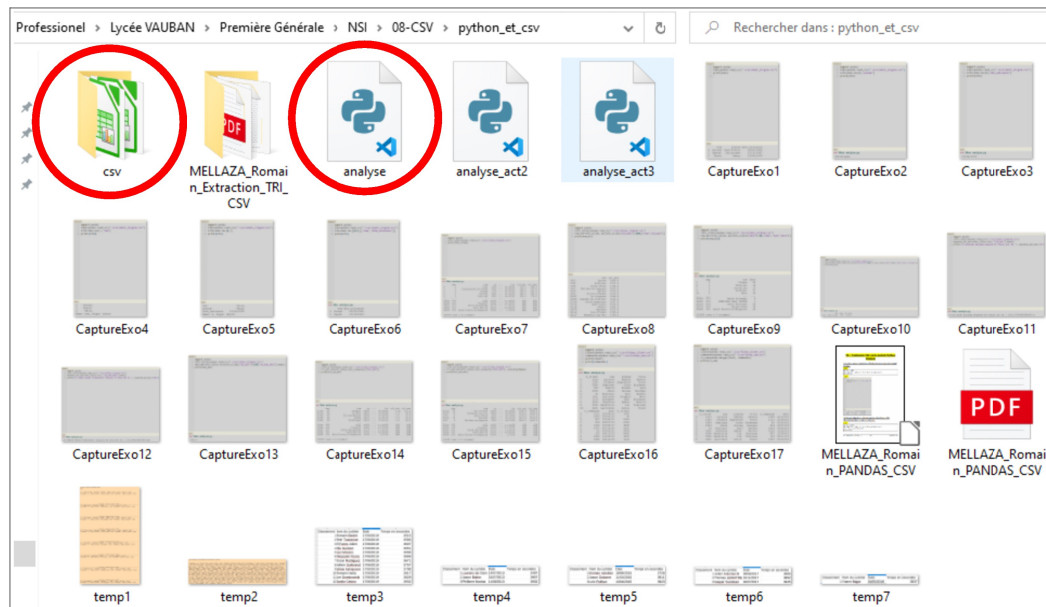








TD – Traitement CSV via le module Python

PANDAS

1) Téléchargement et indexation judicieuse du fichier « ident_virgule.csv »



Nom	Modifié le	Type	Taille
 classement_user	19/03/2022 22:03	Classeur OpenOffi...	1 Ko
 fiches_client	04/03/2022 14:41	Classeur OpenOffi...	1 Ko
 fiches_com	04/03/2022 14:41	Classeur OpenOffi...	1 Ko
 ident_virgule	04/03/2022 13:50	Classeur OpenOffi...	1 Ko
 liste_origine	14/03/2022 21:07	Classeur OpenOffi...	2 Ko
 villes_virgule	04/03/2022 14:16	Classeur OpenOffi...	2 326 Ko

Comme on peut le voir sur ces différentes capture j'ai téléchargé le fichier « **ident_virgule.csv** » et je l'ai placé dans un dossier /csv qui est enfant du dossier « **python_et_csv** » où se trouve mon code python d'analyse.

2) Définition de l'emplacement du fichier csv à traiter :

Code :

```
import pandas
iden=pandas.read_csv(".\csv\ident_virgule.csv")
```

Le code ci-dessus est très simple :

- Avec la première ligne, nous importons la bibliothèque pandas afin de pouvoir l'utiliser
- À la deuxième ligne, nous créons une variable "iden" qui va contenir les données présentes dans le fichier "**ident_virgule.csv**"

3) Lecture simple d'un fichier CSV directement dans le terminal Python :

Code :

```
import pandas
iden=pandas.read_csv("./csv/ident_virgule.csv")
print(iden)
```

Sortie :



The screenshot shows a Python IDE with a file named 'analyse.py'. The code in the editor is:

```
1 import pandas
2 iden=pandas.read_csv("./csv/ident_virgule.csv")
3 print(iden)
4
```

Below the code editor, a 'Shell' window displays the output of the program as a table:

