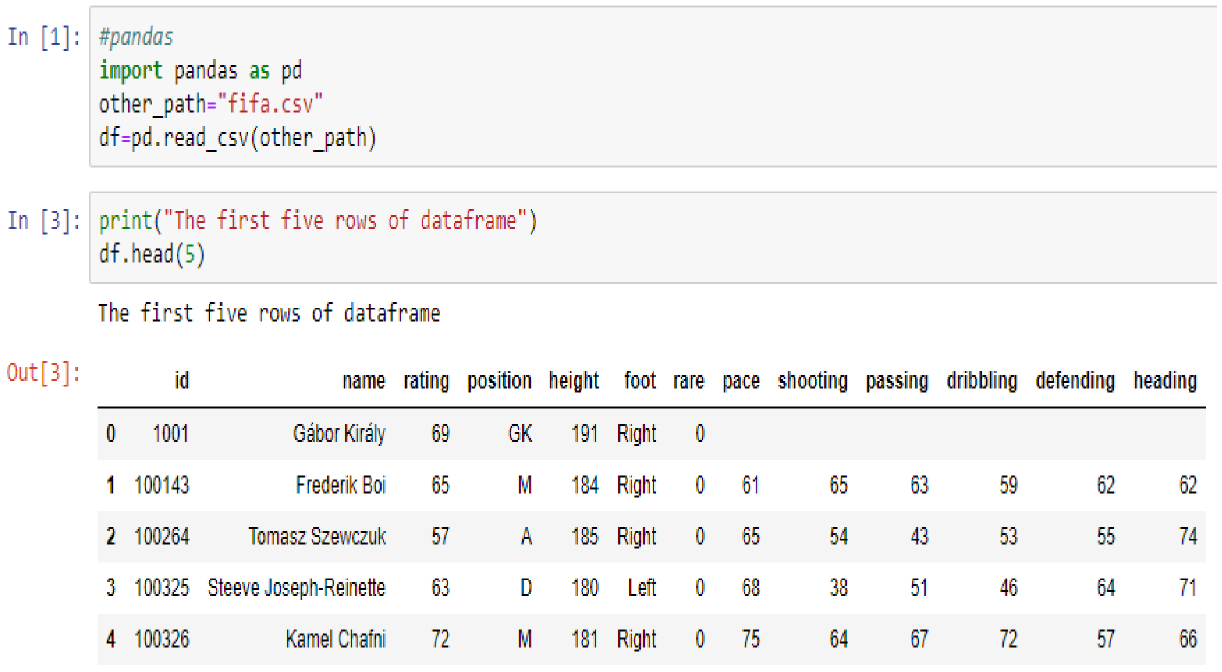
Roll no. 1714008

Pandas exploration using the following dataset.

Head, tail, describe, info, identify missing values and replace, group by and sort.

Head: pandas.DataFrame.head. This function returns the first n rows for the object based on position. It is useful for quickly testing if your object has the right type of data in it.



```
In [1]: #pandas
        import pandas as pd
        other_path="fifa.csv"
        df=pd.read_csv(other_path)
```

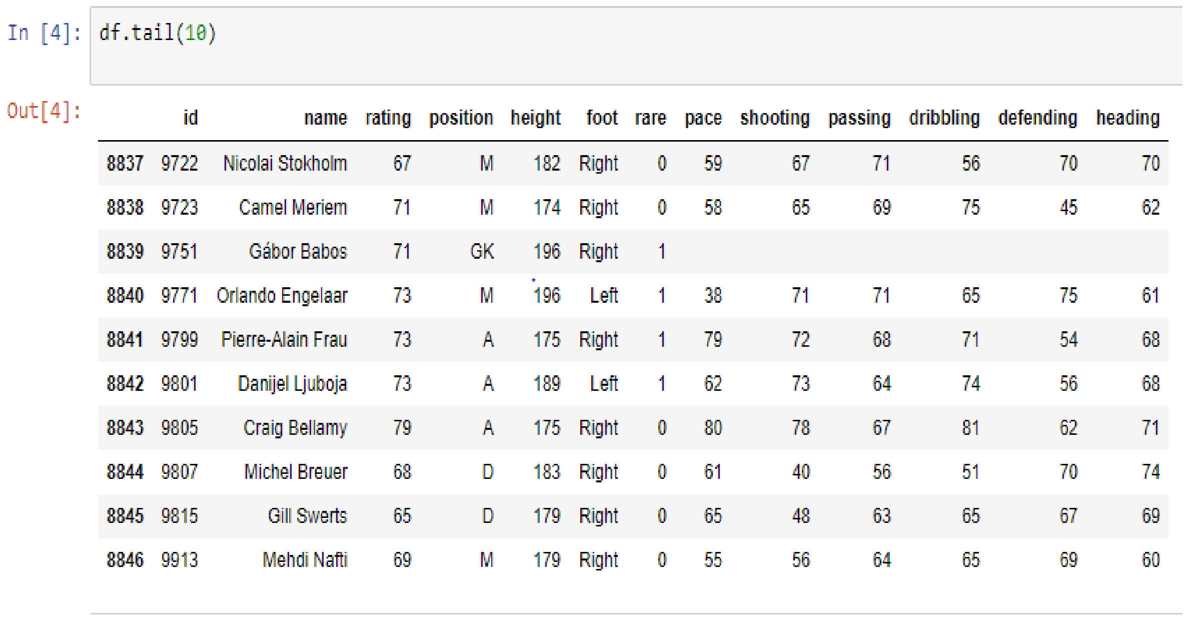
```
In [3]: print("The first five rows of dataframe")
        df.head(5)
```

```
The first five rows of dataframe
```

Out[3]:

	id	name	rating	position	height	foot	rare	pace	shooting	passing	dribbling	defending	heading
0	1001	Gábor Király	69	GK	191	Right	0						
1	100143	Frederik Boi	65	M	184	Right	0	61	65	63	59	62	62
2	100264	Tomasz Szewczuk	57	A	185	Right	0	65	54	43	53	55	74
3	100325	Steeve Joseph-Reinette	63	D	180	Left	0	68	38	51	46	64	71
4	100326	Kamel Chafni	72	M	181	Right	0	75	64	67	72	57	66

Tail : pandas.DataFrame.tail. This function returns last n rows from the object based on position. It is useful for quickly verifying data, for example, after sorting or appending rows.

