In [1]:

```
#########################################################
# DataFrames part I
# Author : Rodrique KAFANDO
# Destination : Master FD&IA - UV-BF
# 12.07.2021
#########################################################
```

DataFrames

- Two dimensional data structure
- Combination of rows and columns
- besoin d'une combinason de plus de 2 refs pour extraire une valeur
- we need two points of references to extract a given value
- 3-dimensional example : imagine we have two tables, we need to provide respectively 1) the table, 2) the cls and 3) the rows.

In [2]:

```
# labraries
import pandas as pd
```

In [3]:

```
nba = pd.read_csv('./pandas/nba.csv')
nba
```

Out[3]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
...	...	...	...	...	...	...	...	...	...
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	24.0	6-1	179.0	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21.0	C	26.0	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	Kansas	947276.0
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

458 rows × 9 columns

Shared Methods and attributs between Series and

DataFrames

In [4]:

```
nba.head()
```

Out[4]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0

In [5]:

```
# index attribute
nba.index
```

Out[5]:

```
RangeIndex(start=0, stop=458, step=1)
```

In [6]:

```
# numpy array (list of lists), where each dim represente a DF rows
# values attribute
nba.values
```

Out[6]:

```
array([['Avery Bradley', 'Boston Celtics', 0.0, ..., 180.0, 'Texas',
        7730337.0],
       ['Jae Crowder', 'Boston Celtics', 99.0, ..., 235.0, 'Marquette',
        6796117.0],
       ['John Holland', 'Boston Celtics', 30.0, ..., 205.0,
        'Boston University', nan],
       ...,
       ['Tibor Pleiss', 'Utah Jazz', 21.0, ..., 256.0, nan, 2900000.0],
       ['Jeff Withey', 'Utah Jazz', 24.0, ..., 231.0, 'Kansas', 947276.0],
       [nan, nan, nan, ..., nan, nan, nan]], dtype=object)
```

In [7]:

```
# shape attributs,
nba.shape
```

Out[7]:

```
(458, 9)
```

In [8]:

```
# dtypes
nba.dtypes
```

Out[8]:

```
Name         object
Team         object
Number      float64
Position     object
Age         float64
Height       object
Weight      float64
College      object
Salary      float64
dtype: object
```

In [9]:

```
# value_counts() method on dtypes
nba.dtypes.value_counts()
```

Out[9]:

```
object     5
float64    4
dtype: int64
```

In [10]:

```
nba.columns
```

Out[10]:

```
Index(['Name', 'Team', 'Number', 'Position', 'Age', 'Height', 'Weight',
       'College', 'Salary'],
      dtype='object')
```

In [11]:

```
nba.axes
```

Out[11]:

```
[RangeIndex(start=0, stop=458, step=1),
 Index(['Name', 'Team', 'Number', 'Position', 'Age', 'Height', 'Weight',
        'College', 'Salary'],
       dtype='object')]
```

In [12]:

```
# get global infos about the DF
nba.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 458 entries, 0 to 457
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Name      457 non-null    object 
 1   Team      457 non-null    object 
 2   Number    457 non-null    float64
 3   Position  457 non-null    object 
 4   Age       457 non-null    float64
 5   Height    457 non-null    object 
 6   Weight    457 non-null    float64
 7   College   373 non-null    object 
 8   Salary    446 non-null    float64
dtypes: float64(4), object(5)
memory usage: 32.3+ KB
```

In [13]:

```
len(nba)
```

Out[13]:

```
458
```

Difference between shared Methods

- show how a method will operate differenetely on a Series and a DataFrame

In [14]:

```
rev = pd.read_csv('./pandas/revenue.csv' , index_col = 'Date')
```

In [15]:

```
rev.head(3)
```

Out[15]:

	New York	Los Angeles	Miami
Date			

	New York	Los Angeles	Miami
Date			
1/1/16	985	122	499
1/2/16	738	788	534
1/3/16	14	20	933

In [16]:

```
# sum() method
s = pd.Series([1,3,4])
s.sum()
```

Out[16]:

```
8
```

In [17]:

```
# will show the sum for each column values
rev.sum()
```

Out[17]:

```
New York       5475
Los Angeles    5134
Miami          5641
dtype: int64
```

