Pandas
Analyse de données

Par

OUHAB Abdelkrim

Afficher dataframe

- 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.columns: liste des colonnes
- df.info(): Informations générales sur le dataframe

In [6]:
```
import pandas as pd
df = pd.read_csv("d:/data/nba_fr.csv")
print(df.shape)
print(len(df))
print(df.columns)
print(df.info())
```

```
(458, 9)
458
Index(['Nom', ' Equipe', ' Numero', ' Poste', ' Age', ' Taille', ' Poids',
       ' Universite', ' Salaire'],
      dtype='object')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 458 entries, 0 to 457
Data columns (total 9 columns):
Nom            457 non-null object
 Equipe        457 non-null object
 Numero        457 non-null float64
 Poste         457 non-null object
 Age           457 non-null float64
 Taille        457 non-null object
 Poids         457 non-null float64
 Universite    373 non-null object
 Salaire       446 non-null float64
dtypes: float64(4), object(5)
memory usage: 23.3+ KB
None
```

Convertir une colonne en liste

- Nom_liste = df[‘nom_colonne'].tolist()

In [2]:

```
1 import pandas as pd
2 df= pd.read_csv("d:/data/ventes.csv")
3 display(df.head(2))
4 Produit_liste = df['Produit'].tolist()
5 print(Produit_liste)
```

	Produit	Categorie	Quantite	Montant
0	Carottes	Legume	9	250
1	Brocoli	Legume	5	239

```
['Carottes', 'Brocoli', 'Banane', 'Haricots', 'Brocoli', 'Banane', 'Spaghetti', 'Macaronis', 'Nouilles', 'Coca-cola', 'Sprite
', 'Miranda', 'Carottes', 'Brocoli', 'Banane', 'Haricots', 'Brocoli', 'Banane', 'Spaghetti', 'Macaronis', 'Nouilles', 'Coca-co
la', 'Sprite', 'Miranda']
```

Informations statistiques sur le dataframe

❑ *DataFrame.describe(percentiles=None, include=None, exclude=None)*

- **percentiles** : Liste de nombres compris entre 0 et 1 indiquant les percentiles à renvoyer. La valeur par défaut est None, ce qui renvoie les 25e, 50e et 75e percentiles.
- **include** : Liste des types de données à inclure dans le résumé, comme int, float, object pour les chaînes de caractères. La valeur par défaut None signifie que tous les types numériques sont inclus.
- **exclude** : Liste des types de données à exclure du résumé. La valeur par défaut « None » signifie qu'aucun type n'est exclu.

```
In [8]: import pandas as pd
        df = pd.read_csv("d:/data/nba_fr.csv")
        display(df.head(1))
        df.describe()
```

	Nom	Equipe	Numero	Poste	Age	Taille	Poids	Universite	Salaire
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0

Out[8]:

	Numero	Age	Poids	Salaire
count	457.000000	457.000000	457.000000	4.460000e+02
mean	17.678337	26.938731	221.522976	4.842684e+06
std	15.966090	4.404016	26.368343	5.229238e+06
min	0.000000	19.000000	161.000000	3.088800e+04
25%	5.000000	24.000000	200.000000	1.044792e+06
50%	13.000000	26.000000	220.000000	2.839073e+06
75%	25.000000	30.000000	240.000000	6.500000e+06
max	99.000000	40.000000	307.000000	2.500000e+07

Fonctions statistiques: sum()

- *DataFrame.sum(axis=None, skipna=None, level=None, numeric_only=None, min_count=0, **kwargs)*
  - **axis (index (0), columns (1))** spécifie s'il faut additionner le long des lignes (index) ou des colonnes.
  - **skipna(bool, par défaut True)** Si True, exclut les valeurs NaN/null lors du calcul de la somme.
  - **level (int ou nom de niveau, par défaut None) :** Si l'axe est un MultiIndex (hiérarchique), somme le long d'un niveau spécifique, en se repliant sur une série.
  - **numeric_only (booléen, valeur par défaut None) :** si la valeur est True, seules les colonnes numériques sont prises en compte.
  - **min_count (int, valeur par défaut 0) :** Nombre minimal de valeurs valides requis pour effectuer la somme. Si le nombre de valeurs non manquantes est inférieur à min_count, le résultat sera NaN.

