Pandas
Manipulation des données

Par

OUHAB Abdelkrim

Afficher dataframe

- Pour l'affichage d'un dataframe, on utilise df, print(df) ou display(df)
- df.head(): affiche les cinq premières lignes
- df.head(n): affiche les n premières lignes

In [1]:
```
import pandas as pd
dataset_path = 'd:/data/unemployment.csv'
df = pd.read_csv(dataset_path)
df.head()
```

Out[1]:

	country	unemployment_rate
0	Marshall Islands	36.0
1	South Africa	33.6
2	Kiribati	30.6
3	Kosovo	30.5
4	American Samoa	29.8

Informations sur le dataframe

- df.shape: nombre de lignes et colonnes
- len(df): nombre de lignes
- df.info(): Informations générales sur le dataframe

```
In [1]: import pandas as pd
        df = pd.read_csv('d:/data/nba.csv')
        print(df.shape)
        print(len(df))
        print(df.info())
```

```
(458, 9)
458
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 458 entries, 0 to 457
Data columns (total 9 columns):
Name        457 non-null object
Team        457 non-null object
Number      457 non-null float64
Position    457 non-null object
Age         457 non-null float64
Height      457 non-null object
Weight      457 non-null float64
College     373 non-null object
Salary      446 non-null float64
dtypes: float64(4), object(5)
memory usage: 23.3+ KB
None
```

Index pour Dataframe

- Un index sert de référence pour chaque ligne du Dataframe.
- Par défaut un index est numérique (0, 1, ….)
- On peut définir les valeurs de colonnes spécifiques comme index
- Deux méthodes pour définir un index:
  - Utiliser l'attribut *index_col* de la méthode *pandas.read_csv()*
  - Utiliser la méthode Dataframe.set_index()

Index pour Dataframe

- *DataFrame.set_index(keys, drop=True, append=False, inplace=False, verify_integrity=False)*
  - **keys :** Un nom de colonne ou une liste de noms de colonnes à définir comme index.
  - **drop** : la colonne spécifiée dans *keys* sera supprimée du DataFrame. Si la valeur est False, elle sera conservée comme colonne normale.
  - **append** : la colonne sera ajoutée à l'index existant, créant ainsi un index à plusieurs niveaux.
  - **inplace** : Si la valeur est True, modifier le DataFrame d'origine sans en renvoyer un nouveau.
  - **verify_integrity** : Si True, vérifie la présence de valeurs d'index dupliquées.

Index pour Dataframe

In [19]:
```
import pandas as pd
df = pd.read_csv("d:/data/notes.csv")
display(df.head(1))

df.set_index("Nom", inplace=True)
display(df.head(1))

dff = pd.read_csv("d:/data/notes.csv", index_col='Nom')
display(dff.head(1))
```

	Nom	Maths	Python2	Comptabilité	Statistiques
0	Sylvie	11.0	18.0	15.0	18.0

Nom	Maths	Python2	Comptabilité	Statistiques
Sylvie	11.0	18.0	15.0	18.0

Nom	Maths	Python2	Comptabilité	Statistiques
Sylvie	11.0	18.0	15.0	18.0

Sélection des données

- La sélection de données consiste à extraire (*filtrer*) des sous-ensembles de données spécifiques à partir d'un dataframe
- On peut sélectionner des lignes, des colonnes et/ou des cellules individuelles
- Il existe principalement trois méthodes pour la sélection des données dans Pandas :
  - l'opérateur d'indexation [] pour la sélection des colonnes
  - les méthodes d'accès .loc() et .iloc() utilisées pour la sélection des lignes et colonnes

Sélection de colonnes: opérateur d'indexation[]

- Dataframe[liste de colonnes à selectionner]

In [22]:
```
import pandas as pd
dataset_path = 'd:/data/nba.csv'
df = pd.read_csv(dataset_path)
df1 = df[["Name","Age","Salary"]]
display(df.head(2))
display(df1.head(2))
```

	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

	Name	Age	Salary
0	Avery Bradley	25.0	7730337.0
1	Jae Crowder	25.0	6796117.0

Sélection de lignes: .loc[]

- Définir *un index* pour le dataframe
- Utiliser Dataframe.loc[["row1", "row2“, …]] ou ["row1“: "row2“]
- “,” veut dire “et”
- “:” veut dire “jusqu’à”

In [16]:
```
import pandas as pd
dataset_path = 'd:/data/nba.csv'
# Définir la colonne Name comme index
df = pd.read_csv(dataset_path, index_col="Name")
# Sélection de deux personnes
rows = df.loc[["Avery Bradley", "R.J. Hunter"]]
display(rows)
```

Name	Team	Number	Position	Age	Height	Weight	College	Salary
Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0

Sélection de lignes et colonnes: .loc[]

- Définir *un index* pour le dataframe
- Utiliser Dataframe.loc[["row1", "row2“, …], ["col1", "col2", …]]

