Royaume du Maroc
Université Mohammed Premier
École Supérieure De Technologie-Oujda
Département : Informatique
Filière : LP Informatique Décisionnelle

Analyse De Données

Compte Rendu Des Travaux Pratiques Sur Python Dans Google Colab ‘TP2/3/4’

Réalisé par : HNIOUA Abdessamad

Encadré par : M. Mounir Grari

Année Universitaire : 2023/2024

Introduction:

L'exploration et l'analyse de données sont des éléments cruciaux dans le domaine de l'informatique décisionnelle. Pour ce faire, l'utilisation de bibliothèques telles que Matplotlib, NumPy et Pandas en Python offre des outils puissants pour visualiser, manipuler et comprendre les données. Matplotlib permet la création de graphiques informatifs, tandis que NumPy simplifie la manipulation efficace des tableaux de données, et Pandas offre une structure de données flexible pour l'analyse et la manipulation des données tabulaires.

Dans ce compte rendu des travaux pratiques, nous explorerons comment ces bibliothèques peuvent être mises en œuvre pour analyser les variations de température, manipuler des tableaux de données, et effectuer des opérations statistiques et descriptives sur des ensembles de données. Ces outils sont indispensables pour extraire des informations significatives à partir des données, fournissant ainsi une base solide pour la prise de décision et l'élaboration de stratégies dans divers domaines.

I. Analyse et Visualisation de Données avec Python sur Matplotlib.

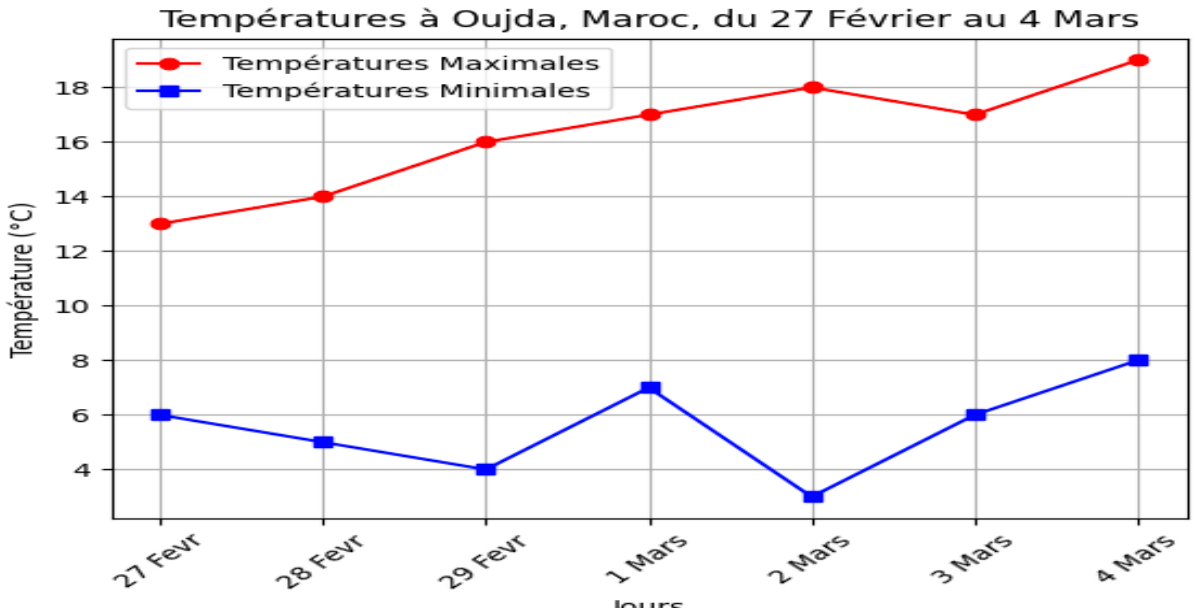
Matplotlib permet aux utilisateurs de créer une grande variété de graphiques scientifiques et de visualiser efficacement leurs données, elle offre la possibilité de générer des graphiques dynamiques avec des fonctionnalités telles que le zoom et la sauvegarde. Les formats de sauvegarde incluent des options matricielles comme PNG et JPEG, ainsi que des formats vectoriels tels que PDF et SVG.

Matplotlib est une bibliothèque essentielle dans le domaine de l'analyse de données, offrant aux analystes et aux scientifiques des données les outils nécessaires pour explorer, visualiser et communiquer les insights tirés des données.