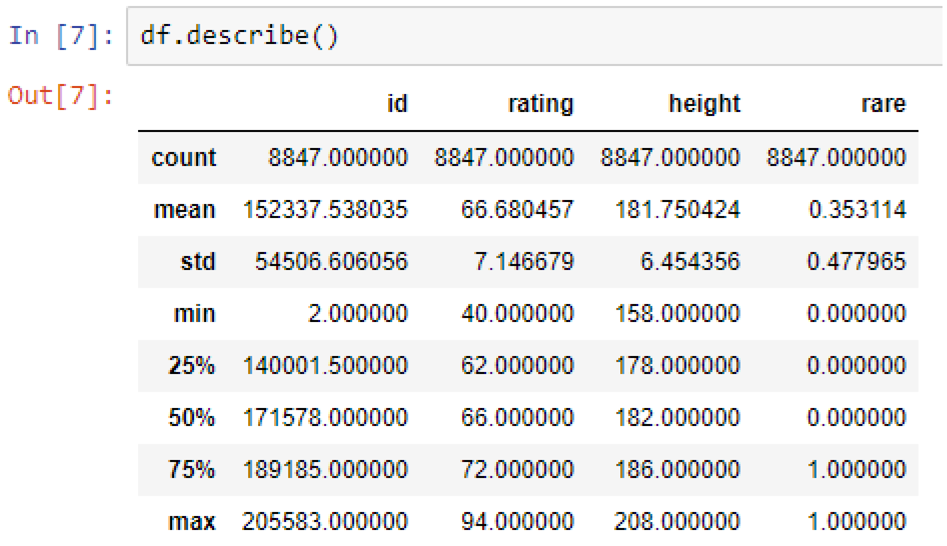


In [4]:
```
df.tail(10)
```

Out[4]:

	id	name	rating	position	height	foot	rare	pace	shooting	passing	dribbling	defending	heading
8837	9722	Nicolai Stokholm	67	M	182	Right	0	59	67	71	56	70	70
8838	9723	Camel Meriem	71	M	174	Right	0	58	65	69	75	45	62
8839	9751	Gábor Babos	71	GK	196	Right	1						
8840	9771	Orlando Engelaar	73	M	196	Left	1	38	71	71	65	75	61
8841	9799	Pierre-Alain Frau	73	A	175	Right	1	79	72	68	71	54	68
8842	9801	Danijel Ljuboja	73	A	189	Left	1	62	73	64	74	56	68
8843	9805	Craig Bellamy	79	A	175	Right	0	80	78	67	81	62	71
8844	9807	Michel Breuer	68	D	183	Right	0	61	40	56	51	70	74
8845	9815	Gill Swerts	65	D	179	Right	0	65	48	63	65	67	69
8846	9913	Mehdi Nafti	69	M	179	Right	0	55	56	64	65	69	60

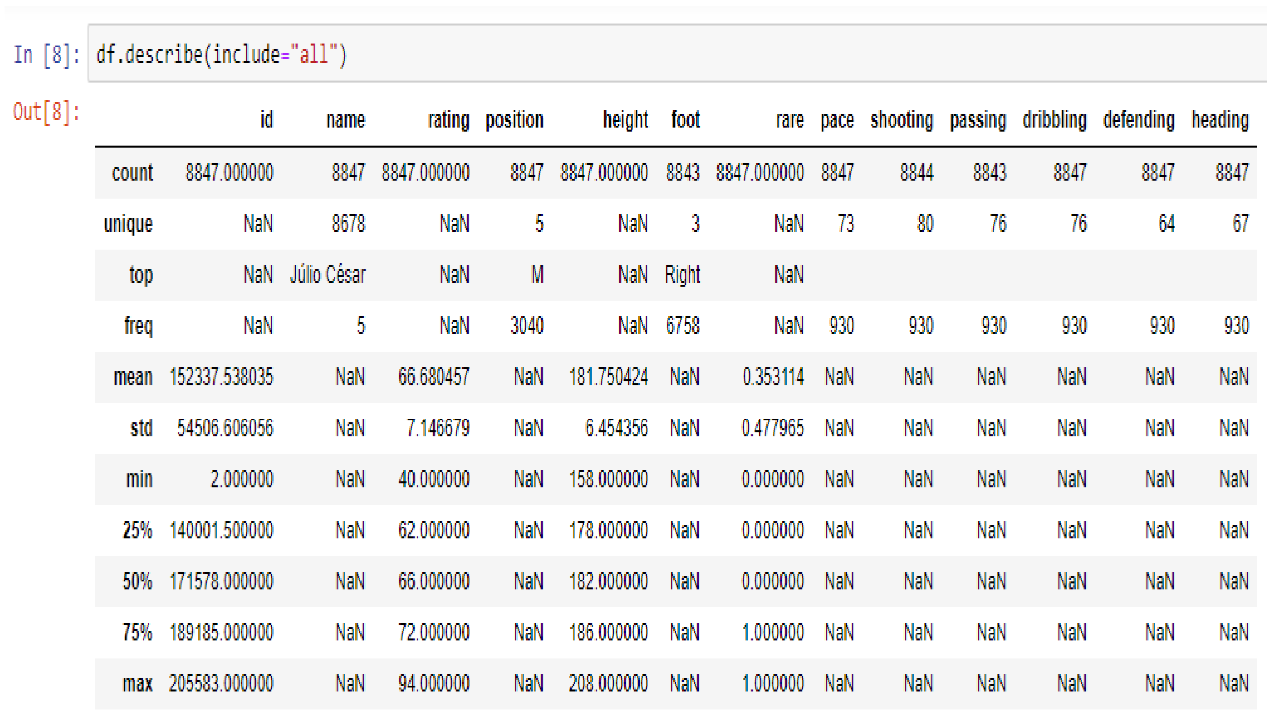
Describe : Pandas **`describe()`** is used to view some basic statistical details like percentile, mean, std etc. of a data frame or a series of numeric values.



