Importing Required Libraries

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
from sklearn.model_selection import train_test_split
```

Business Understanding

Being a football fan and local famous striker means exploring FIFA19 player datsset could be so much fun.

I will focus on the three question below:

Q1: What's the ratio of total wages/ total potential for clubs. Which clubs are the most economical ?

Q2: What's the age distribution like? How is it related to player's overall rating?

Q3: How is a player's skils set influence his potential? Can we predict a player's potential based on his skills' set?

Loading Dataset

In [2]:

```
df_players= pd.read_csv('data.csv')
```

In [3]:

```
df_players.head(10)
```

Out[3]:

	Unnamed: 0	ID	Name	Age	Photo	Nationality	Flag	Overall	Pot
0	0	158023	L. Messi	31	https://cdn.sofifa.org/players/4/19/158023.png	Argentina	https://cdn.sofifa.org/flags/52.png	94	
1	1	20801	Cristiano Ronaldo	33	https://cdn.sofifa.org/players/4/19/20801.png	Portugal	https://cdn.sofifa.org/flags/38.png	94	
2	2	190871	Neymar Jr	26	https://cdn.sofifa.org/players/4/19/190871.png	Brazil	https://cdn.sofifa.org/flags/54.png	92	
3	3	193080	De Gea	27	https://cdn.sofifa.org/players/4/19/193080.png	Spain	https://cdn.sofifa.org/flags/45.png	91	
4	4	192985	K. De Bruyne	27	https://cdn.sofifa.org/players/4/19/192985.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	
5	5	183277	E. Hazard	27	https://cdn.sofifa.org/players/4/19/183277.png	Belgium	https://cdn.sofifa.org/flags/7.png	91	
6	6	177003	L. Modrić	32	https://cdn.sofifa.org/players/4/19/177003.png	Croatia	https://cdn.sofifa.org/flags/10.png	91	
7	7	176580	L. Suárez	31	https://cdn.sofifa.org/players/4/19/176580.png	Uruguay	https://cdn.sofifa.org/flags/60.png	91	
8	8	155862	Sergio Ramos	32	https://cdn.sofifa.org/players/4/19/155862.png	Spain	https://cdn.sofifa.org/flags/45.png	91	
9	9	200389	J. Oblak	25	https://cdn.sofifa.org/players/4/19/200389.png	Slovenia	https://cdn.sofifa.org/flags/44.png	90	

10 rows × 89 columns

In [4]:

In [4]:

```
df_players.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18207 entries, 0 to 18206
Data columns (total 89 columns):
Unnamed: 0                  18207 non-null int64
ID                          18207 non-null int64
Name                        18207 non-null object
Age                         18207 non-null int64
Photo                       18207 non-null object
Nationality                 18207 non-null object
Flag                        18207 non-null object
Overall                     18207 non-null int64
Potential                   18207 non-null int64
Club                        17966 non-null object
Club Logo                   18207 non-null object
Value                       18207 non-null object
Wage                        18207 non-null object
Special                     18207 non-null int64
Preferred Foot              18159 non-null object
International Reputation    18159 non-null float64
Weak Foot                   18159 non-null float64
Skill Moves                 18159 non-null float64
Work Rate                   18159 non-null object
Body Type                   18159 non-null object
Real Face                   18159 non-null object
Position                    18147 non-null object
Jersey Number               18147 non-null float64
Joined                      16654 non-null object
Loaned From                 1264 non-null object
Contract Valid Until        17918 non-null object
Height                      18159 non-null object
Weight                      18159 non-null object
LS                          16122 non-null object
ST                          16122 non-null object
RS                          16122 non-null object
LW                          16122 non-null object
LF                          16122 non-null object
CF                          16122 non-null object
RF                          16122 non-null object
RW                          16122 non-null object
LAM                         16122 non-null object
CAM                         16122 non-null object
RAM                         16122 non-null object
LM                          16122 non-null object
LCM                         16122 non-null object
CM                          16122 non-null object
RCM                         16122 non-null object
RM                          16122 non-null object
LWB                         16122 non-null object
LDM                         16122 non-null object
CDM                         16122 non-null object
RDM                         16122 non-null object
RWB                         16122 non-null object
LB                          16122 non-null object
LCB                         16122 non-null object
CB                          16122 non-null object
RCB                         16122 non-null object
RB                          16122 non-null object
Crossing                    18159 non-null float64
Finishing                   18159 non-null float64
HeadingAccuracy             18159 non-null float64
ShortPassing                18159 non-null float64
Volleys                     18159 non-null float64
Dribbling                   18159 non-null float64
Curve                       18159 non-null float64
FKAccuracy                  18159 non-null float64
LongPassing                 18159 non-null float64
BallControl                 18159 non-null float64
Acceleration                18159 non-null float64
SprintSpeed                 18159 non-null float64
Agility                     18159 non-null float64
Reactions                   18159 non-null float64
Balance                     18159 non-null float64
ShotPower                   18159 non-null float64
```

```
Jumping                     18159 non-null float64
Stamina                     18159 non-null float64
Strength                    18159 non-null float64
LongShots                   18159 non-null float64
Aggression                  18159 non-null float64
Interceptions               18159 non-null float64
Positioning                 18159 non-null float64
Vision                      18159 non-null float64
Penalties                   18159 non-null float64
Composure                   18159 non-null float64
Marking                     18159 non-null float64
StandingTackle              18159 non-null float64
SlidingTackle               18159 non-null float64
GKDiving                    18159 non-null float64
GKHandling                  18159 non-null float64
GKKicking                   18159 non-null float64
GKPositioning               18159 non-null float64
GKReflexes                  18159 non-null float64
Release Clause              16643 non-null object
dtypes: float64(38), int64(6), object(45)
memory usage: 12.4+ MB
```

In [5]:

```
df_players.columns
```

Out[5]:

```
Index(['Unnamed: 0', 'ID', 'Name', 'Age', 'Photo', 'Nationality', 'Flag',
       'Overall', 'Potential', 'Club', 'Club Logo', 'Value', 'Wage', 'Special',
       'Preferred Foot', 'International Reputation', 'Weak Foot',
       'Skill Moves', 'Work Rate', 'Body Type', 'Real Face', 'Position',
       'Jersey Number', 'Joined', 'Loaned From', 'Contract Valid Until',
       'Height', 'Weight', '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', 'Crossing',
       'Finishing', 'HeadingAccuracy', 'ShortPassing', 'Volleys', 'Dribbling',
       'Curve', 'FKAccuracy', 'LongPassing', 'BallControl', 'Acceleration',
       'SprintSpeed', 'Agility', 'Reactions', 'Balance', 'ShotPower',
       'Jumping', 'Stamina', 'Strength', 'LongShots', 'Aggression',
       'Interceptions', 'Positioning', 'Vision', 'Penalties', 'Composure',
       'Marking', 'StandingTackle', 'SlidingTackle', 'GKDiving', 'GKHandling',
       'GKKicking', 'GKPositioning', 'GKReflexes', 'Release Clause'],
      dtype='object')
```

In [6]:

```
df_players.describe()
```

Out[6]:

	Unnamed: 0	ID	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Mov
count	18207.000000	18207.000000	18207.000000	18207.000000	18207.000000	18207.000000	18159.000000	18159.000000	18159.0000
mean	9103.000000	214298.338606	25.122206	66.238699	71.307299	1597.809908	1.113222	2.947299	2.3613
std	5256.052511	29965.244204	4.669943	6.908930	6.136496	272.586016	0.394031	0.660456	0.7561
min	0.000000	16.000000	16.000000	46.000000	48.000000	731.000000	1.000000	1.000000	1.0000
25%	4551.500000	200315.500000	21.000000	62.000000	67.000000	1457.000000	1.000000	3.000000	2.0000
50%	9103.000000	221759.000000	25.000000	66.000000	71.000000	1635.000000	1.000000	3.000000	2.0000
75%	13654.500000	236529.500000	28.000000	71.000000	75.000000	1787.000000	1.000000	3.000000	3.0000
max	18206.000000	246620.000000	45.000000	94.000000	95.000000	2346.000000	5.000000	5.000000	5.0000

8 rows × 44 columns

In [7]:

```
df_players.isnull().sum()
```

Out[7]:

```
Unnamed: 0                      0
ID                              0
Name                            0
Age                             0
Photo                           0
Nationality                     0
Flag                            0
Overall                         0
Potential                       0
Club                          241
Club Logo                       0
Value                           0
Wage                            0
Special                         0
Preferred Foot                 48
International Reputation       48
Weak Foot                      48
Skill Moves                    48
Work Rate                      48
Body Type                      48
Real Face                      48
Position                       60
Jersey Number                  60
Joined                       1553
Loaned From                 16943
Contract Valid Until          289
Height                         48
Weight                         48
LS                           2085
ST                           2085
                             ...
Dribbling                      48
Curve                          48
FKAccuracy                     48
LongPassing                    48
BallControl                    48
Acceleration                   48
SprintSpeed                    48
Agility                        48
Reactions                      48
Balance                        48
ShotPower                      48
Jumping                        48
Stamina                        48
Strength                       48
LongShots                      48
Aggression                     48
Interceptions                  48
Positioning                    48
Vision                         48
Penalties                      48
Composure                      48
Marking                        48
StandingTackle                 48
SlidingTackle                  48
GKDiving                       48
GKHandling                     48
GKKicking                      48
GKPositioning                  48
GKReflexes                     48
Release Clause               1564
Length: 89, dtype: int64
```

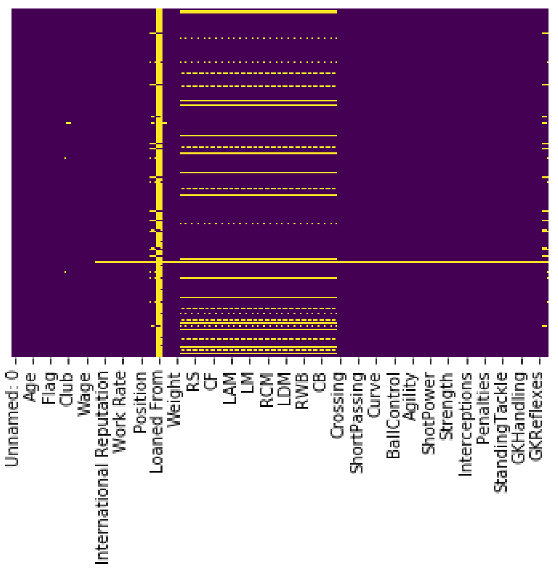
Cleaning The Data

In [8]:

```
sns.heatmap(df_players.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[8]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d037ff08d0>
```



In [9]:

```
df_players.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18207 entries, 0 to 18206
Data columns (total 89 columns):
Unnamed: 0                  18207 non-null int64
ID                          18207 non-null int64
Name                        18207 non-null object
Age                         18207 non-null int64
Photo                       18207 non-null object
Nationality                 18207 non-null object
Flag                        18207 non-null object
Overall                     18207 non-null int64
Potential                   18207 non-null int64
Club                        17966 non-null object
Club Logo                   18207 non-null object
Value                       18207 non-null object
Wage                        18207 non-null object
Special                     18207 non-null int64
Preferred Foot              18159 non-null object
International Reputation    18159 non-null float64
Weak Foot                   18159 non-null float64
Skill Moves                 18159 non-null float64
Work Rate                   18159 non-null object
Body Type                   18159 non-null object
Real Face                   18159 non-null object
Position                    18147 non-null object
Jersey Number               18147 non-null float64
Joined                      16654 non-null object
Loaned From                 1264 non-null object
Contract Valid Until        17918 non-null object
Height                      18159 non-null object
Weight                      18159 non-null object
LS                          16122 non-null object
ST                          16122 non-null object
RS                          16122 non-null object
LW                          16122 non-null object
LF                          16122 non-null object
CF                          16122 non-null object
RF                          16122 non-null object
RW                          16122 non-null object
LAM                         16122 non-null object
CAM                         16122 non-null object
RAM                         16122 non-null object
LM                          16122 non-null object
LCM                         16122 non-null object
CM                          16122 non-null object
RCM                         16122 non-null object
RM                          16122 non-null object
```

```
LWB                            16122 non-null object
LDM                            16122 non-null object
CDM                            16122 non-null object
RDM                            16122 non-null object
RWB                            16122 non-null object
LB                             16122 non-null object
LCB                            16122 non-null object
CB                             16122 non-null object
RCB                            16122 non-null object
RB                             16122 non-null object
Crossing                       18159 non-null float64
Finishing                      18159 non-null float64
HeadingAccuracy                18159 non-null float64
ShortPassing                   18159 non-null float64
Volleys                        18159 non-null float64
Dribbling                      18159 non-null float64
Curve                          18159 non-null float64
FKAccuracy                     18159 non-null float64
LongPassing                    18159 non-null float64
BallControl                    18159 non-null float64
Acceleration                   18159 non-null float64
SprintSpeed                    18159 non-null float64
Agility                        18159 non-null float64
Reactions                      18159 non-null float64
Balance                        18159 non-null float64
ShotPower                      18159 non-null float64
Jumping                        18159 non-null float64
Stamina                        18159 non-null float64
Strength                       18159 non-null float64
LongShots                      18159 non-null float64
Aggression                     18159 non-null float64
Interceptions                  18159 non-null float64
Positioning                    18159 non-null float64
Vision                         18159 non-null float64
Penalties                      18159 non-null float64
Composure                      18159 non-null float64
Marking                        18159 non-null float64
StandingTackle                 18159 non-null float64
SlidingTackle                  18159 non-null float64
GKDiving                       18159 non-null float64
GKHandling                     18159 non-null float64
GKKicking                      18159 non-null float64
GKPositioning                  18159 non-null float64
GKReflexes                     18159 non-null float64
Release Clause                 16643 non-null object
dtypes: float64(38), int64(6), object(45)
memory usage: 12.4+ MB
```

3. Prepare Data

There are some necessary stpes to apply before continue exploring the dataset:

Drop unused columns

Convert string values to number

Handle missing values, drop them if necessary

In [10]:

```
columns_to_drop = ['Unnamed: 0', 'ID', 'Photo', 'Flag','Club Logo', 'Preferred Foot',
                   'Body Type', 'Real Face', 'Jersey Number', 'Joined', 'Loaned From',
                   'Contract Valid Until', 'Height', 'Weight','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', 'Release Clause']
```

In [11]:

```
df_players.drop(columns_to_drop,axis=1,inplace=True)
```

In [12]:

```
df_players.head()
```

Out[12]:

	Name	Age	Nationality	Overall	Potential	Club	Value	Wage	Special	International Reputation	...	Penalties	Composure	Markin
0	L. Messi	31	Argentina	94	94	FC Barcelona	€110.5M	€565K	2202	5.0	...	75.0	96.0	33.
1	Cristiano Ronaldo	33	Portugal	94	94	Juventus	€77M	€405K	2228	5.0	...	85.0	95.0	28.
2	Neymar Jr	26	Brazil	92	93	Paris Saint-Germain	€118.5M	€290K	2143	5.0	...	81.0	94.0	27.
3	De Gea	27	Spain	91	93	Manchester United	€72M	€260K	1471	4.0	...	40.0	68.0	15.
4	K. De Bruyne	27	Belgium	91	92	Manchester City	€102M	€355K	2281	4.0	...	79.0	88.0	68.

5 rows × 48 columns

In [13]:

```
df_players.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 18207 entries, 0 to 18206
Data columns (total 48 columns):
Name                        18207 non-null object
Age                         18207 non-null int64
Nationality                 18207 non-null object
Overall                     18207 non-null int64
Potential                   18207 non-null int64
Club                        17966 non-null object
Value                       18207 non-null object
Wage                        18207 non-null object
Special                     18207 non-null int64
International Reputation    18159 non-null float64
Weak Foot                   18159 non-null float64
Skill Moves                 18159 non-null float64
Work Rate                   18159 non-null object
Position                    18147 non-null object
Crossing                    18159 non-null float64
Finishing                   18159 non-null float64
HeadingAccuracy             18159 non-null float64
ShortPassing                18159 non-null float64
Volleys                     18159 non-null float64
Dribbling                   18159 non-null float64
Curve                       18159 non-null float64
FKAccuracy                  18159 non-null float64
LongPassing                 18159 non-null float64
BallControl                 18159 non-null float64
Acceleration                18159 non-null float64
SprintSpeed                 18159 non-null float64
Agility                     18159 non-null float64
Reactions                   18159 non-null float64
Balance                     18159 non-null float64
ShotPower                   18159 non-null float64
Jumping                     18159 non-null float64
Stamina                     18159 non-null float64
Strength                    18159 non-null float64
LongShots                   18159 non-null float64
Aggression                  18159 non-null float64
Interceptions               18159 non-null float64
Positioning                 18159 non-null float64
Vision                      18159 non-null float64
Penalties                   18159 non-null float64
Composure                   18159 non-null float64
Marking                     18159 non-null float64
StandingTackle              18159 non-null float64
SlidingTackle               18159 non-null float64
GKDiving                    18159 non-null float64
GKHandling                  18159 non-null float64
GKKicking                   18159 non-null float64
```

```
GKPositioning                18159 non-null float64
GKReflexes                   18159 non-null float64
dtypes: float64(37), int64(4), object(7)
memory usage: 6.7+ MB
```

In [14]:

```python
def string_to_number(amount):

    if amount[-1] == 'M':
        return float(amount[1:-1])*1000000
    elif amount[-1] == 'K':
        return float(amount[1:-1])*1000
    else:
        return float(amount[1:])
```

In [15]:

```python
df_players['Value_M'] = df_players['Value'].apply(lambda x: string_to_number(x)/1000000)
df_players['Wages_K'] = df_players['Wage'].apply(lambda x :string_to_number(x)/1000)
```

In [16]:

```python
df_players.drop(['Value','Wage'],axis=1,inplace=True)
```

In [17]:

```python
df_players.head()
```

Out[17]:

	Name	Age	Nationality	Overall	Potential	Club	Special	International Reputation	Weak Foot	Skill Moves	...	Marking	StandingTackle	Sliding
0	L. Messi	31	Argentina	94	94	FC Barcelona	2202	5.0	4.0	4.0	...	33.0	28.0	
1	Cristiano Ronaldo	33	Portugal	94	94	Juventus	2228	5.0	4.0	5.0	...	28.0	31.0	
2	Neymar Jr	26	Brazil	92	93	Paris Saint-Germain	2143	5.0	5.0	5.0	...	27.0	24.0	
3	De Gea	27	Spain	91	93	Manchester United	1471	4.0	3.0	1.0	...	15.0	21.0	
4	K. De Bruyne	27	Belgium	91	92	Manchester City	2281	4.0	5.0	4.0	...	68.0	58.0	

5 rows × 48 columns

In [18]:

```python
df_players.describe()
```

Out[18]:

	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Crossing	Finishin
count	18207.000000	18207.000000	18207.000000	18207.000000	18159.000000	18159.000000	18159.000000	18159.000000	18159.00000
mean	25.122206	66.238699	71.307299	1597.809908	1.113222	2.947299	2.361308	49.734181	45.55091
std	4.669943	6.908930	6.136496	272.586016	0.394031	0.660456	0.756164	18.364524	19.52582
min	16.000000	46.000000	48.000000	731.000000	1.000000	1.000000	1.000000	5.000000	2.00000
25%	21.000000	62.000000	67.000000	1457.000000	1.000000	3.000000	2.000000	38.000000	30.00000
50%	25.000000	66.000000	71.000000	1635.000000	1.000000	3.000000	2.000000	54.000000	49.00000
75%	28.000000	71.000000	75.000000	1787.000000	1.000000	3.000000	3.000000	64.000000	62.00000
max	45.000000	94.000000	95.000000	2346.000000	5.000000	5.000000	5.000000	93.000000	95.00000

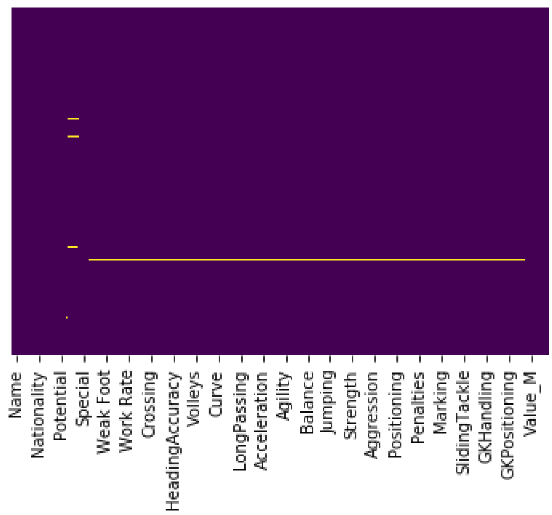
8 rows × 43 columns

In [19]:

```
sns.heatmap(df_players.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[19]:

<matplotlib.axes._subplots.AxesSubplot at 0x1d0386a22b0>



In [20]:

```
df_missing_players = df_players[df_players['Curve'].isnull()]
```

In [21]:

```
df_missing_players.sample(10)
```

Out[21]:

	Name	Age	Nationality	Overall	Potential	Club	Special	International Reputation	Weak Foot	Skill Moves	...	Marking	StandingTackle
13246	I. Sissoko	22	France	62	68	AS Béziers	1494	NaN	NaN	NaN	...	NaN	NaN
13280	Y. Ammour	19	France	62	77	Montpellier HSC	1478	NaN	NaN	NaN	...	NaN	NaN
13247	F. Hart	28	Austria	62	62	SV Mattersburg	1630	NaN	NaN	NaN	...	NaN	NaN
13279	P. Mazzocchi	22	Italy	62	69	Perugia	1681	NaN	NaN	NaN	...	NaN	NaN
13261	H. Al Mansour	25	Saudi Arabia	62	64	Al Nassr	1665	NaN	NaN	NaN	...	NaN	NaN
13240	R. Bingham	24	England	62	66	Hamilton Academical FC	1481	NaN	NaN	NaN	...	NaN	NaN
13253	G. Miller	31	Scotland	62	62	Carlisle United	1535	NaN	NaN	NaN	...	NaN	NaN
13266	L. Bengtsson	20	Sweden	62	73	Hammarby IF	1549	NaN	NaN	NaN	...	NaN	NaN
13252	E. Binaku	22	Albania	62	70	Malmö FF	1587	NaN	NaN	NaN	...	NaN	NaN
13268	L. Garguła	37	Poland	62	62	Miedź Legnica	1542	NaN	NaN	NaN	...	NaN	NaN

10 rows × 48 columns

In [22]:

```
df_missing_players.describe()
```

Out[22]:

	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Crossing	Finishing	HeadingAccuracy	...	Marking	St
count	48.000000	48.0	48.000000	48.000000	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
mean	25.000000	62.0	66.833333	1562.229167	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
std	4.472136	0.0	5.272705	127.956981	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
min	17.000000	62.0	62.000000	1141.000000	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
25%	22.000000	62.0	62.000000	1506.750000	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
50%	25.000000	62.0	65.500000	1576.000000	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
75%	27.000000	62.0	70.000000	1664.250000	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	
max	37.000000	62.0	82.000000	1740.000000	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	

8 rows × 43 columns

We can see that quite a few columns which are related to players' skills got 48 missing values.