- Nous allons utiliser la bibliothèque Matplotlib en Python pour visualiser les variations de température à Oujda, au Maroc, entre le 27 février et le 4 mars. Cette analyse nous permettra de mieux comprendre les tendances météorologiques dans la région de L'Oriental pendant cette semaine.

```
import matplotlib.pyplot as plt
# Données des températures maximales et minimales
temperaturesHaute = [13, 14, 16, 17, 18, 17, 19]
temperaturesFible = [6, 5, 4, 7, 3, 6, 8]
# Jours correspondants
```

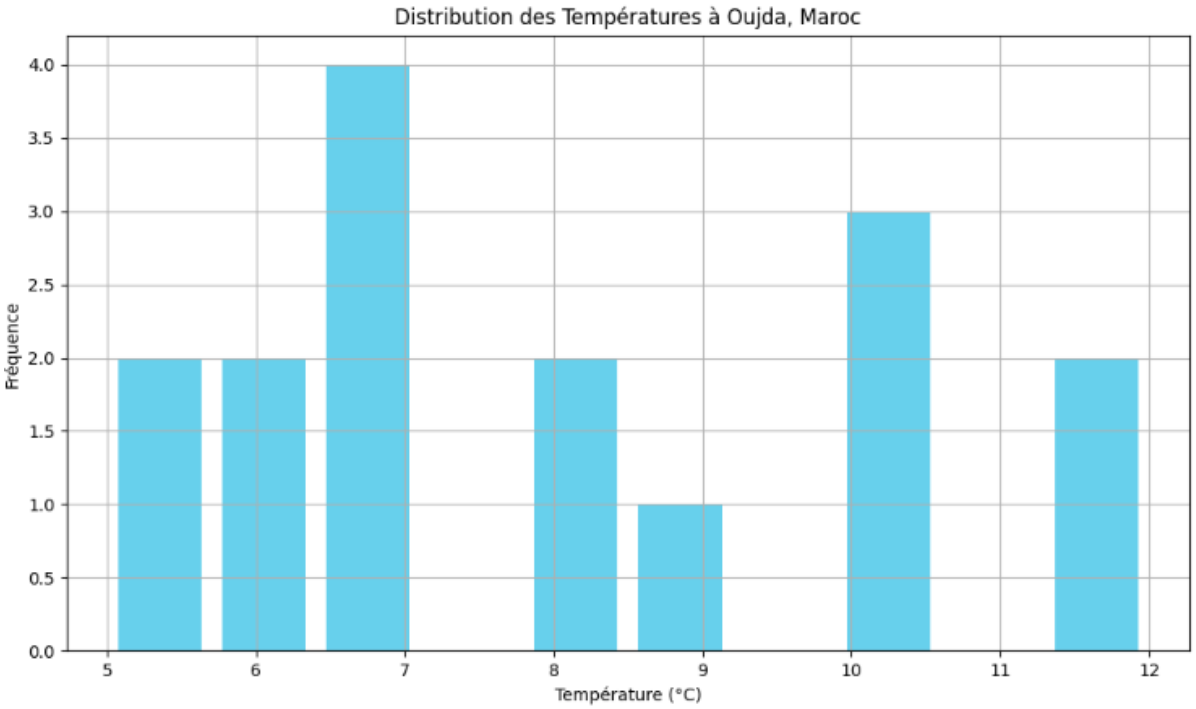
```
jours = ['27 Fevr', '28 Fevr', '29 Fevr', '1 Mars', '2 Mars', '3 Mars',
'4 Mars']
# Tracé des températures maximales
plt.plot(jours, temperaturesHaute, label='Températures Maximales',
color='red', marker='o')
# Tracé des températures minimales
plt.plot(jours, temperaturesFible, label='Températures Minimales',
color='blue', marker='s')
# Ajout des étiquettes et du titre
plt.xlabel('Jours')
plt.ylabel('Température (°C)')
plt.title("Températures à Oujda, Maroc, du 27 Février au 4 Mars")
plt.xticks(rotation=45)
plt.legend()
# Affichage du graphique
plt.grid(True)  # Activation de la grille
plt.show()
```



- En analysant l'histogramme des températures à Oujda, Maroc, nous pouvons déterminer la température la plus dominante entre "Météo Oujda aujourd'hui, 27 février" et "Météo Oujda demain, 28 février". La barre la plus haute sur l'histogramme indique la température la plus fréquente, fournissant ainsi des insights sur les conditions météorologiques prévalant dans la région. Cette approche visuelle nous permet de comprendre rapidement la répartition des températures et d'anticiper les tendances météorologiques avec plus de précision.

```
import matplotlib.pyplot as plt
# Températures à Oujda pour chaque heure de la journée (exemple fictif)
temperatures = [10, 10, 9, 8, 7, 7, 7, 7, 6, 5, 5, 10, 12, 12, 8, 6]
```

