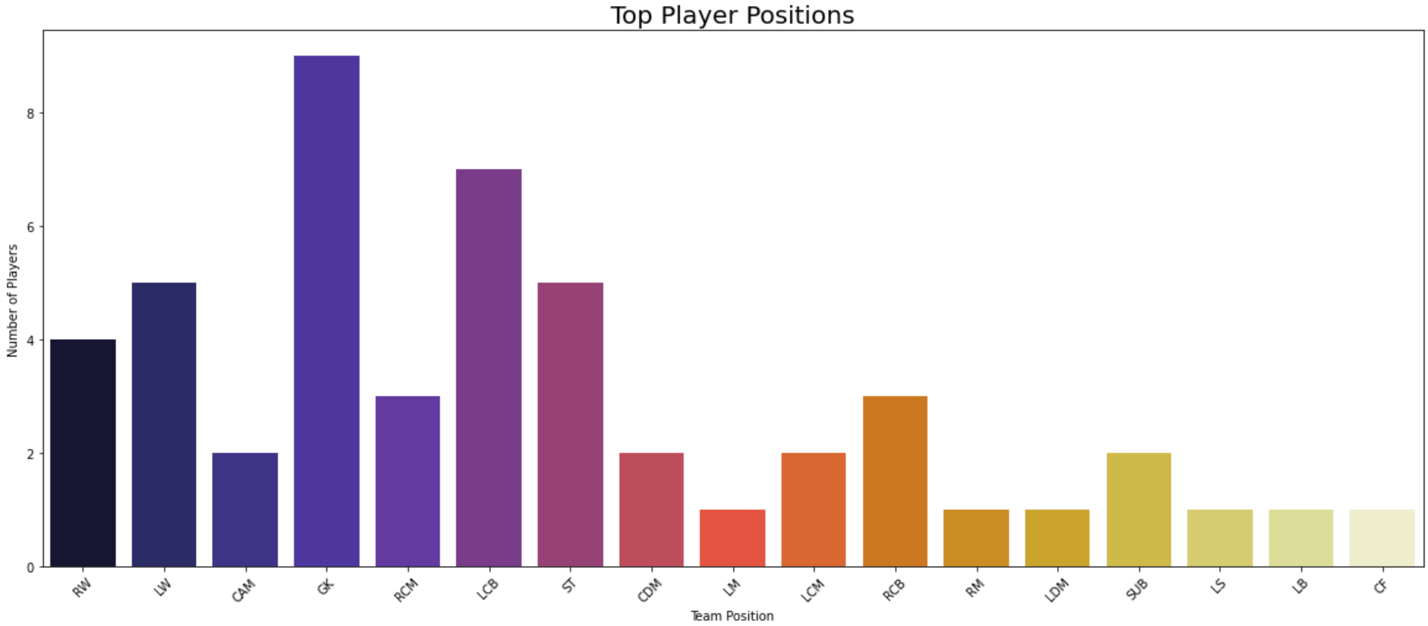
In [108]:

```
newdf = df.sort_values(by='overall', ascending=False)[:50]

plt.figure(figsize=(20,8))
sns.countplot(x='team_position', data=newdf, palette='CMRmap')
plt.xlabel("Team Position")
plt.xticks(rotation=45)
plt.ylabel("Number of Players")
plt.title("Top Player Positions", fontsize=20)
```

Out[108]:

```
Text(0.5, 1.0, 'Top Player Positions')
```



Which body type is more common in top 50 players?