In [18]:

```
# now, how if we want to have the sum on rows instead columns in
DataFrames ? Series don't have that method
rev.sum(axis = 0)
```

Out[18]:

```
New York       5475
Los Angeles    5134
Miami          5641
dtype: int64
```

In [19]:

```
# now, how if we want to have the sum on rows instead columns in
DataFrames ? Series don't have that method
rev.sum(axis = 'index')
```

Out[19]:

```
New York       5475
Los Angeles    5134
Miami          5641
dtype: int64
```

In [20]:

```
# now, how if we want to have the sum on rows instead columns in
DataFrames ? Series don't have that method
rev.sum(axis = 1)
```

Out[20]:

```
Date
1/1/16     1606
1/2/16     2060
1/3/16      967
1/4/16     2519
1/5/16      438
1/6/16     1935
1/7/16     1234
1/8/16     2313
1/9/16     2623
```

```
1/10/16     555
dtype: int64
```

In [21]:

```
# now, how if we want to have the sum on rows instead columns in DataFrames ? Series don't have that method
rev.sum(axis = 'columns')
```

Out[21]:

```
Date
1/1/16     1606
1/2/16     2060
1/3/16      967
1/4/16     2519
1/5/16      438
1/6/16     1935
1/7/16     1234
1/8/16     2313
1/9/16     2623
1/10/16     555
dtype: int64
```

Select One column from a DataFrame

In [22]:

```
# first option, if the column name doesn't contaisn space or more than one string
# it's case sensitive
nba.Name
```

Out[22]:

```
0      Avery Bradley
1        Jae Crowder
2       John Holland
3        R.J. Hunter
4      Jonas Jerebko
           ...
453     Shelvin Mack
454        Raul Neto
455     Tibor Pleiss
456      Jeff Withey
457              NaN
Name: Name, Length: 458, dtype: object
```

In [23]:

```
# second option, garanty to work all the time, use bracket's syntax
nba['Name']
```

Out[23]:

```
0      Avery Bradley
1        Jae Crowder
2       John Holland
3        R.J. Hunter
4      Jonas Jerebko
           ...
453     Shelvin Mack
454        Raul Neto
455     Tibor Pleiss
456      Jeff Withey
457              NaN
Name: Name, Length: 458, dtype: object
```

In [24]:

```
# one column will be extracted as pandas series
type(nba['Name'])
```

Out[24]:
```
pandas.core.series.Series
```

Select two or more columns from a pandas DF

In [25]:
```
nba.head(3)
```

Out[25]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN

In [26]:
```
# indiquer les noms de colonnes à etraire dans une list
nba[ ['Name','Team','Salary'] ]
```

Out[26]:

	Name	Team	Salary
0	Avery Bradley	Boston Celtics	7730337.0
1	Jae Crowder	Boston Celtics	6796117.0
2	John Holland	Boston Celtics	NaN
3	R.J. Hunter	Boston Celtics	1148640.0
4	Jonas Jerebko	Boston Celtics	5000000.0
...	...	...	...
453	Shelvin Mack	Utah Jazz	2433333.0
454	Raul Neto	Utah Jazz	900000.0
455	Tibor Pleiss	Utah Jazz	2900000.0
456	Jeff Withey	Utah Jazz	947276.0
457	NaN	NaN	NaN

458 rows × 3 columns

In [27]:
```
# indiquer les noms de colonnes à etraire dans une list
nba[ ['Salary', 'Name','Team'] ].head(3)
```

Out[27]:

	Salary	Name	Team
0	7730337.0	Avery Bradley	Boston Celtics
1	6796117.0	Jae Crowder	Boston Celtics
2	NaN	John Holland	Boston Celtics

In [28]:
```
# indiquer les noms de colonnes à etraire dans une list
my_choice = ['Salary', 'Name','Team']
nba[my_choice].head(3)
```

Out[28]:

	Salary	Name	Team
0	7730337.0	Avery Bradley	Boston Celtics
1	6796117.0	Jae Crowder	Boston Celtics
2	NaN	John Holland	Boston Celtics