So there were 48 players that simply missing these values.

But we will reserve those players for Q1 and Q2 since there were no missing value in Value_M and Wage_K column.

For Q3, we will drop those player rows since there were just too many missing values here.

DataAnalytics

Ratio of total wages / total potential for clubs(which clubs are most economical?)

In [23]:

```
club_wages = df_players.groupby('Club').sum()
```

In [24]:

```
club_wages
```

Out[24]:

	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Crossing	Finishing	HeadingAccuracy	...	Marking	Sta
Club													
SSV Jahn Regensburg	744	1902	2010	44680	29.0	90.0	65.0	1368.0	1261.0	1605.0	...	1431.0	
1. FC Heidenheim 1846	672	1841	2014	43784	28.0	83.0	65.0	1385.0	1274.0	1397.0	...	1235.0	
1. FC Kaiserslautern	620	1648	1817	39631	26.0	82.0	58.0	1250.0	1082.0	1322.0	...	1100.0	
1. FC Köln	681	1982	2144	46807	37.0	85.0	68.0	1466.0	1291.0	1518.0	...	1474.0	
1. FC Magdeburg	642	1706	1829	39850	27.0	86.0	55.0	1209.0	1166.0	1320.0	...	1215.0	
1. FC Nürnberg	690	1996	2173	47089	31.0	88.0	74.0	1477.0	1321.0	1556.0	...	1380.0	
1. FC Union Berlin	707	1913	2054	45573	29.0	87.0	70.0	1414.0	1337.0	1516.0	...	1293.0	
1. FSV Mainz 05	758	2267	2473	53272	37.0	103.0	78.0	1677.0	1477.0	1739.0	...	1519.0	
AC Ajaccio	622	1496	1605	36844	24.0	69.0	55.0	1172.0	1062.0	1255.0	...	997.0	
AC Horsens	625	1516	1645	37407	25.0	71.0	50.0	1141.0	1111.0	1307.0	...	1180.0	
AD Alcorcón	780	1955	2047	46767	29.0	89.0	68.0	1522.0	1293.0	1532.0	...	1447.0	

Club	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Crossing	Finishing	HeadingAccuracy	...	Marking	Star
ADO Den Haag	694	1867	1994	45310	30.0	84.0	72.0	1487.0	1300.0	1477.0	...	1252.0	
AEK Athens	687	1966	2107	46560	29.0	86.0	70.0	1445.0	1277.0	1533.0	...	1508.0	
AFC Wimbledon	614	1572	1745	38214	26.0	73.0	52.0	1164.0	1041.0	1323.0	...	1117.0	
AIK	704	1757	1880	42401	28.0	79.0	60.0	1282.0	1169.0	1391.0	...	1354.0	
AJ Auxerre	680	1790	1957	42635	29.0	79.0	62.0	1393.0	1173.0	1472.0	...	1188.0	
AS Béziers	670	1613	1711	39302	25.0	72.0	51.0	1156.0	1037.0	1209.0	...	1070.0	
AS Monaco	761	2407	2671	56738	50.0	103.0	89.0	1823.0	1607.0	1859.0	...	1658.0	
AS Nancy Lorraine	748	1940	2114	46292	31.0	82.0	67.0	1478.0	1305.0	1618.0	...	1348.0	
AS Saint-Étienne	610	1701	1792	40723	37.0	70.0	61.0	1360.0	1200.0	1327.0	...	1130.0	
AZ Alkmaar	692	2100	2295	51332	33.0	92.0	82.0	1630.0	1532.0	1655.0	...	1485.0	
Aalborg BK	631	1675	1889	41740	28.0	81.0	61.0	1259.0	1228.0	1378.0	...	1225.0	
Aarhus GF	642	1658	1826	41629	27.0	79.0	61.0	1333.0	1200.0	1275.0	...	1231.0	
Aberdeen	652	1737	1938	42076	27.0	80.0	64.0	1325.0	1135.0	1397.0	...	1218.0	
Accrington Stanley	695	1713	1869	41633	28.0	77.0	57.0	1231.0	1131.0	1404.0	...	1279.0	
Adelaide United	609	1535	1690	37049	25.0	71.0	56.0	1131.0	1079.0	1115.0	...	927.0	
Ajax	669	2209	2406	54668	46.0	97.0	88.0	1832.0	1705.0	1781.0	...	1534.0	
Akhisar Belediyespor	761	1797	1828	44403	27.0	77.0	64.0	1451.0	1299.0	1437.0	...	1333.0	
Al Ahli	778	1971	2110	47723	34.0	95.0	71.0	1528.0	1355.0	1580.0	...	1276.0	
Al Batin	753	1795	1947	42216	30.0	89.0	66.0	1235.0	1190.0	1395.0	...	1132.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
Vitesse	685	1917	2066	46807	30.0	91.0	72.0	1516.0	1446.0	1425.0	...	1374.0	
Vitória	600	1408	1408	34271	20.0	62.0	48.0	1144.0	1059.0	1210.0	...	929.0	
Vitória Guimarães	737	2170	2307	50696	32.0	87.0	84.0	1642.0	1449.0	1653.0	...	1433.0	
Vitória de Setúbal	711	1929	2033	45970	29.0	82.0	68.0	1377.0	1292.0	1581.0	...	1296.0	
Vålerenga Fotball	603	1606	1810	40492	26.0	76.0	62.0	1226.0	1081.0	1339.0	...	1129.0	
Vélez Sarsfield	650	1827	2057	43774	28.0	78.0	66.0	1391.0	1302.0	1419.0	...	1297.0	
Waasland-Beveren	630	1812	1998	43390	28.0	90.0	63.0	1326.0	1169.0	1437.0	...	1212.0	
Walsall	643	1711	1873	43353	27.0	84.0	61.0	1307.0	1205.0	1420.0	...	1298.0	
Waterford FC	597	1430	1620	36153	25.0	63.0	56.0	1105.0	997.0	1142.0	...	1012.0	
Watford	739	2098	2238	49747	41.0	83.0	75.0	1653.0	1405.0	1697.0	...	1600.0	
Wellington Phoenix	533	1306	1400	32571	22.0	61.0	43.0	989.0	920.0	1063.0	...	989.0	
West Bromwich Albion	774	2086	2240	49785	38.0	88.0	75.0	1595.0	1456.0	1698.0	...	1501.0	
West Ham United	814	2339	2498	56496	57.0	95.0	92.0	1837.0	1607.0	1898.0	...	1650.0	
Western Sydney Wanderers	602	1546	1707	37772	26.0	75.0	53.0	1163.0	1114.0	1210.0	...	1109.0	
Wigan Athletic	736	1997	2166	47931	32.0	88.0	74.0	1501.0	1386.0	1573.0	...	1398.0	
Willem II	635	1828	2010	43719	30.0	83.0	67.0	1392.0	1197.0	1301.0	...	1244.0	
Wisła Kraków	637	1624	1799	39707	27.0	73.0	57.0	1278.0	1157.0	1271.0	...	1027.0	
Wisła Płock	627	1587	1715	38457	26.0	67.0	56.0	1179.0	1037.0	1200.0	...	1000.0	
Wolfsberger AC	640	1600	1754	38691	26.0	80.0	54.0	1159.0	998.0	1277.0	...	1193.0	
Wolverhampton Wanderers	754	2271	2560	54427	44.0	102.0	86.0	1725.0	1482.0	1653.0	...	1590.0	
Wycombe Wanderers	685	1593	1687	40215	25.0	73.0	56.0	1267.0	1159.0	1284.0	...	1159.0	