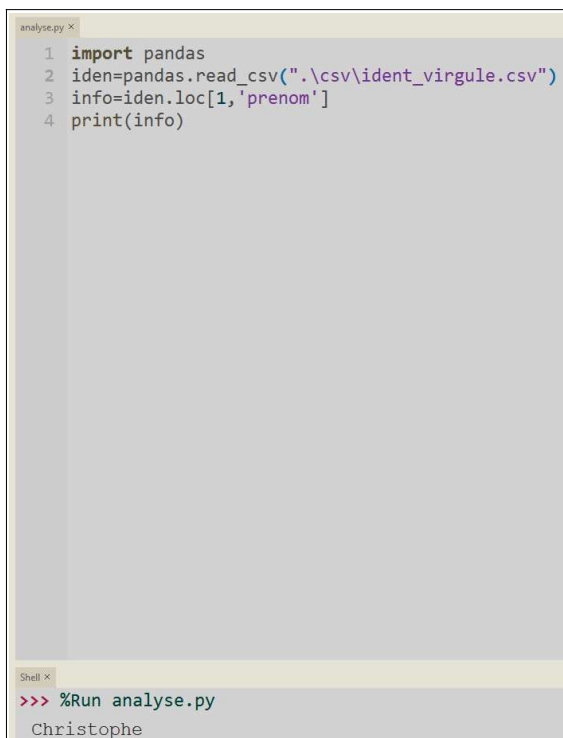
	nom	prenom	date_naissance
0	Durand	Jean-Pierre	23/05/1985
1	Dupont	Christophe	15/12/1967
2	Terta	Henry	12/06/1978

4) Récupération d'une donnée précise d'un fichier CSV directement dans le terminal Python (Prenom) :

Code :

```
import pandas
iden=pandas.read_csv(".\csv\ident_virgule.csv")
info=iden.loc[1,"prenom"]
print(info)
```

Sortie :

A screenshot of a code editor and a terminal window. The code editor shows a file named 'analyse.py' with the following code:

```
1 import pandas
2 iden=pandas.read_csv(".\csv\ident_virgule.csv")
3 info=iden.loc[1,'prenom']
4 print(info)
```

The terminal window below shows the command to run the script:

```
>>> %Run analyse.py
```

and the output:

```
Christophe
```

Comme prévu grâce à l'analyse précédente, on sait qu'à la première ligne dans la colonne « prenom » (`iden.loc[1,«prenom»]`) je dois normalement trouver la donnée « Christophe » en sortie et c'est bien le cas !

5) Récupération d'une donnée précise d'un fichier CSV directement dans le terminal Python (Date) :

Code :

```
import pandas
iden=pandas.read_csv(".\csv\ident_virgule.csv")
info=iden.loc[2,"date_naissance"]
print(info)
```

Sortie :

```
analyse.py x
1 import pandas
2 iden=pandas.read_csv(".\csv\ident_virgule.csv")
3 info=iden.loc[2,'date_naissance']
4 print(info)

Shell x
>>> %Run analyse.py
12/06/1978
```

Comme prévu grâce à l'analyse précédente, on sait qu'à la première ligne dans la colonne «**date_naissance**» (**iden.loc[2,«date_naissance»]**) je dois normalement trouver la donnée « 12/06/1978 » et c'est bien le cas !

6) Récupération de toutes les données d'une colonne précisée, d'un fichier CSV, directement dans le terminal Python :

Code :

```
import pandas
iden=pandas.read_csv(".\csv\ident_virgule.csv")
info=iden.loc[:, "nom"]
print(info)
```

Sortie :

```
analyse.py x
1 import pandas
2 iden=pandas.read_csv(".\csv\ident_virgule.csv")
3 info=iden.loc[:, 'nom']
4 print(info)

Shell x
0    Durand
1    Dupont
2    Terta
Name: nom, dtype: object
```