Add new columns to an existing DF

In [29]:

```
# first method
nba['new_game'] = 'okay'
```

In [30]:

```
nba.head()
```

Out[30]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary	new_game
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0	okay
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0	okay
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN	okay
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0	okay
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0	okay

In [31]:

```
# second method by using insert() method
nba.insert(2, column = 'newcols', value = 'sample')
```

In [32]:

```
nba.head(3)
```

Out[32]:

	Name	Team	newcols	Number	Position	Age	Height	Weight	College	Salary	new_
0	Avery Bradley	Boston Celtics	sample	0.0	PG	25.0	6-2	180.0	Texas	7730337.0	
1	Jae Crowder	Boston Celtics	sample	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0	
2	John Holland	Boston Celtics	sample	30.0	SG	27.0	6-5	205.0	Boston University	NaN	

Broadcasting operations

In [33]:

```
nba.head(3)
```

Out[33]:

	Name	Team	newcols	Number	Position	Age	Height	Weight	College	Salary	new_
0	Avery Bradley	Boston Celtics	sample	0.0	PG	25.0	6-2	180.0	Texas	7730337.0	

	Name	Team	newcols	Number	Position	Age	Height	Weight	College	Salary	new_
1	Jae Crowder	Boston Celtics	sample	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0	
2	John Holland	Boston Celtics	sample	30.0	SG	27.0	6-5	205.0	Boston University	NaN	

In [34]:

```
# ajouter 5 ans sur l'age de chaque joueur
nba.Age.add(5).head(3)
# ajouter 5 ans sur l'age de chaque joueur
nba['sumtest'] = nba.Age.add(5).head(3)
nba
```

Out[34]:

	Name	Team	newcols	Number	Position	Age	Height	Weight	College	Salary	ne
0	Avery Bradley	Boston Celtics	sample	0.0	PG	25.0	6-2	180.0	Texas	7730337.0	
1	Jae Crowder	Boston Celtics	sample	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0	
2	John Holland	Boston Celtics	sample	30.0	SG	27.0	6-5	205.0	Boston University	NaN	
3	R.J. Hunter	Boston Celtics	sample	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0	
4	Jonas Jerebko	Boston Celtics	sample	8.0	PF	29.0	6-10	231.0	NaN	5000000.0	
...	...	...	...	...	...	...	...	...	...	...	
453	Shelvin Mack	Utah Jazz	sample	8.0	PG	26.0	6-3	203.0	Butler	2433333.0	
454	Raul Neto	Utah Jazz	sample	25.0	PG	24.0	6-1	179.0	NaN	900000.0	
455	Tibor Pleiss	Utah Jazz	sample	21.0	C	26.0	7-3	256.0	NaN	2900000.0	
456	Jeff Withey	Utah Jazz	sample	24.0	C	26.0	7-0	231.0	Kansas	947276.0	
457	NaN	NaN	sample	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

458 rows × 12 columns

In [35]:

```
# ajouter 5 ans sur l'age de chaque joueur
nba.Age + 5
```

Out[35]:

```
0      30.0
1      30.0
2      32.0
3      27.0
4      34.0
       ...
453    31.0
454    29.0
455    31.0
456    31.0
457     NaN
Name: Age, Length: 458, dtype: float64
```

In [36]:

```
# can also use sub() and mul() functions
```

A review of .value_counts() lethod

In [37]:

```
# most common team
nba['Team'].value_counts().head(3)
```

Out[37]:

```
New Orleans Pelicans    19
Memphis Grizzlies       18
Milwaukee Bucks         16
Name: Team, dtype: int64
```

In [38]:

```
# most common salary
nba['Salary'].value_counts().head(3)
```

Out[38]:

```
947276.0    31
845059.0    18
525093.0    13
Name: Salary, dtype: int64
```

Drop Null values

In [39]:

```
nba = pd.read_csv('./pandas/nba.csv')
nba
```

Out[39]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
...	...	...	...	...	...	...	...	...	...
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	24.0	6-1	179.0	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21.0	C	26.0	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	Kansas	947276.0
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