	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Crossing	Finishing	HeadingAccuracy	...	Marking	Sta
Yeni Malatyaspor	776	1967	2080	47367	30.0	82.0	70.0	1478.0	1353.0	1628.0	...	1366.0	
Yeovil Town Club	679	1690	1856	41200	28.0	75.0	55.0	1204.0	1184.0	1357.0	...	1111.0	
Yokohama F. Marinos	751	1846	1989	44038	31.0	87.0	66.0	1391.0	1233.0	1365.0	...	1313.0	
Zagłębie Lubin	694	1691	1779	41327	27.0	79.0	61.0	1294.0	1175.0	1313.0	...	1075.0	
Zagłębie Sosnowiec	656	1519	1616	37371	25.0	66.0	51.0	1161.0	1007.0	1152.0	...	1219.0	
Çaykur Rizespor	763	2007	2150	48732	30.0	82.0	71.0	1512.0	1390.0	1596.0	...	1382.0	
Örebro SK	649	1633	1796	39274	27.0	68.0	57.0	1180.0	1042.0	1192.0	...	1170.0	
Östersunds FK	525	1398	1515	34389	22.0	62.0	50.0	1080.0	917.0	943.0	...	1035.0	
Śląsk Wrocław	649	1555	1639	38457	24.0	69.0	53.0	1058.0	1054.0	1289.0	...	1047.0	

651 rows × 43 columns

In [25]:

```
club_player_count = df_players.groupby("Club").count()
```

In [26]:

```
club_player_count
```

Out[26]:

	Name	Age	Nationality	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Work Rate	...	Marking	StandingTackle
Club													
SSV Jahn Regensburg	29	29	29	29	29	29	29	29	29	29	...	29	29
1. FC Heidenheim 1846	28	28	28	28	28	28	28	28	28	28	...	28	28
1. FC Kaiserslautern	26	26	26	26	26	26	26	26	26	26	...	26	26
1. FC Köln	28	28	28	28	28	28	28	28	28	28	...	28	28
1. FC Magdeburg	26	26	26	26	26	26	26	26	26	26	...	26	26
1. FC Nürnberg	29	29	29	29	29	29	29	29	29	29	...	29	29
1. FC Union Berlin	28	28	28	28	28	28	28	28	28	28	...	28	28
1. FSV Mainz 05	32	32	32	32	32	32	32	32	32	32	...	32	32
AC Ajaccio	23	23	23	23	23	23	23	23	23	23	...	23	23
AC Horsens	25	25	25	25	25	25	25	25	25	25	...	25	25
AD Alcorcón	29	29	29	29	29	29	29	29	29	29	...	29	29
ADO Den Haag	28	28	28	28	28	28	28	28	28	28	...	28	28
AEK Athens	28	28	28	28	28	28	28	28	28	28	...	28	28
AFC Wimbledon	26	26	26	26	26	26	26	26	26	26	...	26	26
AIK	27	27	27	27	27	27	27	27	27	27	...	27	27
AJ Auxerre	27	27	27	27	27	27	27	27	27	27	...	27	27
AS Béziers	26	26	26	26	26	26	25	25	25	25	...	25	25
AS Monaco	33	33	33	33	33	33	33	33	33	33	...	33	33
AS Nancy Lorraine	30	30	30	30	30	30	30	30	30	30	...	30	30
AS Saint-Étienne	24	24	24	24	24	24	24	24	24	24	...	24	24
AZ Alkmaar	30	30	30	30	30	30	30	30	30	30	...	30	30
Aalborg BK	27	27	27	27	27	27	27	27	27	27	...	27	27

	Name	Age	Nationality	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Work Rate	...	Marking	StandingTackle
Aarhus GF	27	27	27	27	27	27	27	27	27	27	...	27	27
Aberdeen Club	27	27	27	27	27	27	27	27	27	27	...	27	27
Accrington Stanley	28	28	28	28	28	28	28	28	28	28	...	28	28
Adelaide United	25	25	25	25	25	25	25	25	25	25	...	25	25
Ajax	30	30	30	30	30	30	30	30	30	30	...	30	30
Akhisar Belediyespor	26	26	26	26	26	26	26	26	26	26	...	26	26
Al Ahli	30	30	30	30	30	30	30	30	30	30	...	30	30
Al Batin	30	30	30	30	30	30	30	30	30	30	...	30	30
...	...	...	...	...	...	...	...	...	...	...	...	...	...
Vitesse	28	28	28	28	28	28	28	28	28	28	...	28	28
Vitória	20	20	20	20	20	20	20	20	20	20	...	20	20
Vitória Guimarães	30	30	30	30	30	30	30	30	30	30	...	30	30
Vitória de Setúbal	28	28	28	28	28	28	28	28	28	28	...	28	28
Vålerenga Fotball	26	26	26	26	26	26	26	26	26	26	...	26	26
Vélez Sarsfield	28	28	28	28	28	28	28	28	28	28	...	28	28
Waasland-Beveren	28	28	28	28	28	28	28	28	28	28	...	28	28
Walsall	27	27	27	27	27	27	27	27	27	27	...	27	27
Waterford FC	25	25	25	25	25	25	25	25	25	25	...	25	25
Watford	29	29	29	29	29	29	29	29	29	29	...	29	29
Wellington Phoenix	21	21	21	21	21	21	21	21	21	21	...	21	21
West Bromwich Albion	30	30	30	30	30	30	30	30	30	30	...	30	30
West Ham United	32	32	32	32	32	32	32	32	32	32	...	32	32
Western Sydney Wanderers	25	25	25	25	25	25	25	25	25	25	...	25	25
Wigan Athletic	30	30	30	30	30	30	30	30	30	30	...	30	30
Willem II	28	28	28	28	28	28	28	28	28	28	...	28	28
Wisła Kraków	26	26	26	26	26	26	26	26	26	26	...	26	26
Wisła Płock	26	26	26	26	26	26	26	26	26	26	...	26	26
Wolfsberger AC	26	26	26	26	26	26	26	26	26	26	...	26	26
Wolverhampton Wanderers	33	33	33	33	33	33	33	33	33	33	...	33	33
Wycombe Wanderers	25	25	25	25	25	25	25	25	25	25	...	25	25
Yeni Malatyaspor	30	30	30	30	30	30	30	30	30	30	...	30	30
Yeovil Town	28	28	28	28	28	28	28	28	28	28	...	28	28
Yokohama F. Marinos	30	30	30	30	30	30	30	30	30	30	...	30	30
Zagłębie Lubin	27	27	27	27	27	27	27	27	27	27	...	27	27
Zagłębie Sosnowiec	25	25	25	25	25	25	25	25	25	25	...	25	25
Çaykur Rizespor	30	30	30	30	30	30	30	30	30	30	...	30	30
Örebro SK	27	27	27	27	27	27	27	27	27	27	...	27	27
Östersunds FK	22	22	22	22	22	22	22	22	22	22	...	22	22
Śląsk Wrocław	25	25	25	25	25	25	24	24	24	24	...	24	24

651 rows × 47 columns

In [27]:

```
#Number of clubs and avarage number of players in each club
print('Total Number of Club is {}'.format(club_player_count.shape[0]))
print('Avg Number of players in each club is
{}'.format(round(club_player_count['Name'].mean(),3)))
print('Total Average Wage(k) and Potential Ratio is {}'.format(round(club_wages['Wages_K'].sum() /
club_wages['Potential'].sum(),2)))
```

```
Total Number of Club is 651
Avg Number of players in each club is 27.598
Total Average Wage(k) and Potential Ratio is 0.14
```

In [28]:

```
# Finding this details for all clubs
club_wages['Wage/Potential'] = club_wages['Wages_K'] / club_wages['Potential']
club_wages['Player_Number'] = club_player_count['Name']
club_wages['Player Avg Age']= club_wages['Age'] / club_wages['Player_Number']
```

In [29]:

```
club_wages.sort_values('Wage/Potential',ascending=False, inplace=True)
club_wages.head()
```

Out[29]:

Club	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Crossing	Finishing	HeadingAccuracy	...	GKDiving	GKHa
Real Madrid	793	2582	2793	60025	69.0	106.0	94.0	1934.0	1750.0	1887.0	...	627.0	
FC Barcelona	787	2575	2815	60791	74.0	108.0	94.0	1974.0	1805.0	1850.0	...	599.0	
Juventus	679	2057	2138	47610	63.0	80.0	72.0	1517.0	1282.0	1583.0	...	419.0	
Manchester City	789	2532	2769	60617	69.0	104.0	92.0	1970.0	1726.0	1852.0	...	592.0	
Manchester United	817	2549	2728	62117	69.0	106.0	100.0	2054.0	1862.0	2056.0	...	547.0	

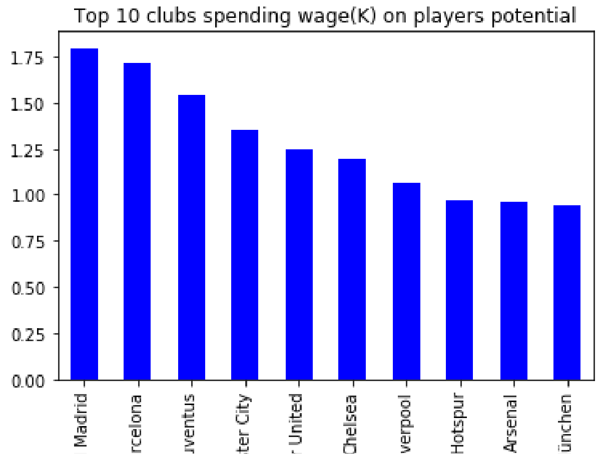
5 rows × 46 columns

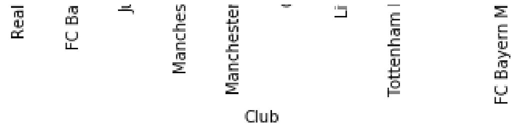
In [30]:

```
club_wages['Wage/Potential'].head(10).plot(kind='bar', color='Blue')
plt.title('Top 10 clubs spending wage(K) on players potential')
```

Out[30]:

```
Text(0.5, 1.0, 'Top 10 clubs spending wage(K) on players potential')
```



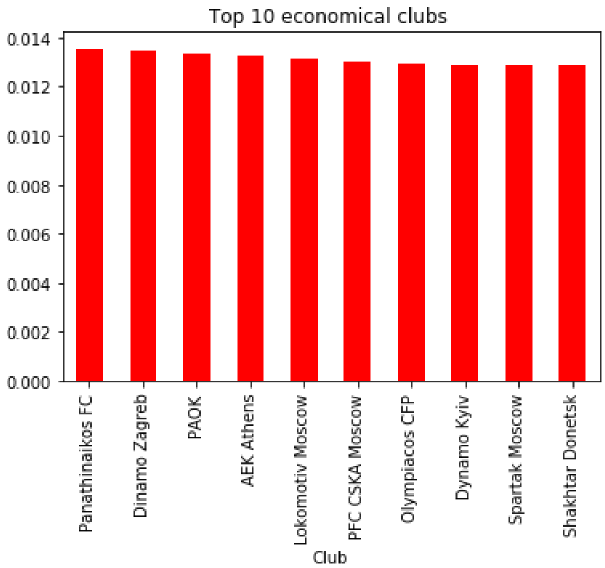


In [31]:

```
club_wages['Wage/Potential'].tail(10).plot(kind='bar', color='red')
plt.title('Top 10 economical clubs ')
```

Out[31]:

```
Text(0.5, 1.0, 'Top 10 economical clubs ')
```



From the result and plot, it's obvious that the 'Giant' clubs including Real Madrid, Bacelona, and clubs from EPL are willing to spend much more wage for high potential players than average clubs. This is how they stay competitive in leagues.

But surprisingly, the economical clubs are not clubs from nowhere that we never heard of. Some of them are even quite famous like AEK Athens, Dynamo Kyiv. This suggests that those clubs' players are potential but underpayed. It maybe a good approach for 'Giant' clubs to import more econimical players from them to reduce their overall wage spent.

Age Distribution and how its related to Players Overall Rating

In [32]:

```
age_count = df_players ['Age'].value_counts()
age_count.sort_index(ascending =True,inplace=True)
```

In [33]:

```
age_count.head()
```

Out[33]:

```
16      42
17     289
18     732
19    1024
20    1240
Name: Age, dtype: int64
```

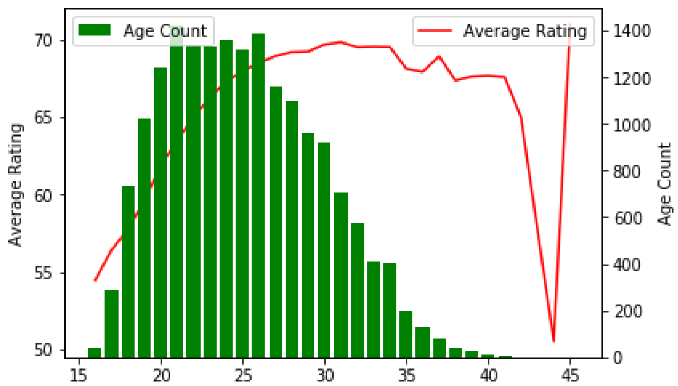
In [34]:

```
age_count_list = age_count.values.tolist()
age_mean = df_players.groupby('Age').mean()
age_overall_rating_list = age_mean['Overall'].values.tolist()
```

In [35]:

```
ages = age_count.index.values.tolist()
fig = plt.figure()
ax1 = fig.add_subplot(111)
ax1.plot(ages,age_overall_rating_list, color = 'red', label='Average Rating')
ax1.legend(loc=1)
ax1.set_ylabel('Average Rating')

ax2 = ax1.twinx()
plt.bar(ages, age_count_list, label='Age Count',color='green')
ax2.legend(loc=2)
ax2.set_ylabel('Age Count')
plt.show()
```



From above plot, we can see that most players are between 20-26 years old. And players' number start to decrease after 26 years old and speed up after 30. Reason behind this could be that many young player didn't get enough opportunities to prove themselves and give up their dream as a football player.

When a football player reaches their late 20s, they have gain enough experience and reaches peak of their rating. The golden era of a football player starts here and ends when his age reaches 35. At this age, his physical body condition drops quickly so as average rating.

There are also quite a few numbers of players with age over 37, 38 years old. This is quite a surprise especially their rating still can remain quite high.

Data Analytics with ML

In [36]:

```
columns_to_drop = ['Name', 'Nationality', 'Club']
df_players.drop(columns_to_drop, axis=1, inplace=True)
```

In [37]:

```
df_players.dropna(axis=0,how ='any',inplace=True)
```

In [38]:

```
df_players
```

Out[38]:

	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Work Rate	Position	Crossing	...	Marking	StandingTackle	Sliding
0	31	94	94	2202	5.0	4.0	4.0	Medium/ Medium	RF	84.0	...	33.0	28.0	
1	33	94	94	2228	5.0	4.0	5.0	High/ Low	ST	84.0	...	28.0	31.0	
2	26	92	93	2143	5.0	5.0	5.0	High/	LW	79.0		27.0	24.0	

