exercices_pandas_v3

October 1, 2025

1 Exercices

[7]:
```
import pandas as pd
import sqlite3
```

2 ==========

3 Partie A — Débutants

4 ==========

5 1. Charger le fichier `customers.csv` et afficher les 5 premières lignes

[8]:
```
customers = pd.read_csv('data/customers.csv')
print("5 premières lignes :")
print(customers.head())
```

```
5 premières lignes :
   id     name  age       city
0   1    Alice   25      Paris
1   2      Bob   35       Lyon
2   3  Charlie   40  Marseille
3   4    Diana   28      Paris
4   5    Ethan   32      Lille
```

6 2. Afficher uniquement la colonne `name`

[9]:
```
print("\nColonne 'name' :")
print(customers['name'])
```

```
Colonne 'name' :
0      Alice
1        Bob
2    Charlie
```

```
3      Diana
4      Ethan
Name: name, dtype: object
```

7 3. Lister les clients de Paris

[10]:
```
paris_clients = customers[customers['city'] == 'Paris']
print("\nClients de Paris :")
print(paris_clients)
```

```
Clients de Paris :
   id   name  age   city
0   1  Alice   25  Paris
3   4  Diana   28  Paris
```

8 4. Trier les clients par âge décroissant

[11]:
```
sorted_customers = customers.sort_values(by='age', ascending=False)
print("\nClients triés par âge décroissant :")
print(sorted_customers)
```

```
Clients triés par âge décroissant :
   id     name  age       city
2   3  Charlie   40  Marseille
1   2      Bob   35       Lyon
4   5    Ethan   32      Lille
3   4    Diana   28      Paris
0   1    Alice   25      Paris
```

9 5. Compter combien de clients viennent de chaque ville

[12]:
```
city_counts = customers['city'].value_counts()
print("\nNombre de clients par ville :")
print(city_counts)
```

```
Nombre de clients par ville :
city
Paris        2
Lyon         1
Marseille    1
Lille        1
Name: count, dtype: int64
```

10 6. Sauvegarder les clients de Lyon dans `data/lyon_clients.csv`

[13]:
```
lyon_clients = customers[customers['city'] == 'Lyon']
lyon_clients.to_csv('data/lyon_clients.csv', index=False)
print("\nClients de Lyon sauvegardés dans 'data/lyon_clients.csv'.")
```

```
Clients de Lyon sauvegardés dans 'data/lyon_clients.csv'.
```

11 (Optionnel) Lister les clients dont le nom commence par “A”

[14]:
```
clients_a = customers[customers['name'].str.startswith('A')]
print("\nClients dont le nom commence par 'A' :")
print(clients_a)
```

```
Clients dont le nom commence par 'A' :
   id   name  age   city
0   1  Alice   25  Paris
```

12 ==========

13 Partie B — Challenge

14 ==========

15 7. Charger la table `orders` depuis la base `marketing.db`

[15]:
```
conn = sqlite3.connect('data/marketing.db')
orders = pd.read_sql_query("SELECT * FROM orders", conn)
print("\n5 premières commandes :")
print(orders.head())
```

```
5 premières commandes :
   id  customer_id  amount  order_date
0   1            1    39.9  2025-06-01
1   2            2   120.0  2025-06-02
2   3            1    59.0  2025-06-03
3   4            3    80.0  2025-06-04
4   5            4    25.0  2025-06-05
```

16 8. Lister les 5 premières commandes avec le nom du client

17 On suppose que 'orders' a 'customer_id' et 'customers' a 'id'

18 Charger le fichier customers.csv

[16]:
```
customers = pd.read_csv('data/customers.csv')
print("Colonnes customers :", customers.columns)
```

```
Colonnes customers : Index(['id', 'name', 'age', 'city'], dtype='object')
```

19 Merge orders avec customers pour avoir le nom du client

[17]:
```
orders_with_names = orders.merge(customers, left_on='customer_id',␣
↪right_on='id')
```

20 Renommer les colonnes pour plus de clarté

[18]:
```
orders_with_names.rename(columns={'id_x':'order_id', 'id_y':'customer_id',␣
↪'order_date':'date'}, inplace=True)
```

21 Afficher les 5 premières commandes avec le nom du client

[19]:
```
print("\n5 premières commandes avec le nom du client :")
print(orders_with_names[['order_id', 'name', 'amount', 'date']].head())
```

```
5 premières commandes avec le nom du client :
   order_id     name  amount        date
0         1    Alice    39.9  2025-06-01
1         2      Bob   120.0  2025-06-02
2         3    Alice    59.0  2025-06-03
3         4  Charlie    80.0  2025-06-04
4         5    Diana    25.0  2025-06-05
```

22 9. Calculer le total dépensé par chaque client

[20]:
```
total_by_client = orders_with_names.groupby('name')['amount'].sum().
↪reset_index()
print("\nTotal dépensé par chaque client :")
print(total_by_client)
```

```
Total dépensé par chaque client :
      name  amount
```

```
0    Alice    98.9
1      Bob   120.0
2  Charlie    80.0
3    Diana    25.0
```

23 10. Calculer le panier moyen par ville

[21]:
```
average_by_city = orders_with_names.groupby('city')['amount'].mean().
 ↪reset_index()
print("\nPanier moyen par ville :")
print(average_by_city)
```

```
Panier moyen par ville :
        city  amount
0       Lyon   120.0
1  Marseille    80.0
2      Paris    41.3
```

24 11. Exporter le rapport en CSV

[22]:
```
average_by_city.to_csv('data/panier_moyen_par_ville.csv', index=False)
print("\nRapport exporté dans 'data/panier_moyen_par_ville.csv'.")
```

```
Rapport exporté dans 'data/panier_moyen_par_ville.csv'.
```

25 (Optionnel) Trouver le top 3 clients par montant total

[23]:
```
top3_clients = total_by_client.sort_values(by='amount', ascending=False).head(3)
print("\nTop 3 clients par montant total :")
print(top3_clients)
```

```
Top 3 clients par montant total :
      name  amount
1      Bob   120.0
0    Alice    98.9
2  Charlie    80.0
```

26 Créer la connexion

[24]:
```
import sqlite3

conn = sqlite3.connect('marketing.db')
```

27 ... ton code qui utilise la base ...

28 Fermer la connexion à la base

[25]:
```
conn.close()
```