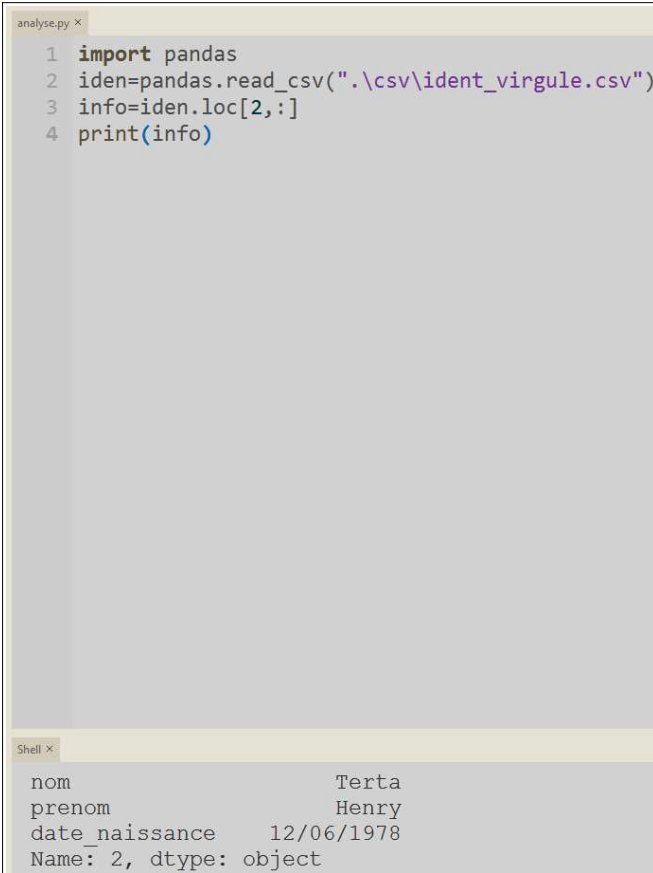
J'utilise l'élément « : » pour afficher dans le terminal toutes les données contenues dans la colonne « nom ». La variable "info" contient bien toutes les données de la colonne d'index "nom", autrement dit, tous les noms

7) Récupération de toutes les données d'une ligne précisée, d'un fichier CSV, directement dans le terminal Python :

Code :

```
import pandas
iden=pandas.read_csv(".\csv\ident_virgule.csv")
info=iden.loc[2,:]  
print(info)
```

Sortie :



The screenshot shows a Python IDE with two panels. The top panel, titled 'analyse.py', contains the following code:

```
1 import pandas
2 iden=pandas.read_csv(".\csv\ident_virgule.csv")
3 info=iden.loc[2,:]
4 print(info)
```

The bottom panel, titled 'Shell', shows the output of the code:

```
nom          Terta
prenom       Henry
date_naissance 12/06/1978
Name: 2, dtype: object
```

J'utilise l'élément « : » pour afficher dans le terminal toutes les données contenues dans la troisième ligne.

En l'occurrence cette troisième ligne contient toutes les informations sur un certain « Henry Terta ».

À noter que c'est bien la troisième ligne, car il ne faut pas oublier la ligne 0.

8) Combinaison pour récupérer toutes les données d'une ligne ou de plusieurs lignes ainsi que d'une colonne ou de plusieurs colonnes, d'un fichier CSV, directement dans le terminal Python :

Code :

```
import pandas
iden=pandas.read_csv(".\csv\ident_virgule.csv")
info=iden.loc[[0,1],["nom","date_naissance"]]
print(info)
```

Sortie :



The screenshot shows a code editor window titled 'analyse.py' with the following Python code:

```
1 import pandas
2 iden=pandas.read_csv(".\csv\ident_virgule.csv")
3 info=iden.loc[[0,1],['nom','date_naissance']]
4 print(info)
```

Below the code editor is a terminal window titled 'Shell' showing the command to run the script and its output:

```
>>> %Run analyse.py
      nom date_naissance
0  Durand   23/05/1985
1  Dupont   15/12/1967
```

En l'occurrence ici je demande au programme de n'afficher que la première et la deuxième ligne avec seulement leur donnée respective des colonnes « nom » et « date_naissance ».