	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Work Rate	Position	Crossing	...	Marking	StandingTackle	Sliding
3	27	91	93	1471	4.0	3.0	1.0	Medium/ Medium	GK	17.0	...	15.0	21.0	
4	27	91	92	2281	4.0	5.0	4.0	High/ High	RCM	93.0	...	68.0	58.0	
5	27	91	91	2142	4.0	4.0	4.0	High/ Medium	LF	81.0	...	34.0	27.0	
6	32	91	91	2280	4.0	4.0	4.0	High/ High	RCM	86.0	...	60.0	76.0	
7	31	91	91	2346	5.0	4.0	3.0	High/ Medium	RS	77.0	...	62.0	45.0	
8	32	91	91	2201	4.0	3.0	3.0	High/ Medium	RCB	66.0	...	87.0	92.0	
9	25	90	93	1331	3.0	3.0	1.0	Medium/ Medium	GK	13.0	...	27.0	12.0	
10	29	90	90	2152	4.0	4.0	4.0	High/ Medium	ST	62.0	...	34.0	42.0	
11	28	90	90	2190	4.0	5.0	3.0	Medium/ Medium	LCM	88.0	...	72.0	79.0	
12	32	90	90	1946	3.0	3.0	2.0	Medium/ High	CB	55.0	...	90.0	89.0	
13	32	90	90	2115	4.0	2.0	4.0	High/ Medium	LCM	84.0	...	59.0	53.0	
14	27	89	90	2189	3.0	3.0	2.0	Medium/ High	LDM	68.0	...	90.0	91.0	
15	24	89	94	2092	3.0	3.0	4.0	High/ Medium	LF	82.0	...	23.0	20.0	
16	24	89	91	2165	3.0	4.0	3.0	High/ High	ST	75.0	...	56.0	36.0	
17	27	89	90	2246	4.0	3.0	4.0	High/ High	CAM	82.0	...	59.0	47.0	
18	26	89	92	1328	3.0	4.0	1.0	Medium/ Medium	GK	15.0	...	25.0	13.0	
19	26	89	90	1311	4.0	2.0	1.0	Medium/ Medium	GK	14.0	...	20.0	18.0	
20	29	89	89	2065	4.0	3.0	3.0	Medium/ Medium	CDM	62.0	...	90.0	86.0	
21	31	89	89	2161	4.0	4.0	3.0	High/ High	LS	70.0	...	52.0	45.0	
22	32	89	89	1473	5.0	4.0	1.0	Medium/ Medium	GK	15.0	...	17.0	10.0	
23	30	89	89	2107	4.0	4.0	4.0	High/ Medium	ST	70.0	...	30.0	20.0	
24	33	89	89	1841	4.0	3.0	2.0	Medium/ High	LCB	58.0	...	93.0	93.0	
25	19	88	95	2118	3.0	4.0	5.0	High/ Medium	RM	77.0	...	34.0	34.0	
26	26	88	89	2146	3.0	3.0	4.0	High/ Medium	RM	78.0	...	38.0	43.0	
27	26	88	90	2170	3.0	3.0	2.0	Medium/ High	CDM	52.0	...	88.0	90.0	
28	26	88	89	2171	4.0	3.0	4.0	Medium/ Medium	LAM	90.0	...	52.0	41.0	
29	27	88	88	2017	3.0	3.0	4.0	High/ Medium	LW	86.0	...	51.0	24.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
18177	18	48	69	1178	1.0	3.0	2.0	Medium/ Medium	ST	32.0	...	18.0	16.0	
18178	18	48	65	738	1.0	2.0	1.0	Medium/ Medium	GK	10.0	...	16.0	11.0	
18179	17	48	64	1166	1.0	3.0	2.0	Medium/ Medium	CB	25.0	...	42.0	51.0	
18180	22	48	58	987	1.0	2.0	1.0	Medium/ Medium	GK	19.0	...	12.0	15.0	
18181	17	48	66	1296	1.0	2.0	2.0	Medium/	RB	45.0	...	43.0	49.0	

	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Work Rate	Position	Crossing	...	Marking	StandingTackle	Sliding
18182	18	48	65	1311	1.0	3.0	2.0	Low/ Medium	CDM	35.0	...	44.0	42.0	
18183	44	48	48	774	1.0	2.0	1.0	Medium/ Medium	GK	11.0	...	15.0	15.0	
18184	18	48	55	1368	1.0	3.0	2.0	Medium/ Medium	CM	33.0	...	47.0	49.0	
18185	19	48	59	1315	1.0	3.0	2.0	Medium/ Medium	LCM	37.0	...	39.0	39.0	
18186	20	47	64	1389	1.0	3.0	2.0	Medium/ Medium	CM	35.0	...	53.0	41.0	
18187	19	47	59	1366	1.0	3.0	2.0	High/ Medium	RB	39.0	...	40.0	42.0	
18188	17	47	62	1297	1.0	3.0	2.0	Medium/ Medium	CM	41.0	...	33.0	38.0	
18189	18	47	61	1290	1.0	3.0	2.0	Medium/ Medium	ST	37.0	...	28.0	15.0	
18190	18	47	67	1285	1.0	3.0	2.0	Medium/ Medium	CM	32.0	...	35.0	44.0	
18191	18	47	65	1250	1.0	3.0	2.0	Medium/ High	LB	47.0	...	45.0	42.0	
18192	18	47	64	1325	1.0	3.0	2.0	Low/ Medium	CDM	39.0	...	41.0	41.0	
18193	18	47	64	1191	1.0	2.0	2.0	Medium/ Medium	RB	36.0	...	41.0	48.0	
18194	18	47	65	731	1.0	3.0	1.0	Medium/ Medium	GK	10.0	...	6.0	10.0	
18195	18	47	67	1325	1.0	3.0	2.0	Medium/ Medium	CM	35.0	...	44.0	37.0	
18196	19	47	61	1333	1.0	3.0	2.0	Medium/ Medium	CM	31.0	...	41.0	44.0	
18197	18	47	61	1362	1.0	3.0	2.0	Medium/ Medium	CM	44.0	...	41.0	47.0	
18198	18	47	70	792	1.0	2.0	1.0	Medium/ Medium	GK	14.0	...	15.0	11.0	
18199	18	47	69	1303	1.0	3.0	2.0	Medium/ High	CM	31.0	...	48.0	49.0	
18200	18	47	62	1203	1.0	2.0	2.0	Medium/ Medium	ST	28.0	...	15.0	17.0	
18201	18	47	68	1098	1.0	3.0	2.0	Medium/ Medium	RB	22.0	...	44.0	47.0	
18202	19	47	65	1307	1.0	2.0	2.0	Medium/ Medium	CM	34.0	...	40.0	48.0	
18203	19	47	63	1098	1.0	2.0	2.0	Medium/ Medium	ST	23.0	...	22.0	15.0	
18204	16	47	67	1189	1.0	3.0	2.0	Medium/ Medium	ST	25.0	...	32.0	13.0	
18205	17	47	66	1228	1.0	3.0	2.0	Medium/ Medium	RW	44.0	...	20.0	25.0	
18206	16	46	66	1321	1.0	3.0	2.0	Medium/ Medium	CM	41.0	...	40.0	43.0	

18147 rows × 45 columns

In [39]:

```
df_players['Work Rate Attack'] = df_players['Work Rate'].map(lambda x: x.split('/')[0])
df_players['Work Rate Defence'] = df_players['Work Rate'].map(lambda x: x.split('/')[1])
df_players.drop('Work Rate', axis=1, inplace=True)
```

In [40]:

```
df_players.head()
```

Out[40]:

	Age	Overall	Potential	Special	International Reputation	Weak Foot	Skill Moves	Position	Crossing	Finishing	...	SlidingTackle	GKDiving	GKHandlin
0	31	94	94	2202	5.0	4.0	4.0	RF	84.0	95.0	...	26.0	6.0	11.
1	33	94	94	2228	5.0	4.0	5.0	ST	84.0	94.0	...	23.0	7.0	11.
2	26	92	93	2143	5.0	5.0	5.0	LW	79.0	87.0	...	33.0	9.0	9.
3	27	91	93	1471	4.0	3.0	1.0	GK	17.0	13.0	...	13.0	90.0	85.
4	27	91	92	2281	4.0	5.0	4.0	RCM	93.0	82.0	...	51.0	15.0	13.