In [7]:
```
df.describe()
```

Out[7]:

	id	rating	height	rare
count	8847.000000	8847.000000	8847.000000	8847.000000
mean	152337.538035	66.680457	181.750424	0.353114
std	54506.606056	7.146679	6.454356	0.477965
min	2.000000	40.000000	158.000000	0.000000
25%	140001.500000	62.000000	178.000000	0.000000
50%	171578.000000	66.000000	182.000000	0.000000
75%	189185.000000	72.000000	186.000000	1.000000
max	205583.000000	94.000000	208.000000	1.000000

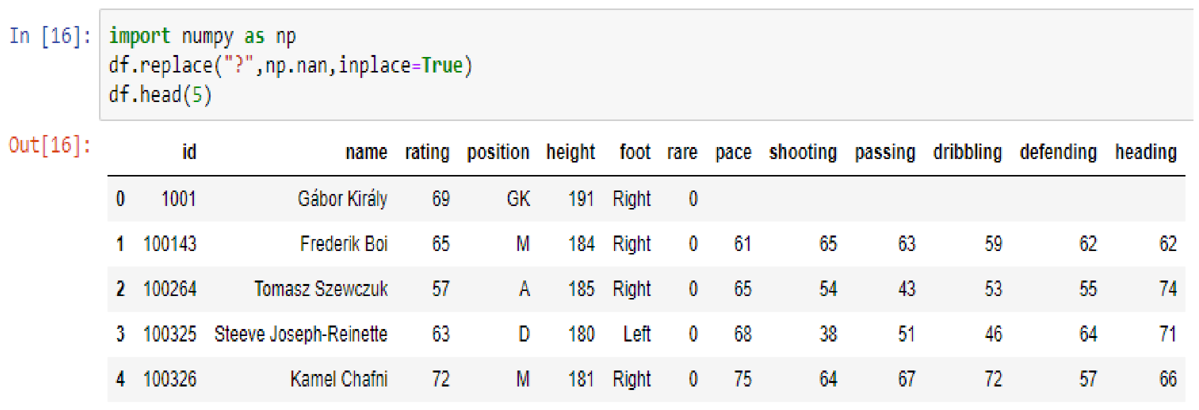


```
In [8]: df.describe(include="all")
```

Out[8]:

	id	name	rating	position	height	foot	rare	pace	shooting	passing	dribbling	defending	heading
count	8847.000000	8847	8847.000000	8847	8847.000000	8843	8847.000000	8847	8844	8843	8847	8847	8847
unique	NaN	8678	NaN	5	NaN	3	NaN	73	80	76	76	64	67
top	NaN	Júlio César	NaN	M	NaN	Right	NaN						
freq	NaN	5	NaN	3040	NaN	6758	NaN	930	930	930	930	930	930
mean	152337.538035	NaN	66.680457	NaN	181.750424	NaN	0.353114	NaN	NaN	NaN	NaN	NaN	NaN
std	54506.606056	NaN	7.146679	NaN	6.454356	NaN	0.477965	NaN	NaN	NaN	NaN	NaN	NaN
min	2.000000	NaN	40.000000	NaN	158.000000	NaN	0.000000	NaN	NaN	NaN	NaN	NaN	NaN
25%	140001.500000	NaN	62.000000	NaN	178.000000	NaN	0.000000	NaN	NaN	NaN	NaN	NaN	NaN
50%	171578.000000	NaN	66.000000	NaN	182.000000	NaN	0.000000	NaN	NaN	NaN	NaN	NaN	NaN
75%	189185.000000	NaN	72.000000	NaN	186.000000	NaN	1.000000	NaN	NaN	NaN	NaN	NaN	NaN
max	205583.000000	NaN	94.000000	NaN	208.000000	NaN	1.000000	NaN	NaN	NaN	NaN	NaN	NaN

Replace : Pandas **`dataframe.replace()`** function is used to replace a string, regex, list, dictionary, series, number etc. from a dataframe. This is a very rich function as it has many variations.
The most powerful thing about this function is that it can work with Python regex (regular expressions).

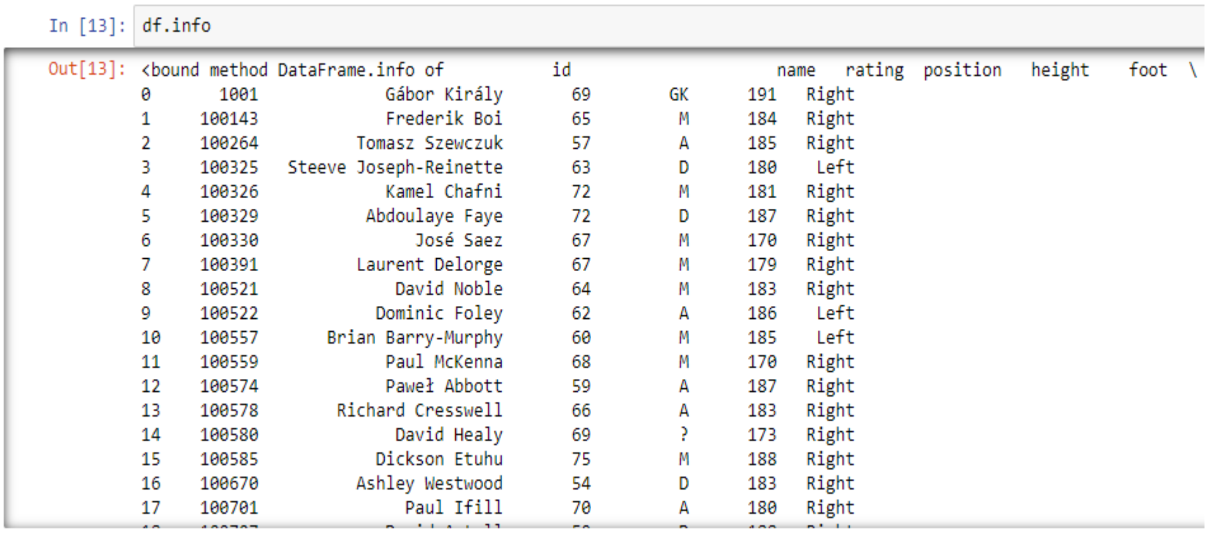


```
In [16]: import numpy as np
         df.replace("?",np.nan,inplace=True)
         df.head(5)
```