458 rows × 9 columns

In [40]:

```
# drop rows with null values - dropnan()  method
nba.dropna()
```

Out[40]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
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
...	...	...	...	...	...	...	...	...	...
449	Rodney Hood	Utah Jazz	5.0	SG	23.0	6-8	206.0	Duke	1348440.0
451	Chris Johnson	Utah Jazz	23.0	SF	26.0	6-6	206.0	Dayton	981348.0
452	Trey Lyles	Utah Jazz	41.0	PF	20.0	6-10	234.0	Kentucky	2239800.0
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	Butler	2433333.0
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	Kansas	947276.0

364 rows × 9 columns

In [41]:

```
# drop rows with null values - dropnan()  method
nba.dropna(how = 'all')
```

Out[41]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
...	...	...	...	...	...	...	...	...	...
452	Trey Lyles	Utah Jazz	41.0	PF	20.0	6-10	234.0	Kentucky	2239800.0
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	24.0	6-1	179.0	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21.0	C	26.0	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	Kansas	947276.0

457 rows × 9 columns

In [42]:

```
# drop rows with null values - dropnan()  method
```

```
#
# nba.dropna(axis=0, how='any') => nba.dropna()
```

In [43]:

```
# drop cols with null values - dropnan()  method
#
nba.dropna(axis=1, how='all')
```

Out[43]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
...	...	...	...	...	...	...	...	...	...
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	24.0	6-1	179.0	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21.0	C	26.0	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	Kansas	947276.0
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

458 rows × 9 columns

In [44]:

```
# from a specific columns
nba.dropna(subset = ['Salary'])
```

Out[44]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
...	...	...	...	...	...	...	...	...	...
452	Trey Lyles	Utah Jazz	41.0	PF	20.0	6-10	234.0	Kentucky	2239800.0
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	24.0	6-1	179.0	NaN	900000.0

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
455	Tibor Pleiss	Utah Jazz	21.0	C	26.0	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	Kansas	947276.0

446 rows × 9 columns

In [ ]:

```

```

Fill in NULL values with .fillna() method

- replace NaN value with a specific one

In [45]:

```
# use directely .fillna() method directly/uppon dataframe
# RQ : will not take into account the data type for each column
nba.head(6)
```

Out[45]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0

In [46]:

```
# nous remarquons que les valeurs de college sont aussi remplacées
par 0, ce qui donnent une certaine incoherence
nba.fillna(0).head(6)
```

Out[46]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	0.0
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	0	5000000.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	0	12000000.0

In [47]:

```
nba
```

Out[47]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
...	...	...	...	...	...	...	...	...	...
453	Shelvin Mack	Utah Jazz	8.0	PG	26.0	6-3	203.0	Butler	2433333.0
454	Raul Neto	Utah Jazz	25.0	PG	24.0	6-1	179.0	NaN	900000.0
455	Tibor Pleiss	Utah Jazz	21.0	C	26.0	7-3	256.0	NaN	2900000.0
456	Jeff Withey	Utah Jazz	24.0	C	26.0	7-0	231.0	Kansas	947276.0
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

458 rows × 9 columns

In [48]:

```
# resoudre le probleme precedent en specifiant la colonne
nba['Salary'].fillna(0, inplace = True)
```

In [49]:

```
nba.head()
```

Out[49]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	0.0
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0

In [50]:

```
# resoudre le probleme precedent en specifiant la colonne, pour
str()
nba['College'].fillna('no college', inplace = True)
```

In [51]:

```
nba.head()
```

Out[51]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	0.0
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	no college	5000000.0

The .astype() method

- convert DataFrame type with astype() method
- require une serie ayant pas de valeur NaN, d'où l'importance de la phase précedente

In [52]:

```
# let's check data type
nba = pd.read_csv('./pandas/nba.csv')
nba.head()
```

Out[52]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0

In [53]:

```
# dtypes attributes
# object => python internal syntax for string
nba.dtypes
# or to get the data types
nba.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 458 entries, 0 to 457
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Name      457 non-null    object 
 1   Team      457 non-null    object 
 2   Number    457 non-null    float64
 3   Position  457 non-null    object 
 4   Age       457 non-null    float64
 5   Height    457 non-null    object 
 6   Weight    457 non-null    float64
 7   College   373 non-null    object 
 8   Salary    446 non-null    float64
dtypes: float64(4), object(5)
memory usage: 32.3+ KB
```