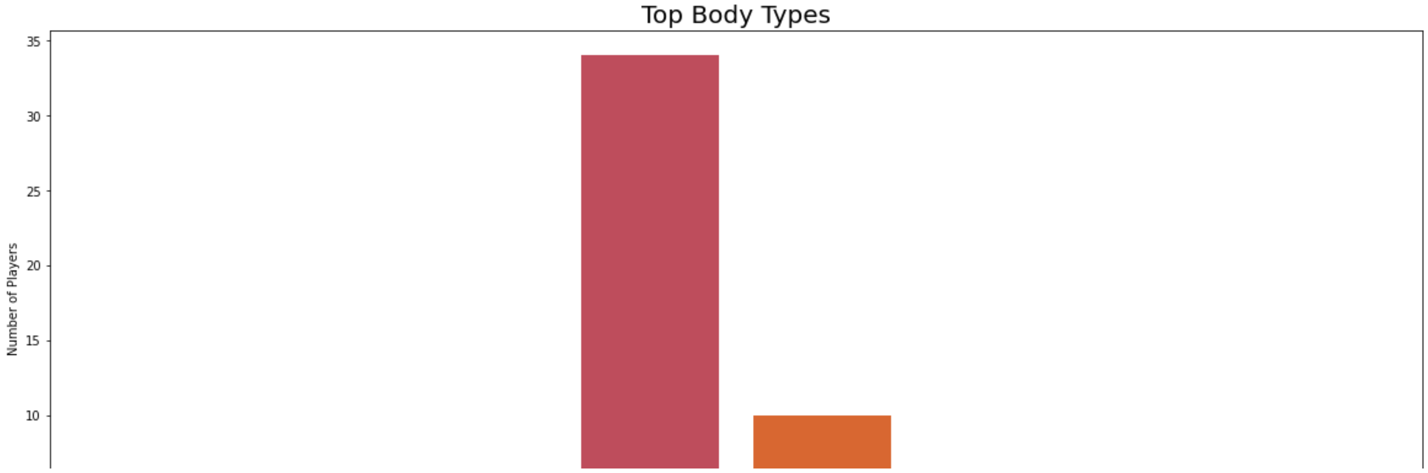
In [117]:

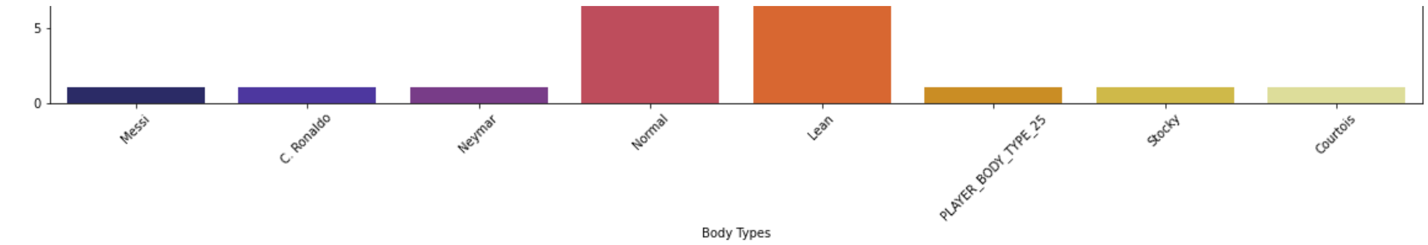
```
newdf = df.sort_values(by='overall', ascending=False)[:50]

plt.figure(figsize=(20,8))
sns.countplot(x='body_type', data=newdf, palette='CMRmap')
plt.xlabel("Body Types")
plt.xticks(rotation=45)
plt.ylabel("Number of Players")
plt.title("Top Body Types", fontsize=20)
```

Out[117]:

```
Text(0.5, 1.0, 'Top Body Types')
```





What is the age group in top 50?

In [119]:

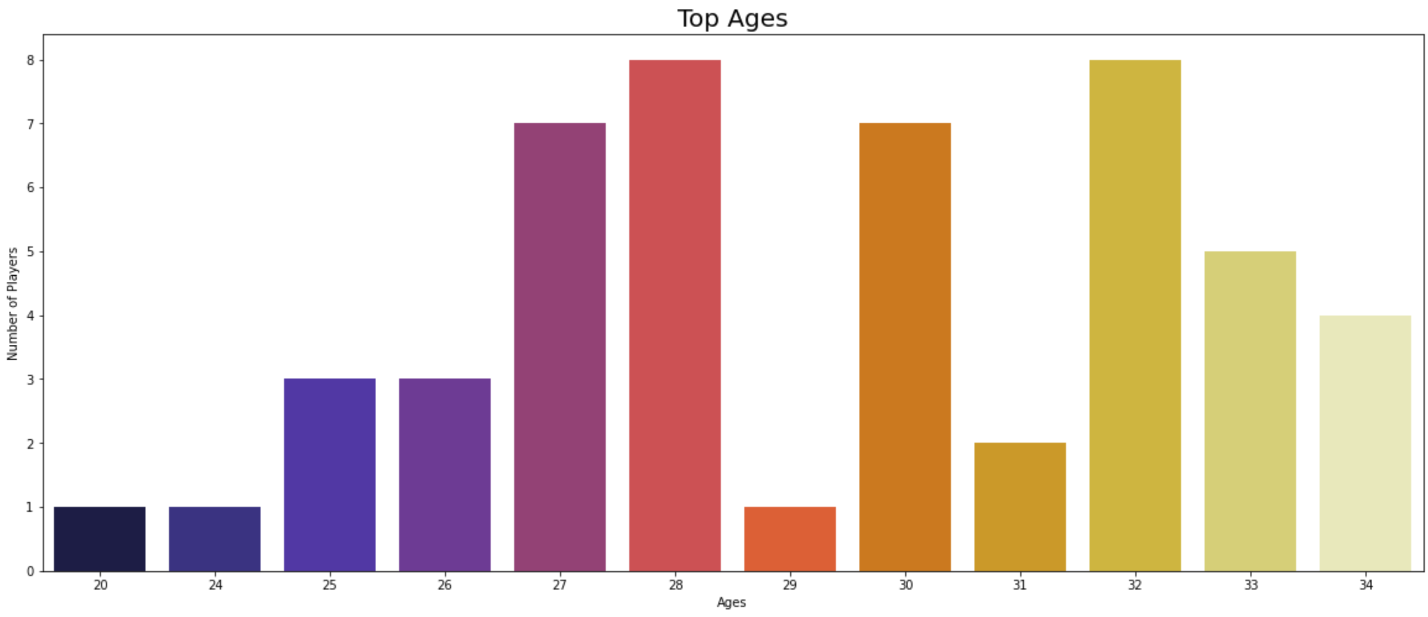
```
newdf = df.sort_values(by='overall', ascending=False)[:50]

plt.figure(figsize=(20,8))
sns.countplot(x='age', data=newdf, palette='CMRmap')
plt.xlabel("Ages")

plt.ylabel("Number of Players")
plt.title("Top Ages", fontsize=20)
```

Out[119]:

```
Text(0.5, 1.0, 'Top Ages')
```



Key Findings/Observations

- **Best player based on overall ratings is Lionel Messi With Rating of 94.**
- **Best player based on Potential ratings is Kylian Mbappe with Rating of 95.**
- **Best defenders are Virgil Van Djik and G. Chiellini with overall rating of 90.**
- **Oldest player is C. Minoz with age 42.**
- **Youngest player is A. Hložek with age 16.**
- **Average Height Of Players: 181.36.**
- **Average Weight Of Players: 75.28.**
- **Most players in the dataset are from England.**
- **FC Barcelona leads with most players in the dataset in club category.**
- **The dataset contains 76.4 right-foot players, and 23.6 left-foot players**
- **Best right foot player is Cristiano Ronaldo**
- **Best right foot defender is V Van Djik**
- **Best right foot GK is J. Oblak**
- **Best left foot player is Lionel Messi**
- **Best left foot defender is G. Chiellini**
- **Best left foot GK is Ederson**
- **As weight increases agility decreases, correlation: -0.55**
- **As weight increases sprint speed decreases, correlation: -0.42**
- **As weight increases Dribbling skills decrease, correlation: -0.27**

- **Weight is helpful in better Defending, correlation: 0.2**
- **Weight decreases pace significantly, Correlation: -0.35**
- **As per the dataset, most contracts were expiring in 2020, followed by 2021.**
- **Spain tops the list of countries with best overall ratings for players.**
- **There's a close competition between FC Barcelona and Real Madrid in getting top players.**
- **Spanish, Brazilian, and French players are have significantly high values in euros.**
- **Manchester city has the most valued players, followed by FC Barcelona and Real Madrid.**
- **Most High rated players play in GK, LCB, and ST positions.**
- **Normal is the most common body type in top 50 players based on overall ratings**
- **Most top players are between age 28 to 32. So it can be implied that players perform at their peak during this age group.**