Fonctions statistiques: sum()

In [42]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
display(df.head())
somme_lignes=df.sum(numeric_only=True)
somme_colonnes=df.sum(axis=1, numeric_only=True)
print("----- somme lignes ------------")
display(somme_lignes)
print("----- somme colonnes ------------")
display(somme_colonnes.head())
df['Somme']=somme_colonnes
print("----- ajout de somme colonnes à df ------------")
display(df.head())
```

	Produit	Categorie	Quantite	Montant
0	Carrots	Vegetable	8	270
1	Broccoli	Vegetable	5	239
2	Banana	Fruit	4	384
3	Beans	Vegetable	5	626
4	Broccoli	Vegetable	11	62

```
----- somme lignes ------------

Quantite     286
Montant     4688
dtype: int64

----- somme colonnes ------------

0    278
1    244
2    388
3    631
4     73
dtype: int64

----- ajout de somme colonnes à df ------------
```

	Produit	Categorie	Quantite	Montant	Somme
0	Carrots	Vegetable	8	270	278
1	Broccoli	Vegetable	5	239	244
2	Banana	Fruit	4	384	388
3	Beans	Vegetable	5	626	631
4	Broccoli	Vegetable	11	62	73

Fonctions statistiques
moyenne, maximum, minimum, médiane, variance, écart type

- *DataFrame.mean(axis=0, skipna=True, level=None, numeric_only=False, **kwargs)*
- *DataFrame.max(axis=None, skipna=None, level=None, numeric_only=None, **kwargs)*
- *DataFrame.min(axis=None, skipna=None, level=None, numeric_only=None, **kwargs)*
- *DataFrame.median(axis=None, skipna=None, level=None, numeric_only=None, **kwargs)*
- *DataFrame.var(*, axis=0, skipna=True, ddof=1, numeric_only=False, **kwargs)*
- *DataFrame.std(axis=None, skipna=None, level=None, ddof=1, numeric_only=None, **kwargs)*
  - ddof (*int): Delta degrés de liberté. Le diviseur utilisé dans les calculs est*

    N - ddof, où N représente le nombre d'éléments.

Fonctions statistiques appliquées aux lignes

In [44]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
moyenne_lignes=df.mean(numeric_only=True)
max_lignes=df.max(numeric_only=True)
min_lignes=df.min(numeric_only=True)
median_lignes=df.median(numeric_only=True)
var_lignes=df.var(numeric_only=True)
std_lignes=df.std(numeric_only=True)
print("----- moyenne lignes ------------")
display(moyenne_lignes)
print("----- max lignes ------------")
display(max_lignes)
print("----- min lignes ------------")
display(min_lignes)
print("----- mediane lignes ------------")
display(median_lignes)
print("----- variance lignes ------------")
display(var_lignes)
print("----- ecart type lignes ------------")
display(std_lignes)
```

```
----- moyenne lignes ------------

Quantite     11.916667
Montant     195.333333
dtype: float64
```

```
----- max lignes ------------

Quantite     20
Montant     626
dtype: int64

----- min lignes ------------

Quantite     4
Montant     25
dtype: int64

----- mediane lignes ------------

Quantite     12.5
Montant     157.5
dtype: float64

----- variance lignes ------------

Quantite       32.427536
Montant     28134.144928
dtype: float64

----- ecart type lignes ------------

Quantite      5.694518
Montant     167.732361
dtype: float64
```