In [56]:

```
# show error, why, and how to solve it
nba['Salary'].astype(int)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-56-a5ef86d4c025> in <module>
      1 # show error, why, and how to solve it
----> 2 nba['Salary'].astype(int)

~/anaconda3/lib/python3.7/site-packages/pandas/core/generic.py in astype(sel
f, dtype, copy, errors)
   5544         else:
   5545             # else, only a single dtype is given
-> 5546             new_data = self._mgr.astype(dtype=dtype, copy=copy, error
s=errors,)
   5547             return self._constructor(new_data).__finalize__(self, met
hod="astype")
   5548 

~/anaconda3/lib/python3.7/site-packages/pandas/core/internals/managers.py in
astype(self, dtype, copy, errors)
    593         self, dtype, copy: bool = False, errors: str = "raise"
    594     ) -> "BlockManager":
--> 595         return self.apply("astype", dtype=dtype, copy=copy, errors=er
rors)
    596 
    597     def convert(

~/anaconda3/lib/python3.7/site-packages/pandas/core/internals/managers.py in
apply(self, f, align_keys, **kwargs)
    404                 applied = b.apply(f, **kwargs)
    405             else:
--> 406                 applied = getattr(b, f)(**kwargs)
    407             result_blocks = _extend_blocks(applied, result_blocks)
    408 

~/anaconda3/lib/python3.7/site-packages/pandas/core/internals/blocks.py in as
type(self, dtype, copy, errors)
    593             vals1d = values.ravel()
    594             try:
--> 595                 values = astype_nansafe(vals1d, dtype, copy=True)
    596             except (ValueError, TypeError):
    597                 # e.g. astype_nansafe can fail on object-dtype of str
ings

~/anaconda3/lib/python3.7/site-packages/pandas/core/dtypes/cast.py in astype_
nansafe(arr, dtype, copy, skipna)
    964 
    965         if not np.isfinite(arr).all():
--> 966             raise ValueError("Cannot convert non-finite values (NA or
inf) to integer")
    967 
    968     elif is_object_dtype(arr):

ValueError: Cannot convert non-finite values (NA or inf) to integer
```

In [57]:

```
# delete  rows which all values are NaN
nba = pd.read_csv('./pandas/nba.csv').dropna(how = 'all')
```

In [58]:

```
nba.head()
```

Out[58]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0

In [59]:

```
nba.Salary.fillna(0,inplace = True)
nba.College.fillna('None',inplace = True)
```

In [60]:

```
# show error, why, and how to solve it
# re_execute the .info to see the added data types
nba['Salary'].astype(int) #or 'int'
```

Out[60]:

```
0      7730337
1      6796117
2            0
3      1148640
4      5000000
        ...
452    2239800
453    2433333
454     900000
455    2900000
456     947276
Name: Salary, Length: 457, dtype: int64
```

In [61]:

```
# nous pouvons utiliser aussi .nuninque() method pour constater
les valeurs uniques de chaque colonne
#
nba.Position.nunique()
```

Out[61]:

```
5
```

In [62]:

```
# definir ensuite ces valeurs uniques comme étant des catégories
=> avantage, il ermet de reduire la taille des données en memo
# verifier la taille => memory usage: 35.7+ KB
nba.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 457 entries, 0 to 456
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Name      457 non-null    object 
 1   Team      457 non-null    object 
 2   Number    457 non-null    float64
 3   Position  457 non-null    object 
 4   Age       457 non-null    float64
 5   Height    457 non-null    object 
 6   Weight    457 non-null    float64
 7   College   457 non-null    object 
 8   Salary    457 non-null    float64
dtypes: float64(4), object(5)
memory usage: 35.7+ KB
```