9) Lecture complexe d'un fichier CSV plus conséquent directement dans le terminal Python :

Code :

```
import pandas
info_villes=pandas.read_csv(".\csv\villes_virgule.csv")
print(info_villes)
```

Sortie :

```
analyse.py x
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 print(info_villes)

Shell x
>>> %Run analyse.py
   dep  nom  cp  ...  lat  alt_min  alt_max
0     1  Ozan  1190  ...  46.38330  170.0  205.0
1     1  Cormoranche-sur-Saône  1290  ...  46.23330  168.0  211.0
2     1  Plagne  1130  ...  46.18330  560.0  922.0
3     1  Tossiat  1250  ...  46.13330  244.0  501.0
4     1  Pouillat  1250  ...  46.33330  333.0  770.0
...  ...  ...  ...  ...  ...  ...
36695  976  Sada  97640  ...  -12.84860  NaN  NaN
36696  976  Tsingoni  97680  ...  -12.78970  NaN  NaN
36697  971  Saint-Barthélemy  97133  ...  17.91670  NaN  NaN
36698  971  Saint-Martin  97150  ...  -63.08290  NaN  NaN
36699  975  Saint-Pierre-et-Miquelon  97500  ...  1.71819  NaN  NaN
[36700 rows x 12 columns]
```

Comme on peut le voir ici la variable "info_villes" contient bien les données contenues dans le fichier ville_virgule.csv. Je constate aussi qu'il manque des données dans le tableau qui s'affiche dans la console (les données manquantes sont symbolisées par des ...), en effet, le tableau contient trop de données pour qu'il soit entièrement affiché dans la console.

10) Sélection de certaines données d'un fichier CSV en fonction de leurs valeurs numériques, directement dans le terminal Python (altitude minimale):

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
nom_alt=info_villes.loc[info_villes["alt_min"]>1500,["nom","alt_min"]]
print(nom_alt)
```

Sortie :

```
analyse.py X
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 nom_alt=info_villes.loc[info_villes["alt_min"]>1500,["nom","alt_min"]]
4 print(nom_alt)
```

```
Shell X
      nom  alt_min
1618  Larche  1606.0
1790  Ristolas  1571.0
1798  Saint-Véran  1756.0
1847  Molines-en-Queyras  1625.0
1904  Abriès  1513.0
1923  Villar-d'Arène  1519.0
26927  La Llagonne  1546.0
26943  Caudiès-de-Conflent  1616.0
27039  Porté-Puymorens  1557.0
27125  Mont-Louis  1516.0
27134  Angles  1531.0
29970  Bessans  1673.0
30114  Val-d'Isère  1785.0
30127  Bonneval-sur-Arc  1759.0
```

J'obtiens bien un tableau contenant toutes les villes du fichier qui ont une altitude minimum supérieure à 1500 m

11) Sélection de certaines données d'un fichier CSV en fonction de leurs valeurs numériques, directement dans le terminal Python (densité de population) :

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
nom_alt=info_villes.loc[info_villes["dens"]<50,["dep","nom","dens"]]
print(nom_alt)
```


Sortie :

```
analyse.py x
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 nom_alt=info_villes.loc[info_villes["dens"]<50,["dep","nom","dens"]]
4 print(nom_alt)
```

```
Shell x
>>> %Run analyse.py
```

	dep	nom	dens
2	1	Plagne	20
4	1	Pouillat	14
7	1	Corcelles	17
9	1	Relevant	37
14	1	Béon	36
...
36654	973	Awala-Yalimapo	6
36655	973	POMPIDOU PAPA ICHTON	1
36672	974	Saint-Philippe	33
36674	974	Sainte-Rose	38
36699	975	Saint-Pierre-et-Miquelon	25

[21295 rows x 3 columns]

J'obtiens bien un tableau contenant toutes les villes du fichier qui ont une densité de population inférieure à 50.

12) Combinaison de plusieurs facteurs de sélections pour isoler certaines données d'un fichier CSV, directement dans le terminal Python :

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
nom_alt=info_villes.loc[(info_villes["alt_min"]>1500) &
(info_villes["dens"]>50),["nom","dens","alt_min"]]
print(nom_alt)
```

Sortie :

```
analyse.py X
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 nom_alt=info_villes.loc[(info_villes["alt_min"]>1500) & (info_villes["dens"]>50),["nom","dens","alt_min"]]
4 print(nom_alt)