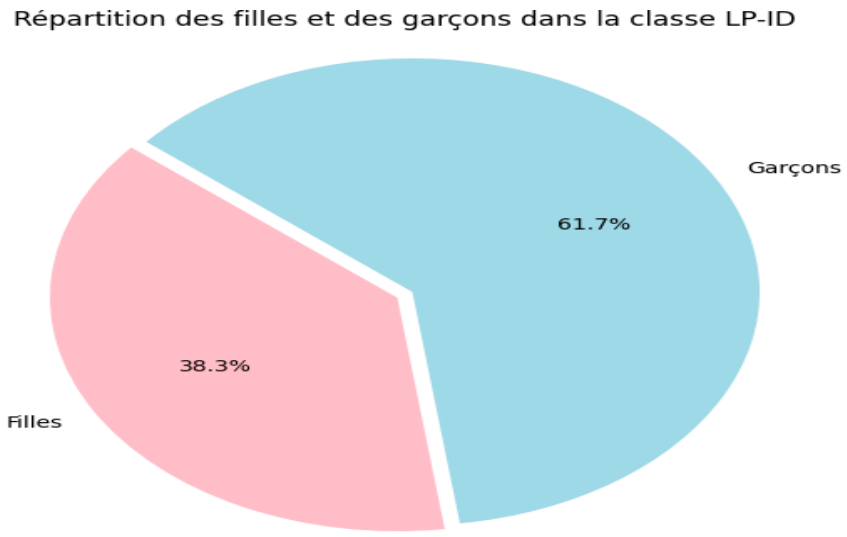
```
# Création de l'histogramme
plt.figure(figsize=(10, 6))
plt.hist(temperatures, bins=10,rwidth=0.8, color='skyblue')
# Titres et labels
plt.title('Distribution des Températures à Oujda, Maroc')
plt.xlabel('Température (°C)')
plt.ylabel('Fréquence')

# Affichage
plt.grid(True)
plt.tight_layout()
plt.show()
```



Le diagramme est affiché avec un titre approprié "Répartition des filles et des garçons dans la classe LP-ID" pour fournir une clarté sur le contenu du graphique.

```
import matplotlib.pyplot as plt
# Données
noms = ['Filles', 'Garçons']
effectifs = [18, (47-18)]
# Explosion
explode = (0, 0.05)  # Explode the second slice (Garçons)
# Création du diagramme circulaire
plt.figure(figsize=(8, 6))
plt.pie(effectifs, labels=noms, explode=explode, autopct='%1.1f%%',
startangle=140, colors=['pink', 'lightblue'])
```

```
plt.title("Répartition des filles et des garçons dans la classe LP-ID")
plt.axis('equal')  # Aspect ratio equal ensures that pie is drawn as a
circle.
plt.show()
```



II. Manipulation efficace des données avec NumPy:«Traitement des tableaux de nombres».

Ce Travaux Pratiques vise à explorer les fonctionnalités de NumPy pour charger, stocker et manipuler des données, en mettant l'accent sur la représentation des données sous forme de tableaux numériques. Nous allons découvrir comment NumPy facilite la gestion des données provenant de diverses sources et comment cette approche simplifiée permet de réaliser des analyses complexes de manière efficace et intuitive.

- **Créer des tableaux numpy à une dimension.**

```
# Plusieurs manières de créer des tableaux numpy à une dimension

import numpy as np

T = np.array([5,2,8,17,6,14])
print(T)
T1 = np.arange(20)
print(T1)
T2 = np.arange(0,10,2)
print(T2)
T3 = np.linspace(0,20,6)
print(T3)
```

```
[ 5  2  8 17  6 14]
[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19]
[0 2 4 6 8]
[ 0.  4.  8. 12. 16. 20.]
```

- **Récupérer les éléments d'un tableau numpy.**

```
T=np.array([5,2,8,17,6,14])
print(T)
print(T[0])
print(T[2])
print(T[-1])
print(T[-2])
print(T[1:5])
print(T[3:])
print(len(T))
T4 = np.array([])
print(T4)
```

```
[ 5  2  8 17  6 14]
5
8
14
6
[ 2  8 17  6]
[17  6 14]
6
[]
```