In [63]:

```
nba.Position = nba.Position.astype('category')
```

In [64]:

```
nba.info() # to memory usage: 32.8+ KB
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 457 entries, 0 to 456
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype   
---  ------    --------------  -----   
 0   Name      457 non-null    object  
 1   Team      457 non-null    object  
 2   Number    457 non-null    float64 
 3   Position  457 non-null    category
 4   Age       457 non-null    float64 
 5   Height    457 non-null    object  
 6   Weight    457 non-null    float64 
 7   College   457 non-null    object  
 8   Salary    457 non-null    float64 
dtypes: category(1), float64(4), object(4)
memory usage: 32.8+ KB
```

In [65]:

```
nba.Team = nba.Team.astype('category')
```

In [66]:

```
nba.info() # from memory usage: 32.8+ KB to 31.1+ KB, insignifiant
dû au nombre pas treès elevé
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 457 entries, 0 to 456
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype   
---  ------    --------------  -----   
 0   Name      457 non-null    object  
 1   Team      457 non-null    category
 2   Number    457 non-null    float64 
 3   Position  457 non-null    category
 4   Age       457 non-null    float64 
 5   Height    457 non-null    object  
 6   Weight    457 non-null    float64 
 7   College   457 non-null    object  
 8   Salary    457 non-null    float64 
dtypes: category(2), float64(4), object(3)
memory usage: 31.1+ KB
```

Sort a DataFrame with .sort_values() Partie I

- à la différence des Series, il faut specifier les references, ligne_col

In [67]:

```
nba = pd.read_csv('./pandas/nba.csv')
```

In [68]:

```
# 'by' parameter is looking to have a specific column name
nba.sort_values(by = 'Name').head()
```

Out[68]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
152	Aaron Brooks	Chicago Bulls	0.0	PG	31.0	6-0	161.0	Oregon	2250000.0
356	Aaron Gordon	Orlando Magic	0.0	PF	20.0	6-9	220.0	Arizona	4171680.0
328	Aaron Harrison	Charlotte Hornets	9.0	SG	21.0	6-6	210.0	Kentucky	525093.0
404	Adreian Payne	Minnesota Timberwolves	33.0	PF	25.0	6-10	237.0	Michigan State	1938840.0
312	Al Horford	Atlanta Hawks	15.0	C	30.0	6-10	245.0	Florida	12000000.0

In [69]:

```
# 'by' parameter is looking to have a specific column name
# ascending = False  => + grand au + petit, if True, +petit au
+grand
nba.sort_values(by = 'Salary',ascending = False).head()
```

Out[69]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
109	Kobe Bryant	Los Angeles Lakers	24.0	SF	37.0	6-6	212.0	NaN	25000000.0
169	LeBron James	Cleveland Cavaliers	23.0	SF	31.0	6-8	250.0	NaN	22970500.0
33	Carmelo Anthony	New York Knicks	7.0	SF	32.0	6-8	240.0	Syracuse	22875000.0
251	Dwight Howard	Houston Rockets	12.0	C	30.0	6-11	265.0	NaN	22359364.0
339	Chris Bosh	Miami Heat	1.0	PF	32.0	6-11	235.0	Georgia Tech	22192730.0

In [70]:

```
nba.sort_values(by = 'Salary',ascending = False, na_position =
'first').head()
```

Out[70]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN
46	Elton Brand	Philadelphia 76ers	42.0	PF	37.0	6-9	254.0	Duke	NaN
171	Dahntay Jones	Cleveland Cavaliers	30.0	SG	35.0	6-6	225.0	Duke	NaN
264	Jordan Farmar	Memphis Grizzlies	4.0	PG	29.0	6-2	180.0	UCLA	NaN
269	Ray McCallum	Memphis Grizzlies	5.0	PG	24.0	6-3	190.0	Detroit	NaN

Sort a DataFrame with .sort_values() Partie II

In [71]:

```
nba.sort_values(by = ['Team','Name'],ascending = False).head()
```

Out[71]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
379	Ramon Sessions	Washington Wizards	7.0	PG	30.0	6-3	190.0	Nevada	2170465.0
378	Otto Porter Jr.	Washington Wizards	22.0	SF	23.0	6-8	198.0	Georgetown	4662960.0
375	Nene Hilario	Washington Wizards	42.0	C	33.0	6-11	250.0	NaN	13000000.0
376	Markieff Morris	Washington Wizards	5.0	PF	26.0	6-10	245.0	Kansas	8000000.0
381	Marcus Thornton	Washington Wizards	15.0	SF	29.0	6-4	205.0	LSU	200600.0