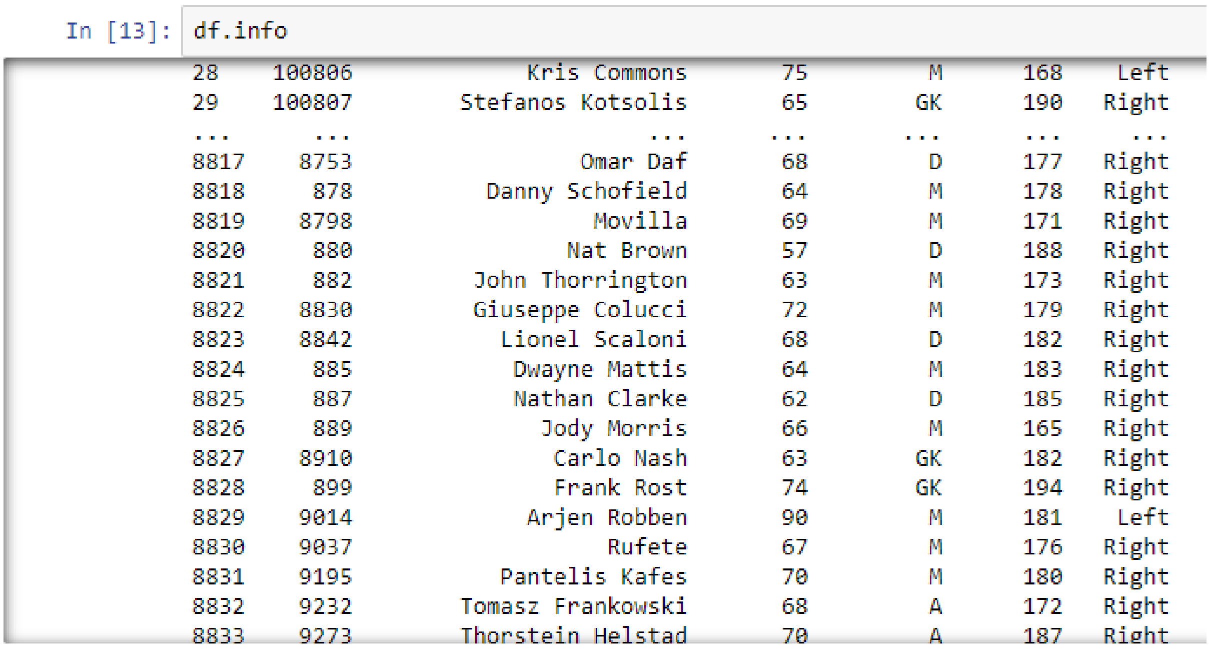
Out[16]:

	id	name	rating	position	height	foot	rare	pace	shooting	passing	dribbling	defending	heading
0	1001	Gábor Király	69	GK	191	Right	0						
1	100143	Frederik Boi	65	M	184	Right	0	61	65	63	59	62	62
2	100264	Tomasz Szewczuk	57	A	185	Right	0	65	54	43	53	55	74
3	100325	Steeve Joseph-Reinette	63	D	180	Left	0	68	38	51	46	64	71
4	100326	Kamel Chafni	72	M	181	Right	0	75	64	67	72	57	66

__Info:__ Pandas **`dataframe.info()`** function is used to get a concise summary of the dataframe. It comes really handy when doing exploratory analysis of the data. To get a quick overview of the dataset we use the `dataframe.info()` function.



```
In [13]: df.info

Out[13]: <bound method DataFrame.info of            id                name  rating  position  height   foot  \
         0       1001         Gábor Király      69        GK     191  Right
         1     100143         Frederik Boi      65         M     184  Right
         2     100264      Tomasz Szewczuk      57         A     185  Right
         3     100325  Steeve Joseph-Reinette   63         D     180   Left
         4     100326         Kamel Chafni      72         M     181  Right
         5     100329        Abdoulaye Faye      72         D     187  Right
         6     100330            José Saez      67         M     170  Right
         7     100391      Laurent Delorge      67         M     179  Right
         8     100521          David Noble      64         M     183  Right
         9     100522        Dominic Foley      62         A     186   Left
         10    100557    Brian Barry-Murphy     60         M     185   Left
         11    100559          Paul McKenna      68         M     170  Right
         12    100574         Paweł Abbott      59         A     187  Right
         13    100578     Richard Cresswell     66         A     183  Right
         14    100580          David Healy      69         ?     173  Right
         15    100585        Dickson Etuhu      75         M     188  Right
         16    100670      Ashley Westwood      54         D     183  Right
         17    100701           Paul Ifill      70         A     180  Right
```



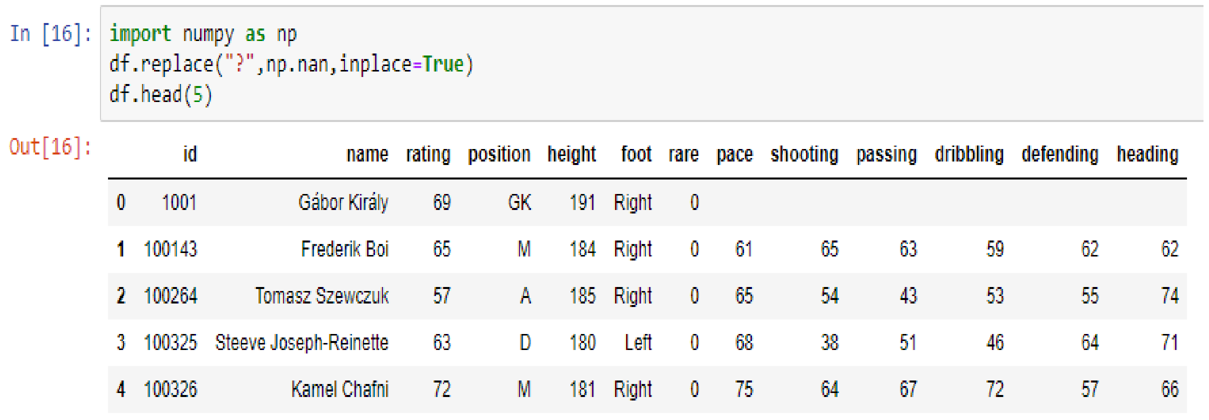
```
In [13]: df.info

         28    100806       Kris Commons      75         M     168   Left
         29    100807    Stefanos Kotsolis    65        GK     190  Right
         ...      ...                ...     ...       ...     ...    ...
         8817    8753           Omar Daf      68         D     177  Right
         8818     878     Danny Schofield     64         M     178  Right
         8819    8798            Movilla      69         M     171  Right
         8820     880           Nat Brown      57         D     188  Right
         8821     882    John Thorrington     63         M     173  Right
         8822    8830    Giuseppe Colucci     72         M     179  Right
         8823    8842      Lionel Scaloni     68         D     182  Right
         8824     885       Dwayne Mattis     64         M     183  Right
         8825     887       Nathan Clarke     62         D     185  Right
         8826     889         Jody Morris     66         M     165  Right
         8827    8910          Carlo Nash     63        GK     182  Right
         8828     899          Frank Rost     74        GK     194  Right
         8829    9014        Arjen Robben     90         M     181   Left
         8830    9037              Rufete     67         M     176  Right
         8831    9195       Pantelis Kafes    70         M     180  Right
         8832    9232    Tomasz Frankowski    68         A     172  Right
         8833    9273    Thorstein Helstad    70         A     187  Right
```