Fonctions statistiques
appliquées aux colonnes

In [54]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
somme_colonnes=df.sum(axis=1, numeric_only=True)
moyenne_colonnes=df.mean(axis=1, numeric_only=True)
max_colonnes=df.max(axis=1,numeric_only=True)
min_colonnes=df.min(axis=1,numeric_only=True)
median_colonnes=df.median(axis=1,numeric_only=True)
var_colonnes=df.var(axis=1,numeric_only=True)
std_colonnes=df.std(axis=1,numeric_only=True)
df['somme'] = somme_colonnes
df['moyenne'] = moyenne_colonnes
df['maximum'] = max_colonnes
df['minimum'] = min_colonnes
df['mediane'] = median_colonnes
df['variance'] = var_colonnes
df['ecart_type'] = std_colonnes
display(df.head())
```

	Produit	Categorie	Quantite	Montant	somme	moyenne	maximum	minimum	mediane	variance	ecart_type
0	Carrots	Vegetable	8	270	278	139.0	270	8	139.0	34322.0	185.261977
1	Broccoli	Vegetable	5	239	244	122.0	239	5	122.0	27378.0	165.462987
2	Banana	Fruit	4	384	388	194.0	384	4	194.0	72200.0	268.700577
3	Beans	Vegetable	5	626	631	315.5	626	5	315.5	192820.5	439.113311
4	Broccoli	Vegetable	11	62	73	36.5	62	11	36.5	1300.5	36.062446

Méthode Dataframe.pivot_table()

- La méthode Dataframe.pivot_table() permet de créer un tableau croisé dynamique pour synthétiser et agréger les données du dataframe
- *DataFrame.pivot_table(values=None, index=None, columns=None, aggfunc='mean', fill_value=None, margins=False, dropna=True)*
  - **values** : Colonnes à agréger.
  - **index** : Colonnes à utiliser comme nouvel index de ligne.
  - **columns** : Colonnes à utiliser comme nouveaux en-têtes de colonnes.
  - **aggfunc** : Fonctions d'agrégation telles que la moyenne, la somme, le nombre, etc. Par défaut, il s'agit de la moyenne.
  - **fill_value** : Valeur permettant de remplacer les données manquantes.
  - **margins** : Indique s'il faut ajouter les totaux ; la valeur par défaut est false.
  - **dropna** : Indique s'il faut exclure les valeurs manquantes du DataFrame, la valeur par défaut est True.

Méthode Dataframe.pivot_table()
Exemple1: Montant des ventes par produit

In [4]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
display(df)
# Montant des ventes par produit
pivot = df.pivot_table(index=['Produit'], values=['Montant'], aggfunc='sum')
display(pivot)
```

	Produit	Categorie	Quantite	Montant
0	Carrots	Vegetable	8	270
1	Broccoli	Vegetable	5	239
2	Banana	Fruit	4	384
3	Beans	Vegetable	5	626
4	Broccoli	Vegetable	11	62
5	Banana	Fruit	8	90

	Montant
Produit	
Banana	474
Beans	626
Broccoli	301
Carrots	270

Méthode Dataframe.pivot_table()
Exemple2: Montant des ventes par produit

In [5]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
display(df)
# Montant des ventes par produit
pivot = df.pivot_table(index=['Produit'], values=['Montant'], aggfunc='sum', margins='True')
display(pivot)
```

	Produit	Categorie	Quantite	Montant
0	Carrots	Vegetable	8	270
1	Broccoli	Vegetable	5	239
2	Banana	Fruit	4	384
3	Beans	Vegetable	5	626
4	Broccoli	Vegetable	11	62
5	Banana	Fruit	8	90

	Montant
Produit	
Banana	474
Beans	626
Broccoli	301
Carrots	270
All	1671

Méthode Dataframe.pivot_table()
Exemple3: Montant des ventes par catégorie

In [6]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
display(df)
# Montant des ventes par produit
pivot = df.pivot_table(index=['Categorie'], values=['Montant'], aggfunc='sum', margins='True')
display(pivot)
```

	Produit	Categorie	Quantite	Montant
0	Carrots	Vegetable	8	270
1	Broccoli	Vegetable	5	239
2	Banana	Fruit	4	384
3	Beans	Vegetable	5	626
4	Broccoli	Vegetable	11	62
5	Banana	Fruit	8	90

	Montant
Categorie	
Fruit	474
Vegetable	1197
All	1671

Méthode Dataframe.pivot_table()
Exemple4: Montant des ventes par catégorie et produit

In [9]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
display(df)
# Montant des ventes par produit
pivot = df.pivot_table(index=['Categorie', 'Produit'], values=['Montant'], aggfunc='sum')
display(pivot)
```

	Produit	Categorie	Quantite	Montant
0	Carrots	Vegetable	8	270
1	Broccoli	Vegetable	5	239
2	Banana	Fruit	4	384
3	Beans	Vegetable	5	626
4	Broccoli	Vegetable	11	62
5	Banana	Fruit	8	90

		Montant
Categorie	Produit	
Fruit	Banana	474
Vegetable	Beans	626
	Broccoli	301
	Carrots	270

Méthode Dataframe.pivot_table()

Exemple5: moyenne, médiane et minimum des ventes par catégorie

In [10]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
display(df)
# Montant des ventes par produit
pivot = df.pivot_table(index=['Categorie'], values=['Montant'], aggfunc={'median', 'mean', 'min'})
display(pivot)
```