5 rows × 46 columns

In [41]:

```
# One Hot Encoding for Position, Work Rate Attack, Work Rate Defence
one_hot_columns = ['Position', 'Work Rate Attack', 'Work Rate Defence']
df_players = pd.get_dummies(df_players, columns=one_hot_columns, prefix = one_hot_columns).
```

In [42]:

```
print(df_players.head())

df_players.shape
```

```
   Age  Overall  Potential  Special  International Reputation  Weak Foot  \
0   31       94         94     2202                       5.0        4.0
1   33       94         94     2228                       5.0        4.0
2   26       92         93     2143                       5.0        5.0
3   27       91         93     1471                       4.0        3.0
4   27       91         92     2281                       4.0        5.0

   Skill Moves  Crossing  Finishing  HeadingAccuracy  \
0          4.0      84.0       95.0             70.0
1          5.0      84.0       94.0             89.0
2          5.0      79.0       87.0             62.0
3          1.0      17.0       13.0             21.0
4          4.0      93.0       82.0             55.0

             ...                Position_RS  Position_RW  Position_RWB  \
0            ...                          0            0             0
1            ...                          0            0             0
2            ...                          0            0             0
3            ...                          0            0             0
4            ...                          0            0             0

   Position_ST  Work Rate Attack_High  Work Rate Attack_Low  \
0            0                      0                    0
1            1                      1                    0
2            0                      1                    0
3            0                      0                    0
4            0                      1                    0

   Work Rate Attack_Medium  Work Rate Defence_ High  Work Rate Defence_ Low  \
0                        1                        0                       0
1                        0                        0                       1
2                        0                        0                       0
3                        1                        0                       0
4                        0                        1                       0

   Work Rate Defence_ Medium
0                          1
1                          0
2                          1
3                          1
4                          0

[5 rows x 76 columns]
```

Out[42]:

```
(18147, 76)
```

Train Model and Measure Performance

In [48]:

```
from sklearn.tree import DecisionTreeClassifier
dtree = DecisionTreeClassifier()
```

In [49]:

```
y = df_players['Potential']
X = df_players.drop(['Value_M', 'Wages_K', 'Potential', 'Overall'], axis=1)
```

In [50]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.3, random_state=42)
```

In [51]:

```
dtree.fit(X_train,y_train)
```

Out[51]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

Predictions and evaluation using DecisionTree

In [52]:

```
predictions = dtree.predict(X_test)
from sklearn.metrics import classification_report,confusion_matrix
```

In [53]:

```
print(classification_report(y_test,predictions))
```

```
             precision    recall  f1-score   support

         48       0.00      0.00      0.00         1
         50       0.00      0.00      0.00         1
         51       0.00      0.00      0.00         1
         52       0.50      0.17      0.25         6
         53       0.00      0.00      0.00         1
         54       0.50      0.50      0.50         2
         55       0.00      0.00      0.00         6
         56       0.00      0.00      0.00         8
         57       0.05      0.07      0.06        14
         58       0.07      0.10      0.08        21
         59       0.18      0.14      0.16        51
         60       0.04      0.04      0.04        48
         61       0.06      0.06      0.06        72
         62       0.18      0.14      0.16       119
         63       0.10      0.12      0.11       143
         64       0.12      0.13      0.12       204
         65       0.13      0.15      0.14       208
         66       0.18      0.15      0.16       321
         67       0.14      0.13      0.14       286
         68       0.12      0.14      0.13       328
         69       0.15      0.14      0.15       353
         70       0.12      0.12      0.12       357
         71       0.13      0.11      0.12       363
         72       0.10      0.12      0.11       302
         73       0.12      0.12      0.12       317
```

```
          73       0.12      0.12      0.12       317
          74       0.16      0.15      0.15       316
          75       0.12      0.12      0.12       283
          76       0.16      0.17      0.16       224
          77       0.14      0.13      0.14       233
          78       0.11      0.10      0.10       168
          79       0.12      0.11      0.12       149
          80       0.08      0.11      0.09       113
          81       0.14      0.12      0.13       111
          82       0.15      0.19      0.17        85
          83       0.10      0.11      0.11        61
          84       0.05      0.05      0.05        43
          85       0.07      0.08      0.08        36
          86       0.00      0.00      0.00        19
          87       0.25      0.07      0.11        27
          88       0.09      0.06      0.07        17
          89       0.00      0.00      0.00        11
          90       0.00      0.00      0.00         5
          91       0.00      0.00      0.00         5
          92       0.00      0.00      0.00         3
          93       0.00      0.00      0.00         1
          94       0.00      0.00      0.00         2

   micro avg       0.13      0.13      0.13      5445
   macro avg       0.10      0.09      0.09      5445
weighted avg       0.13      0.13      0.13      5445
```

```
C:\Anaconda3\lib\site-packages\sklearn\metrics\classification.py:1143: UndefinedMetricWarning: Pre
cision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
C:\Anaconda3\lib\site-packages\sklearn\metrics\classification.py:1143: UndefinedMetricWarning: Pre
cision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
C:\Anaconda3\lib\site-packages\sklearn\metrics\classification.py:1143: UndefinedMetricWarning: Pre
cision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples.
  'precision', 'predicted', average, warn_for)
```

In [54]:

```
print(confusion_matrix(y_test,predictions))
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

So we can't use decision tree in predictions we have to use Random Forrest

In [42]:

```
y = df_players['Potential']
X = df_players.drop(['Value_M', 'Wages_K', 'Potential', 'Overall'], axis=1)
```

In [43]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.3, random_state=42)
```

In [44]:

```
from sklearn.metrics import r2_score, mean_squared_error, median_absolute_error
from sklearn.ensemble import RandomForestRegressor
```

In [45]:

```
ForestRegressor = RandomForestRegressor(n_estimators=500)
ForestRegressor.fit(X_train, y_train)
y_test_preds = ForestRegressor.predict(X_test)
print(r2_score(y_test, y_test_preds))
print(mean_squared_error(y_test, y_test_preds))
```

```
0.8714064884235864
4.914184478971534
```

In [46]:

```
coefs_df = pd.DataFrame()

coefs_df['Features'] = X_train.columns
coefs_df['Coefs'] = ForestRegressor.feature_importances_
coefs_df.sort_values('Coefs', ascending=False).head(10)
```

Out[46]:

	Features	Coefs
14	BallControl	0.262929
18	Reactions	0.201994
0	Age	0.179893
32	StandingTackle	0.066983
38	GKReflexes	0.025664
34	GKDiving	0.023093
1	Special	0.019232
7	HeadingAccuracy	0.016610
26	Interceptions	0.015520
31	Marking	0.015283

Ball control, reactions, and age are the main three features that decides a player's potential. This is same to our perception.

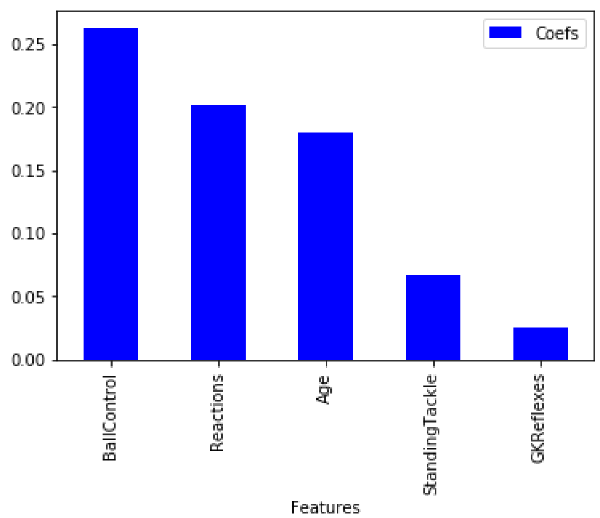
Young players with excellent ball control and fast reactions tends to give us an outstanding performance in football match.

In [47]:

```
coefs_df.set_index('Features', inplace=True)
coefs_df.sort_values('Coefs', ascending=False).head(5).plot(kind='bar', color='blue')
```

Out[47]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d03987d358>
```



In [ ]:

```

```