- **Ajouter une valeur ou supprimer une valeur d'un tableau numpy.**

```
T=np.array([5,2,8,17,6,14])

print(T)
T1=np.append(T,18)      # Crée un tableau T1 à partir du tableau T
print(T1)               # en ajoutant l'élément 18 à la fin du
print(T)                # tableau T. Ne modifie pas le tableau T.

T=np.append(T,18)       # Ajoute l'élément 18 à la fin du tableau T.
print(T)

T=np.insert(T,2,20)     # Insère l'élément 20 à la position d'indice 2
print(T)                # (donc en troisième position) du tableau T.

T=np.delete(T,2)        # Supprime l'élément d'indice 2 (ici l'élément
20)
print(T)                # du tableau T.
T=np.delete(T,-1)       # Supprime l'élément d'indice -1 (donc le
dernier
print(T)                # élément) du tableau T.
```

```
[ 5  2  8 17  6 14]
[ 5  2  8 17  6 14 18]
[ 5  2  8 17  6 14]
[ 5  2  8 17  6 14 18]
[ 5  2 20  8 17  6 14 18]
[ 5  2  8 17  6 14 18]
[ 5  2  8 17  6 14]
```

- **Analyser le contenu d'un tableau numpy.**

```
from random import choice

T=np.array([5,2,8,17,6,14])
print("T = ",T,'\n')

print(min(T),'et',np.min(T))
print(max(T), 'et',np.max(T),'\n')

Ttrie_endroit=sorted(T)
print("Le tableau Ttrie_endroit est une copie triée de T :")
print(" Ttrie_endroit = ",Ttrie_endroit)
print("Comme vous pouvez le constater, le tableau T n'est pas modifié
:")
print(" T = : ",T,'\n')

Ttrie_envers=sorted(T,reverse=True)
print("Le tableau Ttrie_envers est une copie triée de T :")
print(" Ttrie_envers = ",Ttrie_envers)
print("Comme vous pouvez le constater, le tableau T n'est pas modifié
:")
print(" T =",T,'\n')

T.sort()
print("Comme vous pouvez le constater, le tableau T est modifié :")
print(" T =",T,'\n')

print(choice(T))
```

```
T =  [ 5  2  8 17  6 14]

2 et 2
17 et 17

Le tableau Ttrie_endroit est une copie triée de T :
 Ttrie_endroit =  [2, 5, 6, 8, 14, 17]
Comme vous pouvez le constater, le tableau T n'est pas modifié :
 T = :  [ 5  2  8 17  6 14]

Le tableau Ttrie_envers est une copie triée de T :
 Ttrie_envers =  [17, 14, 8, 6, 5, 2]
Comme vous pouvez le constater, le tableau T n'est pas modifié :
 T = [ 5  2  8 17  6 14]

Comme vous pouvez le constater, le tableau T est modifié :
 T = [ 2  5  6  8 14 17]

5
```

- **Création de tableau numpy à deux dimensions.**

```
    # On peut créer des tableaux à plusieurs dimensions
```

```
tableau = np.array([['Anne','Tom','Léo','Eva'],
[6,7,8,9],[10,20,30,40]])
print("Tableau\n", tableau,'\n')
print("Ligne\n", tableau[0],'\n')
print("Elément d'une ligne\n", tableau[0][0],'\n')
print("Elément d'une ligne\n", tableau[1][2],'\n')
print("Elément d'une ligne, à partir de la fin\n", tableau[-1][-
1],'\n')
print("Elément d'une ligne, à partir de la fin\n",tableau[-1][0], "\n")
print("Transposition\n", tableau.T, "\n")
print("Partie de tableau\n", tableau[0:3,1:3], "\n")
print("Colonne\n", tableau[:,0], "\n")
print(tableau.shape)
```

```
Tableau
 [['Anne' 'Tom' 'Léo' 'Eva']
 ['6' '7' '8' '9']
 ['10' '20' '30' '40']]

Ligne
 ['Anne' 'Tom' 'Léo' 'Eva']

Elément d'une ligne
 Anne

Elément d'une ligne
 8

Elément d'une ligne, à partir de la fin
 40

Elément d'une ligne, à partir de la fin
 10
```

```
Transposition
 [['Anne' '6' '10']
 ['Tom' '7' '20']
 ['Léo' '8' '30']
 ['Eva' '9' '40']]

Partie de tableau
 [['Tom' 'Léo']
 ['7' '8']
 ['20' '30']]

Colonne
 ['Anne' '6' '10']

(3, 4)
```

NumPy est essentiel pour l'analyse de données grâce à ses performances optimisées et ses opérations vectorisées, simplifiant les calculs sur des tableaux multidimensionnels. Il offre une manipulation efficace des données, des fonctionnalités intégrées avec d'autres bibliothèques Python, et gère à la fois les données structurées et non structurées. En résumé, NumPy constitue une base solide pour le traitement efficace et flexible des données dans divers domaines, de la science des données à l'apprentissage automatique.