Shell X
>>> %Run analyse.py
      nom  dens  alt_min
27125  Mont-Louis    633   1516.0
```

J'obtiens en sortie une seule ville française qui a une densité de population supérieure à 50 ainsi qu'une altitude minimale supérieure à 1500m, il s'agit de « Mont-Louis ».

13) Effectuer des calculs avec des données d'un fichier CSV, et afficher le résultat directement dans le terminal Python (moyenne altitude minimale) :

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
moyenne_alt_min=info_villes.loc[:, "alt_min"].mean()
print("L'altitude minimale moyenne en France est de :",
moyenne_alt_min, "m")
```

Sortie :

```
analyse.py X
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 moyenne_alt_min=info_villes.loc[:, "alt_min"].mean()
4 print("L'altitude minimum moyenne En France est de :", moyenne_alt_min,"m")

Shell X
>>> %Run analyse.py
L'altitude minimum moyenne En France est de : 193.15756945963685 m
```

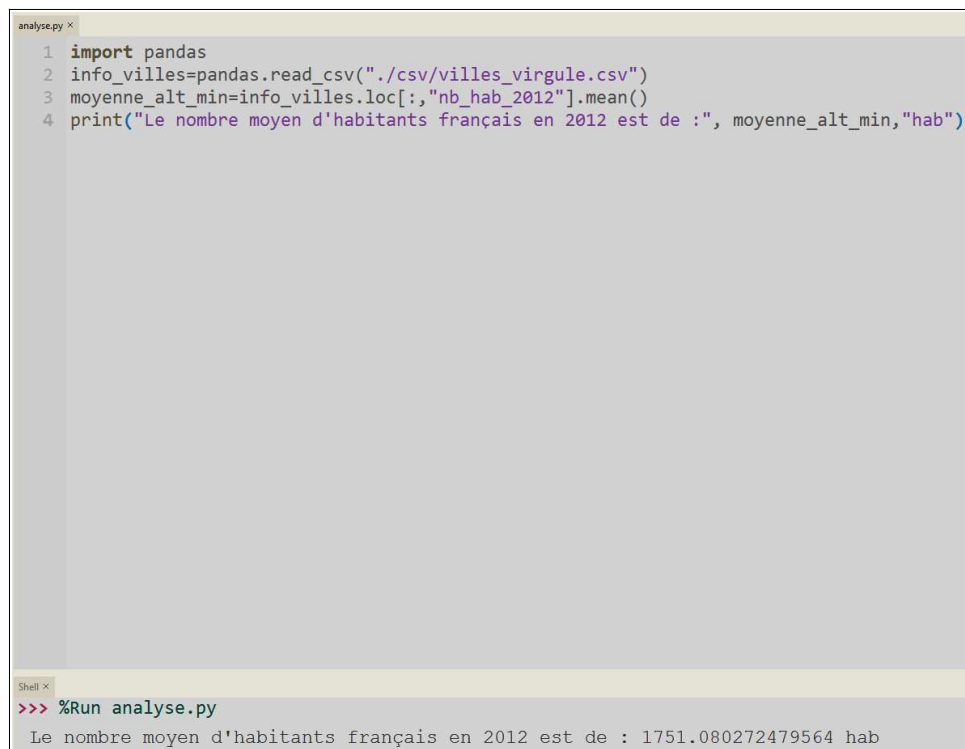
J'obtiens bien en sortie la valeur de 193m comme l'altitude minimale moyenne en France.

14) Effectuer des calculs avec des données d'un fichier CSV, et afficher le résultat directement dans le terminal Python (moyenne nombre d'habitants par ville) :

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
moyenne_alt_min=info_villes.loc[:, "nb_hab_2012"].mean()
print("Le nombre moyen d'habitants français en 2012 est de :",
moyenne_alt_min, "hab")
```

Sortie :

A screenshot of a code editor window titled 'analyse.py' showing a Python script. The script imports pandas, reads a CSV file, calculates the mean of 'nb_hab_2012', and prints the result. Below the code editor is a terminal window titled 'Shell' showing the command '%Run analyse.py' and the output 'Le nombre moyen d'habitants français en 2012 est de : 1751.080272479564 hab'.