Identify missing values

Convert "?" to NaN
In the car dataset, missing data comes with the question mark "?". We replace "?" with NaN (Not a Number), which is Python's default missing value marker, for reasons of computational speed and convenience. Here we use the function:

```
.replace(A, B, inplace = True)
```

to replace A by B



```
In [16]: import numpy as np
         df.replace("?",np.nan,inplace=True)
         df.head(5)
```

Out[16]:

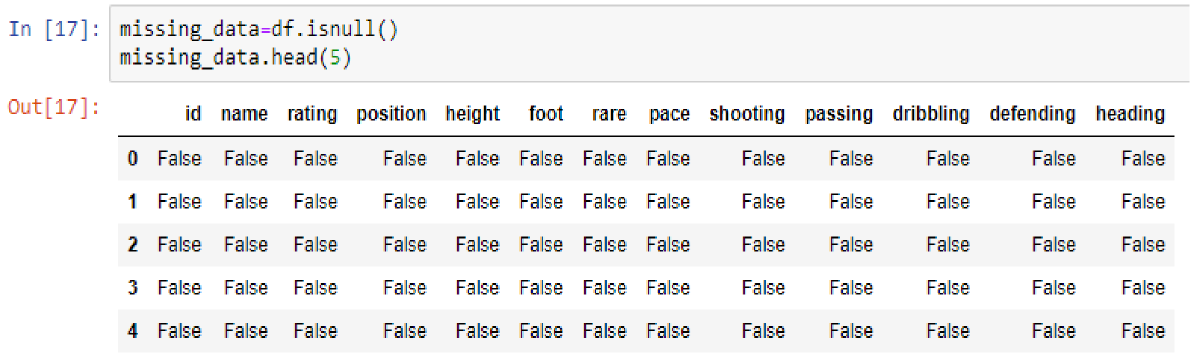
	id	name	rating	position	height	foot	rare	pace	shooting	passing	dribbling	defending	heading
0	1001	Gábor Király	69	GK	191	Right	0						
1	100143	Frederik Boi	65	M	184	Right	0	61	65	63	59	62	62
2	100264	Tomasz Szewczuk	57	A	185	Right	0	65	54	43	53	55	74
3	100325	Steeve Joseph-Reinette	63	D	180	Left	0	68	38	51	46	64	71
4	100326	Kamel Chafni	72	M	181	Right	0	75	64	67	72	57	66

Evaluating for Missing Data

The missing values are converted to Python's default. We use Python's built-in functions to identify these missing values. There are two methods to detect missing data:

1. **.isnull()**
2. **.notnull()**

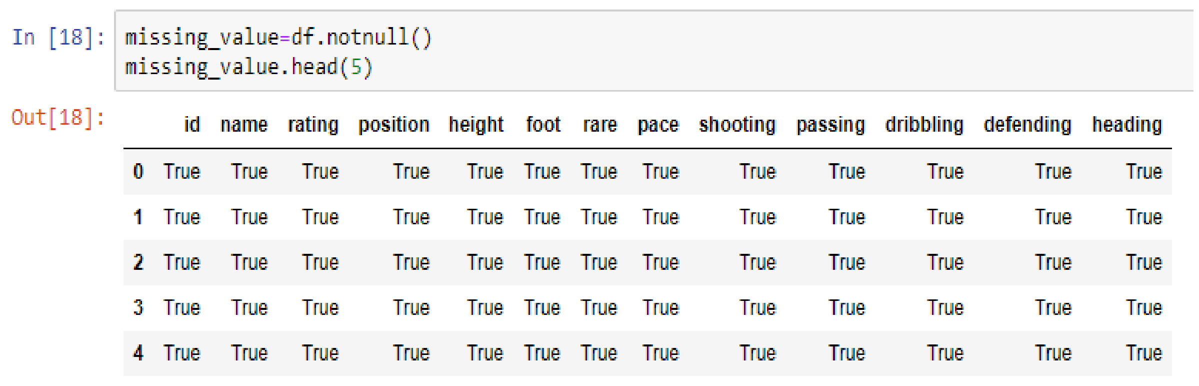
The output is a boolean value indicating whether the value that is passed into the argument is in fact missing data.



```
In [17]: missing_data=df.isnull()
         missing_data.head(5)
```

Out[17]:

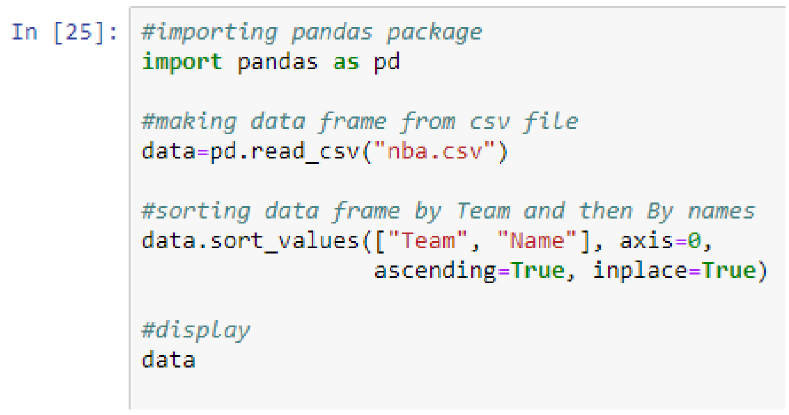
	id	name	rating	position	height	foot	rare	pace	shooting	passing	dribbling	defending	heading
0	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False