III. Exploration et Analyse de Données avec Pandas.

```
df = pd.read_csv("data.csv")
df.describe()
```

	id	tax
count	1000.000000	654.0
mean	500.500000	20.0
std	288.819436	0.0
min	1.000000	20.0
25%	250.750000	20.0
50%	500.500000	20.0
75%	750.250000	20.0
max	1000.000000	20.0

- La méthode describe() en Pandas est utilisée pour générer des statistiques descriptives pour un

DataFrame. Lorsque vous appelez df.describe(), Pandas calcule et affiche les statistiques suivantes pour chaque colonne numérique du DataFrame

```
df.head()
```

	id	date	first_name	last_name	email	gender	ip_address	country	price_paid	tax
0	1	03/28/2021	Marylin	Alders	malders0@shop-pro.jp	Male	160.119.8.119	Canada	$5.36	NaN
1	2	03/19/2021	Hinda	Harridge	hharridge1@gnu.org	Female	210.51.172.132	Canada	$8.32	NaN
2	3	03/17/2021	Erl	Kilminster	ekilminster2@etsy.com	Male	213.8.101.145	United States	$3.73	20.0
3	4	03/19/2021	Grata	Brantl	gbrantl3@umn.edu	Female	5.140.237.101	United States	$3.70	20.0
4	5	03/24/2021	Kanya	Beasant	kbeasant4@jigsy.com	Male	106.252.162.233	Morocco	$9.24	20.0

- Cela affichera les cinq premières lignes de votre DataFrame df. C'est utile pour examiner rapidement les données et comprendre leur format, les types de données des colonnes, ainsi que pour détecter d'éventuels problèmes d'importation ou de formatage.

```
df.tail()
```

	id	date	first_name	last_name	email	gender	ip_address	country	price_paid	tax
995	996	03/24/2021	Duff	Errigo	derrigorn@elpais.com	Female	193.84.247.144	France	$3.63	NaN
996	997	03/17/2021	Ame	Rastall	arastallro@toplist.cz	Female	188.219.12.109	NaN	$4.59	20.0
997	998	03/13/2021	Betty	Stickels	bstickelsrp@cloudflare.com	Male	199.80.60.87	United States	$7.54	20.0
998	999	03/17/2021	Paquito	Tesoe	ptesoerq@rakuten.co.jp	Male	0.183.63.11	United States	$5.81	NaN
999	1000	03/23/2021	Rochelle	Pringuer	rpringuerrr@house.gov	NaN	170.193.249.83	Canada	$8.01	20.0

- tail(): La méthode tail() retourne les dernières lignes du DataFrame. Par défaut, elle retourne les cinq dernières lignes, mais vous pouvez spécifier le nombre de lignes à retourner en passant un argument entier à la méthode tail().

```
df.head(2)
```

	id	date	first_name	last_name	email	gender	ip_address	country	price_paid	tax
0	1	03/28/2021	Marylin	Alders	malders0@shop-pro.jp	Male	160.119.8.119	Canada	$5.36	NaN
1	2	03/19/2021	Hinda	Harridge	hharridge1@gnu.org	Female	210.51.172.132	Canada	$8.32	NaN
2	3	03/17/2021	Erl	Kilminster	ekilminster2@etsy.com	Male	213.8.101.145	United States	$3.73	20.0
3	4	03/19/2021	Grata	Brantl	gbrantl3@umn.edu	Female	5.140.237.101	United States	$3.70	20.0
4	5	03/24/2021	Kanya	Beasant	kbeasant4@jigsy.com	Male	106.252.162.233	Morocco	$9.24	20.0
5	6	03/25/2021	Titos	Braybrooke	tbraybrooke5@umn.edu	Female	173.100.21.179	United States	$5.82	NaN
6	7	03/14/2021	Ava	Kencott	akencott6@phoca.cz	Female	118.251.45.43	Canada	$6.29	NaN
7	8	03/08/2021	Codi	Feasley	cfeasley7@ed.gov	Male	26.190.191.249	France	$3.01	20.0
8	9	03/09/2021	Waylon	Heersema	wheersema8@tumblr.com	Male	63.242.174.112	Canada	$3.70	20.0
9	10	03/23/2021	Guglielma	Gookes	ggookes9@amazon.co.uk	Male	169.100.6.0	France	$9.05	NaN

- head(n): La méthode head(n) retourne les premières n lignes du DataFrame. Elle est similaire à head(), mais au lieu de retourner les cinq premières lignes par défaut, elle retournera les n premières lignes spécifiées par l'utilisateur.