In [72]:

```
# que faire si nous souhaitons appliquer ascending sur une et
descending sur l'autre?
nba.sort_values(by = ['Team','Name'],ascending = [False,True]
).head()
```

Out[72]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
368	Alan Anderson	Washington Wizards	6.0	SG	33.0	6-6	220.0	Michigan State	4000000.0
369	Bradley Beal	Washington Wizards	3.0	SG	22.0	6-5	207.0	Florida	5694674.0
372	Drew Gooden	Washington Wizards	90.0	PF	34.0	6-10	250.0	Kansas	3300000.0
380	Garrett Temple	Washington Wizards	17.0	SG	30.0	6-6	195.0	LSU	1100602.0
374	JJ Hickson	Washington Wizards	21.0	C	27.0	6-9	242.0	North Carolina State	273038.0

In [ ]:

```

```

.sort_index() method on DataFrame

In [73]:

```
nba.sort_index(axis = 1).head(3)
```

Out[73]:

	Age	College	Height	Name	Number	Position	Salary	Team	Weight
0	25.0	Texas	6-2	Avery Bradley	0.0	PG	7730337.0	Boston Celtics	180.0
1	25.0	Marquette	6-6	Jae Crowder	99.0	SF	6796117.0	Boston Celtics	235.0
2	27.0	Boston University	6-5	John Holland	30.0	SG	NaN	Boston Celtics	205.0

In [74]:

```
nba.sort_index(axis = 0).head(3)
```

Out[74]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	NaN

Rank values with the .rank() method

- this method rank with the position of each value from the overall
- ex : lower rank to the high salary

In [ ]:

```

```

In [75]:

```
nba['Salary'] = nba['Salary'].fillna(0).astype('int')
```

In [76]:

```
nba['Salary'].rank(ascending = False)
```

Out[76]:

```
0       97.0
1      110.0
2      452.5
3      322.0
4      147.0
       ...  
453    241.0
454    383.0
455    214.5
456    367.0
457    452.5
Name: Salary, Length: 458, dtype: float64
```

In [77]:

```
nba['Salary_rank'] = nba['Salary'].rank()
```

In [78]:

```
nba.head()
```

Out[78]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary	Salary_rank
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337	362.0
1	Jae Crowder	Boston Celtics	99.0	SF	25.0	6-6	235.0	Marquette	6796117	349.0
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	0	6.5
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640	137.0
4	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000	312.0

In [79]:

```
nba.sort_values('Salary_rank', ascending = False)
```

Out[79]:

	Name	Team	Number	Position	Age	Height	Weight	College	Salary	Salar

	Name	Team	Number	Position	Age	Height	Weight	College	Salary	Salar
109	Kobe Bryant	Los Angeles Lakers	24.0	SF	37.0	6-6	212.0	NaN	25000000	
169	LeBron James	Cleveland Cavaliers	23.0	SF	31.0	6-8	250.0	NaN	22970500	
33	Carmelo Anthony	New York Knicks	7.0	SF	32.0	6-8	240.0	Syracuse	22875000	
251	Dwight Howard	Houston Rockets	12.0	C	30.0	6-11	265.0	NaN	22359364	
339	Chris Bosh	Miami Heat	1.0	PF	32.0	6-11	235.0	Georgia Tech	22192730	
...	...	...	...	...	...	...	...	...	...	
269	Ray McCallum	Memphis Grizzlies	5.0	PG	24.0	6-3	190.0	Detroit	0	
409	Greg Smith	Minnesota Timberwolves	4.0	PF	25.0	6-10	250.0	Fresno State	0	
2	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston University	0	
264	Jordan Farmar	Memphis Grizzlies	4.0	PG	29.0	6-2	180.0	UCLA	0	
457	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0	

458 rows × 10 columns

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```