	Produit	Categorie	Quantite	Montant
0	Carrots	Vegetable	8	270
1	Broccoli	Vegetable	5	239
2	Banana	Fruit	4	384
3	Beans	Vegetable	5	626
4	Broccoli	Vegetable	11	62
5	Banana	Fruit	8	90

	Montant		
	mean	median	min
Categorie			
Fruit	237.00	237.0	90.0
Vegetable	299.25	254.5	62.0

Méthode Dataframe.pivot_table()

Exemple5: moyenne, médiane et minimum des ventes par produit

In [1]:
```
import pandas as pd
df = pd.read_csv("d:/data/ventes.csv")
display(df)
# Montant des ventes par produit
pivot = df.pivot_table(index=['Produit'], values=['Montant'], aggfunc={'median', 'mean', 'min'})
display(pivot)
```

	Produit	Categorie	Quantite	Montant
0	Carrots	Vegetable	8	270
1	Broccoli	Vegetable	5	239
2	Banana	Fruit	4	384
3	Beans	Vegetable	5	626
4	Broccoli	Vegetable	11	62
5	Banana	Fruit	8	90

	Montant		
	mean	median	min
Produit			
Banana	237.0	237.0	90.0
Beans	626.0	626.0	626.0
Broccoli	150.5	150.5	62.0
Carrots	270.0	270.0	270.0

Méthode Dataframe. groupby()

- La méthode pivot_table() est utilisée pour construire un tableau récapitulatif similaire à Excel
- La méthode groupby() permet une flexibilité maximale et des agrégations personnalisées complexes sur des données volumineuses par rapport à la méthode pivot_table()
- La méthode groupby() comprend trois étapes principales : groupage (*division en groupes)*, *application d'une fonction* et *combinaison*.
- Le groupage consiste à diviser le DataFrame en groupes selon certains critères en se basant sur une ou plusieurs colonnes
- La deuxième étape permet d'appliquer une fonction pour chaque groupe. Différentes fonctions peuvent être appliquées comme:
  - Agrégation : Calcul de statistiques descriptives (par exemple, somme, moyenne, nombre d'éléments) pour chaque groupe
  - Transformation : Modification des valeurs au sein de chaque groupe
  - Filtrage : Conservation ou suppression de groupes selon certaines conditions
- La combinaison : les résultats de la fonction appliquée sont combinés dans un nouveau DataFrame ou une nouvelle Serie

Méthode Dataframe. groupby()

- *DataFrame.groupby(by=None, axis=0, level=None, as_index=True, sort=True, group_keys=True, observed=False, dropna=True)*
  - **by** *: Paramètre obligatoire pour spécifier la ou les colonnes selon lesquelles effectuer le regroupement.*
  - **axis** *: Facultatif, spécifie l'axe de regroupement (par défaut : 0 pour les lignes).*
  - **level** *: Facultatif, utilisé pour le regroupement selon un niveau spécifique dans un MultiIndex.*
  - **as_index** *: Facultatif, indique si les étiquettes de groupe doivent être utilisées comme index (par défaut : True).*
  - **sort** *: Facultatif, indique si les clés de groupe doivent être triées (par défaut : True).*
  - **group_keys** *: Facultatif, indique si les clés de groupe doivent être ajoutées à l'index (par défaut : True).*
  - **dropna** *: Facultatif, indique si les lignes/colonnes contenant des valeurs NULL doivent être incluses (par défaut : True).*

Méthode Dataframe. groupby()
Exemple1: Regroupement par Equipe

In [36]:
```
import pandas as pd
df = pd.read_csv("d:/data/nba_fr.csv")
df_par_equipe = df.groupby('Equipe')
# Afficher le nom du groupe et le groupe
for name, group in df_par_equipe:
    print("Nom du groupe:",name)
    display(group.head(2))
```

```
Nom du groupe: Atlanta Hawks
```

	Nom	Equipe	Numero	Poste	Age	Taille	Poids	Universite	Salaire
309	Kent Bazemore	Atlanta Hawks	24.0	SF	26.0	6-5	201.0	Old Dominion	2000000.0
310	Tim Hardaway Jr.	Atlanta Hawks	10.0	SG	24.0	6-6	205.0	Michigan	1304520.0

```
Nom du groupe: Boston Celtics
```

	Nom	Equipe	Numero	Poste	Age	Taille	Poids	Universite	Salaire
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