- Après avoir utilisé set_index("email"), chaque ligne du DataFrame sera identifiée par la valeur unique de la colonne "email". Cela peut être utile dans de nombreuses situations,

```
df.set_index("email")
```

email	id	date	first_name	last_name	gender	ip_address	country	price_paid	tax
malders0@shop-pro.jp	1	03/28/2021	Marylin	Alders	Male	160.119.8.119	Canada	$5.36	NaN
hharridge1@gnu.org	2	03/19/2021	Hinda	Harridge	Female	210.51.172.132	Canada	$8.32	NaN
ekilminster2@etsy.com	3	03/17/2021	Erl	Kilminster	Male	213.8.101.145	United States	$3.73	20.0
gbrantl3@umn.edu	4	03/19/2021	Grata	Brantl	Female	5.140.237.101	United States	$3.70	20.0
kbeasant4@jigsy.com	5	03/24/2021	Kanya	Beasant	Male	106.252.162.233	Morocco	$9.24	20.0
...	...	...	...	...	...	...	...	...	...
derrigorn@elpais.com	996	03/24/2021	Duff	Errigo	Female	193.84.247.144	France	$3.63	NaN
arastallro@toplist.cz	997	03/17/2021	Ame	Rastall	Female	188.219.12.109	NaN	$4.59	20.0
bstickelsrp@cloudflare.com	998	03/13/2021	Betty	Stickels	Male	199.80.60.87	United States	$7.54	20.0
ptesoerq@rakuten.co.jp	999	03/17/2021	Paquito	Tesoe	Male	0.183.63.11	United States	$5.81	NaN
rpringuerrr@house.gov	1000	03/23/2021	Rochelle	Pringuer	NaN	170.193.249.83	Canada	$8.01	20.0

1000 rows × 9 columns

notamment lors de l'indexation pour des recherches rapides ou pour la réalisation de jointures avec d'autres DataFrames basées sur l'index.

```
[8]  df[10:20]
```

	id	date	first_name	last_name	email	gender	ip_address	country	price_paid	tax
10	11	03/11/2021	Lief	Woodcraft	lwoodcrafta@sitemeter.com	Male	201.76.240.51	Canada	$5.71	NaN
11	12	03/16/2021	Bailie	Wyman	bwymanb@lulu.com	Male	222.60.138.222	United States	$3.15	20.0
12	13	03/05/2021	Janice	Morillas	jmorillasc@miibeian.gov.cn	Female	24.25.198.252	Canada	$5.30	20.0
13	14	03/24/2021	Cull	Vassie	cvassied@unblog.fr	NaN	18.203.61.94	France	$5.97	20.0
14	15	03/21/2021	Oralie	Maryon	omaryone@va.gov	Female	215.248.16.198	United States	$4.53	20.0
15	16	03/24/2021	Julina	Faint	jfaintf@amazon.co.jp	Female	96.193.4.151	United States	$4.10	20.0
16	17	03/27/2021	Eve	Allaway	eallawayg@ebay.co.uk	Female	184.107.59.156	United States	$6.44	20.0
17	18	03/20/2021	Esta	Boulden	ebouldenh@posterous.com	NaN	220.109.140.190	United States	$3.16	20.0
18	19	03/16/2021	Judye	Behan	jbehani@google.es	Female	27.196.2.18	France	$4.90	20.0
19	20	03/01/2021	Sibeal	Mounch	smounchj@rakuten.co.jp	Male	218.17.7.177	Canada	$5.82	20.0

- Dans cet exemple, iloc est utilisé pour effectuer la sélection basée sur les indices de position. La syntaxe [10:20]

- Dans cet exemple, df_email est votre DataFrame, et nous utilisons loc pour sélectionner les lignes ayant comme index les adresses e-mail 'hharridge1@gnu.org' et 'derrigorn@elpais.com'.

```
[12] df_email.loc[['hharridge1@gnu.org', 'derrigorn@elpais.com']]
```

email	id	date	first_name	last_name	gender	ip_address	country	price_paid	tax
hharridge1@gnu.org	2	03/19/2021	Hinda	Harridge	Female	210.51.172.132	Canada	$8.32	NaN
derrigorn@elpais.com	996	03/24/2021	Duff	Errigo	Female	193.84.247.144	France	$3.63	NaN