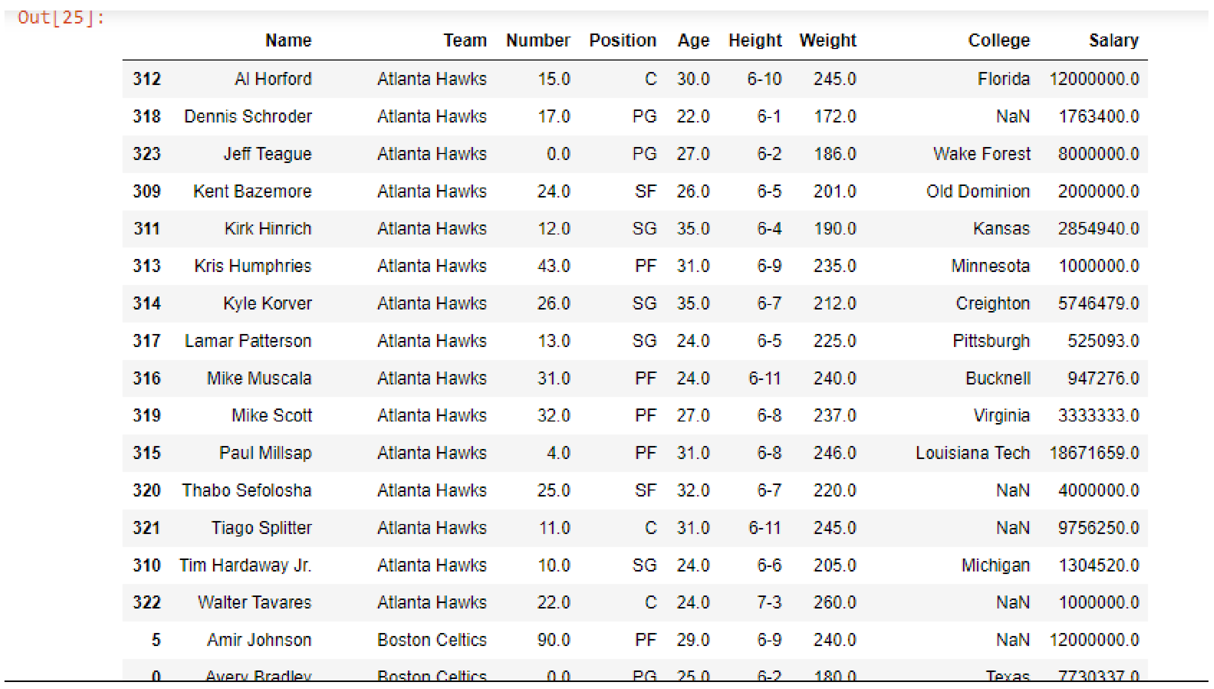

In [18]:
```python
missing_value=df.notnull()
missing_value.head(5)
```

Out[18]:

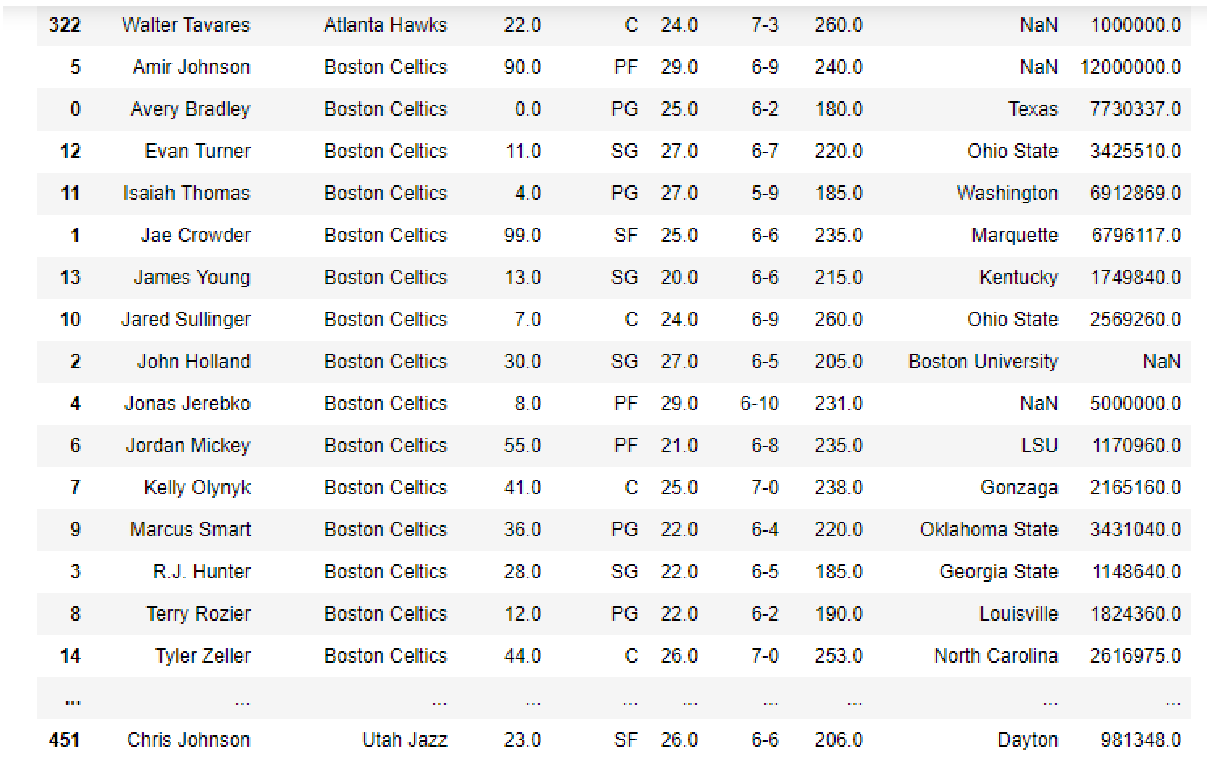
	id	name	rating	position	height	foot	rare	pace	shooting	passing	dribbling	defending	heading
0	True	True	True	True	True	True	True	True	True	True	True	True	True
1	True	True	True	True	True	True	True	True	True	True	True	True	True
2	True	True	True	True	True	True	True	True	True	True	True	True	True
3	True	True	True	True	True	True	True	True	True	True	True	True	True
4	True	True	True	True	True	True	True	True	True	True	True	True	True

Sort: Pandas `sort_values()` function sorts a data frame in Ascending or Descending order of passed Column. It's different than the sorted Python function since it cannot sort a data frame and particular column cannot be selected.