Méthode Dataframe. groupby()
Exemple2: Regroupement par Equipe et Poste

```
In [39]: import pandas as pd
         df = pd.read_csv("d:/data/nba_fr.csv")
         df_par_equipe = df.groupby(['Equipe','Poste'])
         # Afficher le nom du groupe et le groupe
         for name, group in df_par_equipe:
             print("Nom du groupe:",name)
             display(group.head(2))
```

```
Nom du groupe: ('Atlanta Hawks', 'C')
```

	Nom	Equipe	Numero	Poste	Age	Taille	Poids	Universite	Salaire
312	Al Horford	Atlanta Hawks	15.0	C	30.0	6-10	245.0	Florida	12000000.0
321	Tiago Splitter	Atlanta Hawks	11.0	C	31.0	6-11	245.0	NaN	9756250.0

```
Nom du groupe: ('Atlanta Hawks', 'PF')
```

	Nom	Equipe	Numero	Poste	Age	Taille	Poids	Universite	Salaire
313	Kris Humphries	Atlanta Hawks	43.0	PF	31.0	6-9	235.0	Minnesota	1000000.0
315	Paul Millsap	Atlanta Hawks	4.0	PF	31.0	6-8	246.0	Louisiana Tech	18671659.0

Méthode Dataframe. groupby()
Exemple3: Somme des salaire par Equipe

In [46]:
```
import pandas as pd
df = pd.read_csv("d:/data/nba_fr.csv")
df_somme = df.groupby('Equipe')['Salaire'].sum()
display(df_somme)
```

Equipe	
Atlanta Hawks	72902950.0
Boston Celtics	58541068.0
Brooklyn Nets	52528475.0
Charlotte Hornets	78340920.0
Chicago Bulls	86783378.0
Cleveland Cavaliers	106988689.0
Dallas Mavericks	71198732.0
Denver Nuggets	60121930.0
Detroit Pistons	67168263.0
Golden State Warriors	88868997.0
Houston Rockets	75283021.0
Indiana Pacers	66751826.0
Los Angeles Clippers	94854640.0
Los Angeles Lakers	71770431.0
Memphis Grizzlies	76550880.0
Miami Heat	82515673.0
Milwaukee Bucks	69603517.0
Minnesota Timberwolves	59709697.0
New Orleans Pelicans	82750774.0

Méthode Dataframe. groupby()
Exemple4: Agrégations multiples

❑ Après le groupage, plusieurs fonctions peuvent être appliquées (somme, moyenne, nombre, etc.)

```
In [3]: import pandas as pd
        df = pd.read_csv("d:/data/nba_fr.csv")
        aggregated_data = df.groupby(['Equipe','Poste']).agg({
            'Salaire': ['sum', 'mean'],
            'Nom': 'count'
        })
        display(aggregated_data)
```

		Salaire		Nom
		sum	mean	count
Equipe	Poste			
Atlanta Hawks	C	22756250.0	7.585417e+06	3
	PF	23952268.0	5.988067e+06	4
	PG	9763400.0	4.881700e+06	2
	SF	6000000.0	3.000000e+06	2
	SG	10431032.0	2.607758e+06	4
...	...	...	...	...
Washington Wizards	C	24490429.0	8.163476e+06	3
	PF	11300000.0	5.650000e+06	2
	PG	18022415.0	9.011208e+06	2
	SF	11158800.0	2.789700e+06	4

Méthode Dataframe. groupby()
Exemple5: Agrégations multiples

- ❑ Après le groupage, plusieurs fonctions peuvent être appliquées (somme, moyenne, nombre, etc.)
- ❑*On peut aussi nommer les colonnes*

In [2]:
```
import pandas as pd
df = pd.read_csv("d:/data/nba_fr.csv")
aggregated_data = df.groupby(['Equipe', 'Poste']).agg(
    total_salaire=('Salaire', 'sum'),
    moyenne_salaire=('Salaire', 'mean'),
    nombre_joueurs=('Nom', 'count')
)

display(aggregated_data)
```

		total_salaire	moyenne_salaire	nombre_joueurs
Equipe	Poste			
Atlanta Hawks	C	22756250.0	7.585417e+06	3
	PF	23952268.0	5.988067e+06	4
	PG	9763400.0	4.881700e+06	2
	SF	6000000.0	3.000000e+06	2
	SG	10431032.0	2.607758e+06	4
...	...	...	...	...
Washington Wizards	C	24490429.0	8.163476e+06	3
	PF	11300000.0	5.650000e+06	2
	PG	18022415.0	9.011208e+06	2