- Il retire le symbole "$" de la colonne "price_paid", le convertit en type float, puis filtre les lignes où la valeur de "price_paid" est supérieure ou égale à 5.0.

```
df_test = df.copy()
df_test.price_paid = df_test.price_paid.apply(lambda x: x.replace("$", ""))
df_test.price_paid = df_test.price_paid.astype(float)
df_test[df_test["price_paid"] >= 5.0]
```

	id	date	first_name	last_name	email	gender	ip_address	country	price_paid	tax
0	1	03/28/2021	Marylin	Alders	malders0@shop-pro.jp	Male	160.119.8.119	Canada	5.36	NaN
1	2	03/19/2021	Hinda	Harridge	hharridge1@gnu.org	Female	210.51.172.132	Canada	8.32	NaN
4	5	03/24/2021	Kanya	Beasant	kbeasant4@jigsy.com	Male	106.252.162.233	Morocco	9.24	20.0
5	6	03/25/2021	Titos	Braybrooke	tbraybrooke5@umn.edu	Female	173.100.21.179	United States	5.82	NaN
6	7	03/14/2021	Ava	Kencott	akencott6@phoca.cz	Female	118.251.45.43	Canada	6.29	NaN
...	...	...	...	...	...	...	...	...	...	...
992	993	03/04/2021	Maurene	Guilloneau	mguilloneaurk@about.me	Male	108.73.141.204	France	9.19	NaN
994	995	03/17/2021	Carolyne	Bruntjen	cbruntjenrm@reference.com	Female	7.29.168.78	France	5.68	20.0
997	998	03/13/2021	Betty	Stickels	bstickelsrp@cloudflare.com	Male	199.80.60.87	United States	7.54	20.0
998	999	03/17/2021	Paquito	Tesoe	ptesoerq@rakuten.co.jp	Male	0.183.63.11	United States	5.81	NaN
999	1000	03/23/2021	Rochelle	Pringuer	rpringuerrr@house.gov	NaN	170.193.249.83	Canada	8.01	20.0

696 rows × 10 columns

- La méthode drop() en Pandas est utilisée pour supprimer des lignes ou des colonnes d'un DataFrame. Lorsque vous appelez df.drop("ip_address", axis=1), vous demandez à Pandas de supprimer la colonne spécifiée, dans ce cas "ip_address", en spécifiant axis=1.

```
df.drop("ip_address", axis=1)
```

	id	date	first_name	last_name	email	gender	country	price_paid	tax
0	1	03/28/2021	Marylin	Alders	malders0@shop-pro.jp	Male	Canada	$5.36	NaN
1	2	03/19/2021	Hinda	Harridge	hharridge1@gnu.org	Female	Canada	$8.32	NaN
2	3	03/17/2021	Erl	Kilminster	ekilminster2@etsy.com	Male	United States	$3.73	20.0
3	4	03/19/2021	Grata	Brantl	gbrantl3@umn.edu	Female	United States	$3.70	20.0
4	5	03/24/2021	Kanya	Beasant	kbeasant4@jigsy.com	Male	Morocco	$9.24	20.0
...	...	...	...	...	...	...	...	...	...
995	996	03/24/2021	Duff	Errigo	derrigorn@elpais.com	Female	France	$3.63	NaN
996	997	03/17/2021	Ame	Rastall	arastallro@toplist.cz	Female	NaN	$4.59	20.0
997	998	03/13/2021	Betty	Stickels	bstickelsrp@cloudflare.com	Male	United States	$7.54	20.0
998	999	03/17/2021	Paquito	Tesoe	ptesoerq@rakuten.co.jp	Male	United States	$5.81	NaN
999	1000	03/23/2021	Rochelle	Pringuer	rpringuerrr@house.gov	NaN	Canada	$8.01	20.0

1000 rows × 9 columns

- Vous créez d'abord une liste columns_to_drop qui contient les noms des colonnes que vous souhaitez supprimer. Ensuite, vous utilisez la méthode drop() en spécifiant columns=columns_to_drop pour indiquer les colonnes à supprimer. L'argument inplace=True

est utilisé pour modifier le DataFrame d'origine directement. Ces deux lignes réalisent la suppression des colonnes spécifiées du DataFrame df.

```
] df.drop(['first_name', 'last_name', 'email', 'ip_address'], axis=1, inplace=True)
```

```
df.head()
```

	id	date	gender	country	price_paid	tax
0	1	03/28/2021	Male	Canada	$5.36	NaN
1	2	03/19/2021	Female	Canada	$8.32	NaN
2	3	03/17/2021	Male	United States	$3.73	20.0
3	4	03/19/2021	Female	United States	$3.70	20.0
4	5	03/24/2021	Male	Morocco	$9.24	20.0