In [18]:
```
import pandas as pd
dataset_path = 'd:/data/nba.csv'
# Définir la colonne Name comme index
df = pd.read_csv(dataset_path, index_col="Name")
selection = df.loc[["Avery Bradley", "R.J. Hunter"], ["Team", "Number", "Position"]]
print(selection)
```

```
                      Team  Number Position
Name
Avery Bradley  Boston Celtics     0.0       PG
R.J. Hunter    Boston Celtics    28.0       SG
```

Sélection d’une cellule: .loc[]

- Définir un index pour le dataframe
- Utiliser Dataframe.loc["row", “col“]

In [31]:
```
import pandas as pd
df = pd.read_csv('d:/data/nba.csv', index_col='Name')
value = df.loc['Jahlil Okafor', 'Salary']
print("Salaire de Jahlil Okafor est: ", value)
```

```
Salaire de Jahlil Okafor est:  4582680.0
```

Sélection de lignes et colonnes: .iloc[]

- Sélection *basée sur position* de lignes et/ou colonnes

In [26]:
```
import pandas as pd
data = pd.read_csv('d:/data/nba.csv')
rows = data.iloc[[3, 5, 7]]
display(rows)
selection = data.iloc[[3, 4], [1, 2]]
display(selection)
```

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
3	R.J. Hunter	Boston Celtics	28.0	SG	22.0	6-5	185.0	Georgia State	1148640.0
5	Amir Johnson	Boston Celtics	90.0	PF	29.0	6-9	240.0	NaN	12000000.0
7	Kelly Olynyk	Boston Celtics	41.0	C	25.0	7-0	238.0	Gonzaga	2165160.0

	Team	Number
3	Boston Celtics	28.0
4	Boston Celtics	8.0

Sélection en utilisant une condition

- Plusieurs méthodes existent (loc, NumPy, query)
- Dataframe.query(condition)

```
In [3]: import pandas as pd
        data = pd.read_csv("d:/data/nba.csv")
        selection = data.query("Age < 20")
        display(selection)
```

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
122	Devin Booker	Phoenix Suns	1.0	SG	19.0	6-6	206.0	Kentucky	2127840.0
226	Rashad Vaughn	Milwaukee Bucks	20.0	SG	19.0	6-6	202.0	UNLV	1733040.0

Sélection en utilisant plusieurs conditions

- Plusieurs méthodes existent (loc, NumPy, query)
- Dataframe.query(condition)

In [28]:
```
import pandas as pd
data = pd.read_csv('d:/data/nba.csv')
selection = data.query("Age < 25 and College == 'Duke'")
display(selection)
```

	Name	Team	Number	Position	Age	Height	Weight	College	Salary
56	Jahlil Okafor	Philadelphia 76ers	8.0	C	20.0	6-11	275.0	Duke	4582680.0
104	Austin Rivers	Los Angeles Clippers	25.0	PG	23.0	6-4	200.0	Duke	3110796.0
168	Kyrie Irving	Cleveland Cavaliers	2.0	PG	24.0	6-3	193.0	Duke	16407501.0
223	Jabari Parker	Milwaukee Bucks	12.0	PF	21.0	6-8	250.0	Duke	5152440.0
352	Justise Winslow	Miami Heat	20.0	SF	20.0	6-7	225.0	Duke	2481720.0
401	Tyus Jones	Minnesota Timberwolves	1.0	PG	20.0	6-2	195.0	Duke	1282080.0
449	Rodney Hood	Utah Jazz	5.0	SG	23.0	6-8	206.0	Duke	1348440.0

Trier un DataFrame

- *DataFrame.sort_values(by, axis=0, ascending=True, inplace=False, kind='quicksort', na_position='last')*

- ***by:*** *Nom(s) de colonne(s) utilisée(s) pour le tri (unique ou liste)*
- ***axis:*** *0 ou « index » pour trier les lignes ; 1 ou « columns » pour trier les colonnes*
- ***ascending:*** *booléen ; True pour ascendant, False pour descendant*
- ***inplace:*** *booléen ; si True, modifie le DataFrame d'origine*
- ***kind:*** *Algorithme de tri : « quicksort », « mergesort » ou « heapsort »*
- ***na_position:*** *« first » ou « last » ; définit la position des valeurs NaN*

Trier un DataFrame selon une colonne

In [4]:
```
import pandas as pd
data = {'nom': ['Alice', 'Bob', 'Charlie', 'David'],
        'age': [45, 30, 60, 40],
        'score': [85, 90, 95, 80]}
df = pd.DataFrame(data)
sorted_df = df.sort_values(by=['score'])
display(sorted_df)
```

	nom	age	score
3	David	40	80
0	Alice	45	85
1	Bob	30	90
2	Charlie	60	95

Trier un DataFrame selon plusieurs colonnes

In [37]:
```
import pandas as pd
data = {'Name': ['Alice', 'Bob', 'Charlie', 'David'],
        'Age': [25, 30, 35, 40],
        'Score': [85, 90, 95, 80]}
df = pd.DataFrame(data)
sorted_df = df.sort_values(by=['Age', 'Score'])
display(sorted_df)
```

	Name	Age	Score
0	Alice	25	85
1	Bob	30	90
2	Charlie	35	95
3	David	40	80