In [25]:
```python
#importing pandas package
import pandas as pd

#making data frame from csv file
data=pd.read_csv("nba.csv")

#sorting data frame by Team and then By names
data.sort_values(["Team", "Name"], axis=0,
                 ascending=True, inplace=True)

#display
data
```

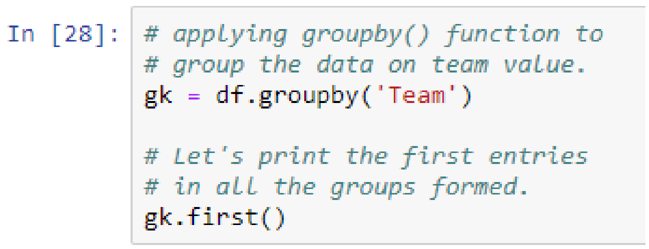

Out[25]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
312	Al Horford	Atlanta Hawks	15.0	C	30.0	6-10	245.0	Florida	12000000.0
318	Dennis Schroder	Atlanta Hawks	17.0	PG	22.0	6-1	172.0	NaN	1763400.0
323	Jeff Teague	Atlanta Hawks	0.0	PG	27.0	6-2	186.0	Wake Forest	8000000.0
309	Kent Bazemore	Atlanta Hawks	24.0	SF	26.0	6-5	201.0	Old Dominion	2000000.0
311	Kirk Hinrich	Atlanta Hawks	12.0	SG	35.0	6-4	190.0	Kansas	2854940.0
313	Kris Humphries	Atlanta Hawks	43.0	PF	31.0	6-9	235.0	Minnesota	1000000.0
314	Kyle Korver	Atlanta Hawks	26.0	SG	35.0	6-7	212.0	Creighton	5746479.0
317	Lamar Patterson	Atlanta Hawks	13.0	SG	24.0	6-5	225.0	Pittsburgh	525093.0
316	Mike Muscala	Atlanta Hawks	31.0	PF	24.0	6-11	240.0	Bucknell	947276.0
319	Mike Scott	Atlanta Hawks	32.0	PF	27.0	6-8	237.0	Virginia	3333333.0
315	Paul Millsap	Atlanta Hawks	4.0	PF	31.0	6-8	246.0	Louisiana Tech	18671659.0
320	Thabo Sefolosha	Atlanta Hawks	25.0	SF	32.0	6-7	220.0	NaN	4000000.0
321	Tiago Splitter	Atlanta Hawks	11.0	C	31.0	6-11	245.0	NaN	9756250.0
310	Tim Hardaway Jr.	Atlanta Hawks	10.0	SG	24.0	6-6	205.0	Michigan	1304520.0
322	Walter Tavares	Atlanta Hawks	22.0	C	24.0	7-3	260.0	NaN	1000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0

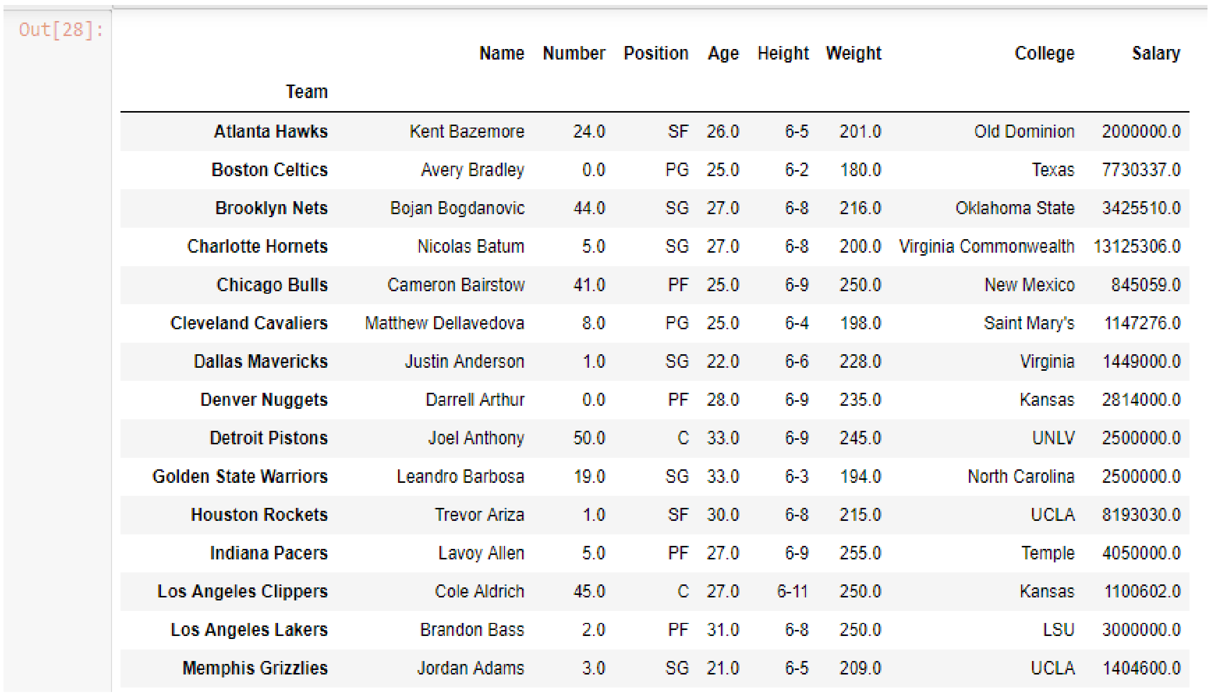
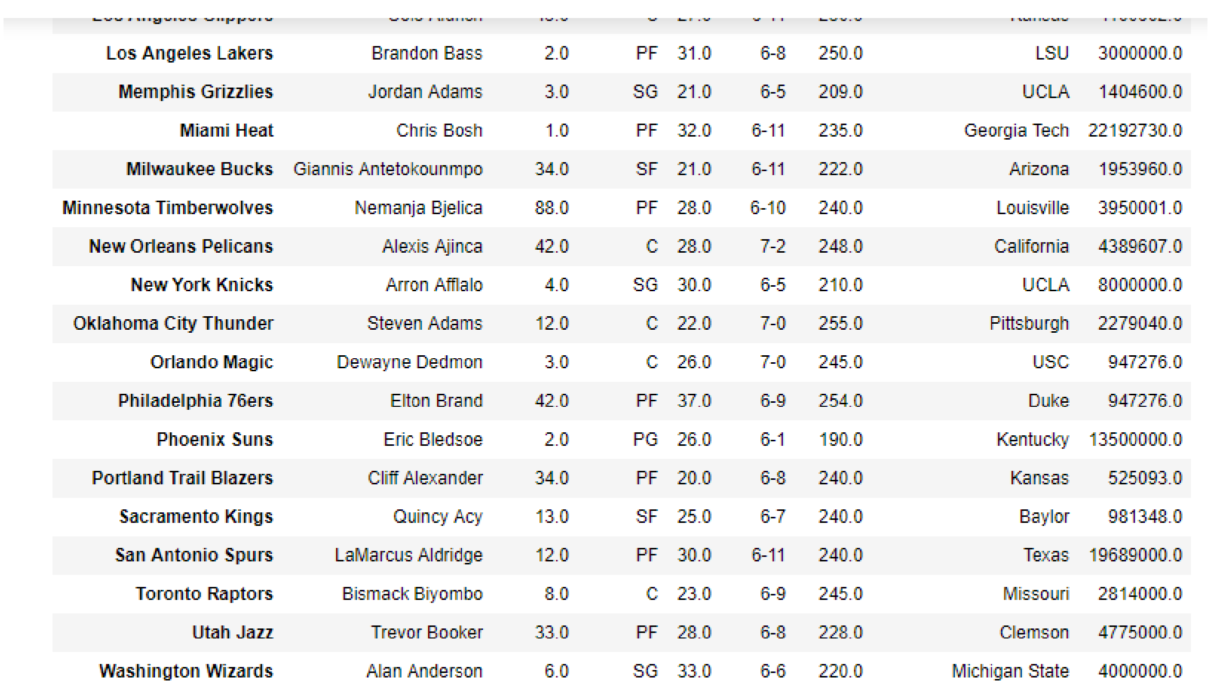