- L'instruction df.dropna(subset=["tax"]) en Pandas permet de supprimer les lignes qui contiennent des valeurs manquantes (NaN) dans la colonne spécifiée, ici "tax".

```
df.dropna(subset=["tax"])
```

	id	date	first_name	last_name	email	gender	ip_address	country	price_paid	tax
2	3	03/17/2021	Erl	Kilminster	ekilminster2@etsy.com	Male	213.8.101.145	United States	$3.73	20.0
3	4	03/19/2021	Grata	Brantl	gbrantl3@umn.edu	Female	5.140.237.101	United States	$3.70	20.0
4	5	03/24/2021	Kanya	Beasant	kbeasant4@jigsy.com	Male	106.252.162.233	Morocco	$9.24	20.0
7	8	03/08/2021	Codi	Feasley	cfeasley7@ed.gov	Male	26.190.191.249	France	$3.01	20.0
8	9	03/09/2021	Waylon	Heersema	wheersema8@tumblr.com	Male	63.242.174.112	Canada	$3.70	20.0
...	...	...	...	...	...	...	...	...	...	...
991	992	03/07/2021	Elihu	Crush	ecrushrj@berkeley.edu	Male	174.38.173.53	Morocco	$5.05	20.0
994	995	03/17/2021	Carolyne	Bruntjen	cbruntjenrm@reference.com	Female	7.29.168.78	France	$5.68	20.0
996	997	03/17/2021	Ame	Rastall	arastallro@toplist.cz	Female	188.219.12.109	NaN	$4.59	20.0
997	998	03/13/2021	Betty	Stickels	bstickelsrp@cloudflare.com	Male	199.80.60.87	United States	$7.54	20.0
999	1000	03/23/2021	Rochelle	Pringuer	rpringuerrr@house.gov	NaN	170.193.249.83	Canada	$8.01	20.0

654 rows × 10 columns

- Cette expression est couramment utilisée pour calculer le prix total après l'application d'une taxe. Elle suppose que la colonne "tax" contient les pourcentages de taxe, par exemple 10 pour une taxe de 10%, et que la colonne "price_paid" contient les prix initiaux avant l'application de la taxe.

```
df["price_total"] = df["price_paid"] * (1 - df["tax"] / 100)
```

```
df
```

	id	date	first_name	last_name	email	gender	ip_address	country	price_paid	tax	price_total
2	3	03/17/2021	Erl	Kilminster	ekilminster2@etsy.com	Male	213.8.101.145	United States	3.73	20.0	2.984
3	4	03/19/2021	Grata	Brantl	gbrantl3@umn.edu	Female	5.140.237.101	United States	3.70	20.0	2.960
4	5	03/24/2021	Kanya	Beasant	kbeasant4@jigsy.com	Male	106.252.162.233	Morocco	9.24	20.0	7.392
7	8	03/08/2021	Codi	Feasley	cfeasley7@ed.gov	Male	26.190.191.249	France	3.01	20.0	2.408
8	9	03/09/2021	Waylon	Heersema	wheersema8@tumblr.com	Male	63.242.174.112	Canada	3.70	20.0	2.960
...	...	...	...	...	...	...	...	...	...	...	...
991	992	03/07/2021	Elihu	Crush	ecrushrj@berkeley.edu	Male	174.38.173.53	Morocco	5.05	20.0	4.040
994	995	03/17/2021	Carolyne	Bruntjen	cbruntjenrm@reference.com	Female	7.29.168.78	France	5.68	20.0	4.544
996	997	03/17/2021	Ame	Rastall	arastallro@toplist.cz	Female	188.219.12.109	NaN	4.59	20.0	3.672
997	998	03/13/2021	Betty	Stickels	bstickelsrp@cloudflare.com	Male	199.80.60.87	United States	7.54	20.0	6.032
999	1000	03/23/2021	Rochelle	Pringuer	rpringuerrr@house.gov	NaN	170.193.249.83	Canada	8.01	20.0	6.408

654 rows × 11 columns

- Cette ligne de code supprime les colonnes "first_name", "last_name", "email", "ip_address" et "id" du DataFrame df. La modification est effectuée directement sur le DataFrame d'origine (df) en raison de inplace=True.

```
df.drop(["first_name", "last_name", "email", "ip_address", "id"], axis=1, inplace=True)
df
```

	date	gender	country	price_paid	tax	price_total
2	03/17/2021	Male	United States	3.73	20.0	2.984
3	03/19/2021	Female	United States	3.70	20.0	2.960
4	03/24/2021	Male	Morocco	9.24	20.0	7.392
7	03/08/2021	Male	France	3.01	20.0	2.408
8	03/09/2021	Male	Canada	3.70	20.0	2.960
...	...	...	...	...	...	...
991	03/07/2021	Male	Morocco	5.05	20.0	4.040
994	03/17/2021	Female	France	5.68	20.0	4.544
996	03/17/2021	Female	NaN	4.59	20.0	3.672
997	03/13/2021	Male	United States	7.54	20.0	6.032
999	03/23/2021	NaN	Canada	8.01	20.0	6.408

654 rows × 6 columns