322	Walter Tavares	Atlanta Hawks	22.0	C	24.0	7-3	260.0	NaN	1000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
12	Evan Turner	Boston Celtics	11.0	SG	27.0	6-7	220.0	Ohio State	3425510.0
11	Isaiah Thomas	Boston Celtics	4.0	PG	27.0	5-9	185.0	Washington	6912869.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
13	James Young	Boston Celtics	13.0	SG	20.0	6-6	215.0	Kentucky	1749840.0
10	Jared Sullinger	Boston Celtics	7.0	C	24.0	6-9	260.0	Ohio State	2569260.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
6	Jordan Mickey	Boston Celtics	55.0	PF	21.0	6-8	235.0	LSU	1170960.0
7	Kelly Olynyk	Boston Celtics	41.0	C	25.0	7-0	238.0	Gonzaga	2165160.0
9	Marcus Smart	Boston Celtics	36.0	PG	22.0	6-4	220.0	Oklahoma State	3431040.0
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
8	Terry Rozier	Boston Celtics	12.0	PG	22.0	6-2	190.0	Louisville	1824360.0
14	Tyler Zeller	Boston Celtics	44.0	C	26.0	7-0	253.0	North Carolina	2616975.0
...	...	...	...	...	...	...	...	...	...
451	Chris Johnson	Utah Jazz	23.0	SF	26.0	6-6	206.0	Dayton	981348.0

Groupby: Pandas **`dataframe.groupby()`** function is used to split the data into groups based on some criteria. pandas objects can be split on any of their axes. The abstract definition of grouping is to provide a mapping of labels to group names.



```
In [28]: # applying groupby() function to
         # group the data on team value.
         gk = df.groupby('Team')

         # Let's print the first entries
         # in all the groups formed.
         gk.first()
```

Out[28]:

	Name	Number	Position	Age	Height	Weight	College	Salary
Team								
Atlanta Hawks	Kent Bazemore	24.0	SF	26.0	6-5	201.0	Old Dominion	2000000.0
Boston Celtics	Avery Bradley	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
Brooklyn Nets	Bojan Bogdanovic	44.0	SG	27.0	6-8	216.0	Oklahoma State	3425510.0
Charlotte Hornets	Nicolas Batum	5.0	SG	27.0	6-8	200.0	Virginia Commonwealth	13125306.0
Chicago Bulls	Cameron Bairstow	41.0	PF	25.0	6-9	250.0	New Mexico	845059.0
Cleveland Cavaliers	Matthew Dellavedova	8.0	PG	25.0	6-4	198.0	Saint Mary's	1147276.0
Dallas Mavericks	Justin Anderson	1.0	SG	22.0	6-6	228.0	Virginia	1449000.0
Denver Nuggets	Darrell Arthur	0.0	PF	28.0	6-9	235.0	Kansas	2814000.0
Detroit Pistons	Joel Anthony	50.0	C	33.0	6-9	245.0	UNLV	2500000.0
Golden State Warriors	Leandro Barbosa	19.0	SG	33.0	6-3	194.0	North Carolina	2500000.0
Houston Rockets	Trevor Ariza	1.0	SF	30.0	6-8	215.0	UCLA	8193030.0
Indiana Pacers	Lavoy Allen	5.0	PF	27.0	6-9	255.0	Temple	4050000.0
Los Angeles Clippers	Cole Aldrich	45.0	C	27.0	6-11	250.0	Kansas	1100602.0
Los Angeles Lakers	Brandon Bass	2.0	PF	31.0	6-8	250.0	LSU	3000000.0
Memphis Grizzlies	Jordan Adams	3.0	SG	21.0	6-5	209.0	UCLA	1404600.0
Miami Heat	Chris Bosh	1.0	PF	32.0	6-11	235.0	Georgia Tech	22192730.0
Milwaukee Bucks	Giannis Antetokounmpo	34.0	SF	21.0	6-11	222.0	Arizona	1953960.0
Minnesota Timberwolves	Nemanja Bjelica	88.0	PF	28.0	6-10	240.0	Louisville	3950001.0
New Orleans Pelicans	Alexis Ajinca	42.0	C	28.0	7-2	248.0	California	4389607.0
New York Knicks	Arron Afflalo	4.0	SG	30.0	6-5	210.0	UCLA	8000000.0
Oklahoma City Thunder	Steven Adams	12.0	C	22.0	7-0	255.0	Pittsburgh	2279040.0
Orlando Magic	Dewayne Dedmon	3.0	C	26.0	7-0	245.0	USC	947276.0
Philadelphia 76ers	Elton Brand	42.0	PF	37.0	6-9	254.0	Duke	947276.0
Phoenix Suns	Eric Bledsoe	2.0	PG	26.0	6-1	190.0	Kentucky	13500000.0
Portland Trail Blazers	Cliff Alexander	34.0	PF	20.0	6-8	240.0	Kansas	525093.0
Sacramento Kings	Quincy Acy	13.0	SF	25.0	6-7	240.0	Baylor	981348.0
San Antonio Spurs	LaMarcus Aldridge	12.0	PF	30.0	6-11	240.0	Texas	19689000.0
Toronto Raptors	Bismack Biyombo	8.0	C	23.0	6-9	245.0	Missouri	2814000.0
Utah Jazz	Trevor Booker	33.0	PF	28.0	6-8	228.0	Clemson	4775000.0
Washington Wizards	Alan Anderson	6.0	SG	33.0	6-6	220.0	Michigan State	4000000.0