```
analyse.py x
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 moyenne_alt_min=info_villes.loc[:, "nb_hab_2012"].mean()
4 print("Le nombre moyen d'habitants français en 2012 est de :", moyenne_alt_min, "hab")

Shell x
>>> %Run analyse.py
Le nombre moyen d'habitants français en 2012 est de : 1751.080272479564 hab
```

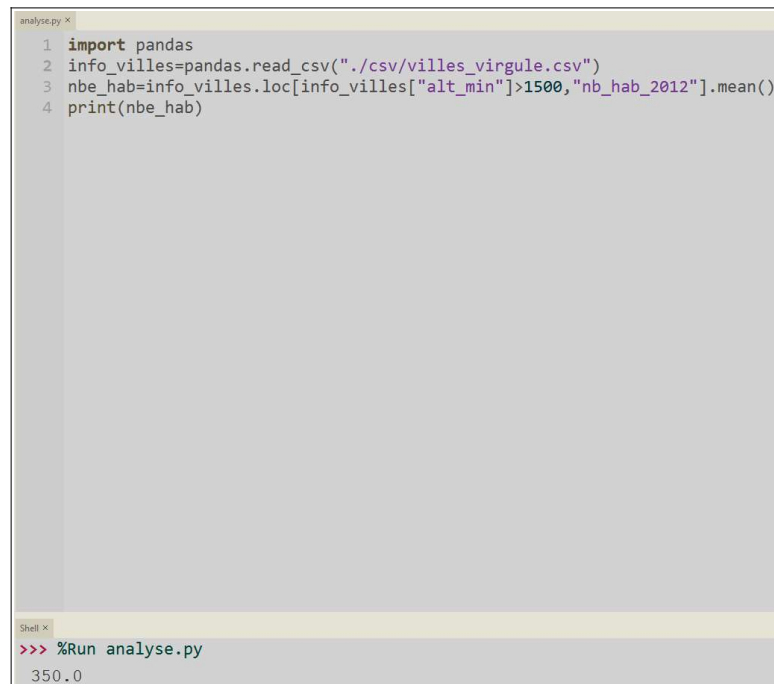
Le nombre moyen d'habitants par ville en France et en 2012 est d'environ 1751 habitants.

15) Imposer une condition sur les lignes qui seront utilisées pour le calcul d'un fichier CSV, et afficher le résultat directement dans le terminal Python :

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
nbe_hab=info_villes.loc[info_villes["alt_min"]>1500,"nb_hab_2012"].mean()
print(nbe_hab)
```

Sortie :



```
analyse.py x
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 nbe_hab=info_villes.loc[info_villes["alt_min"]>1500,"nb_hab_2012"].mean()
4 print(nbe_hab)

Shell x
>>> %Run analyse.py
350.0
```

Grâce à ce que j'obtiens en sortie, je sais que les villes françaises ayant une altitude minimale supérieure à 1500 m avaient en moyenne 350 habitants en 2012.

16) Trier le tableau en fonction des valeurs d'un descripteur d'un fichier CSV, et afficher le résultat directement dans le terminal Python (altitude minimale croissante) :

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
tri_alt_min=info_villes.sort_values(by=["alt_min"])
print(tri_alt_min)
```

Sortie :

```
analyse.py x
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 tri_alt_min=info_villes.sort_values(by=["alt_min"])
4 print(tri_alt_min)

Shell x
>>> %Run analyse.py
      dep      nom      cp  ...      lat  alt_min  alt_max
11100  29      Quimper  29000  ...  48.00000    -5.0    151.0
21570  56      Le H zo  56450  ...  47.58330    -1.0    38.0
21409  56      Ile-aux-Moines  56780  ...  47.59670    -1.0    31.0
21466  56      Baden    56870  ...  47.61670    -1.0    43.0
10963  29      Treffiat  29730  ...  47.81670    -1.0    26.0
...      ...      ...      ...      ...      ...
36695  976      Sada     97640  ... -12.84860     NaN     NaN
36696  976      Tsingoni  97680  ... -12.78970     NaN     NaN
36697  971      Saint-Barth lemy  97133  ...  17.91670     NaN     NaN
36698  971      Saint-Martin  97150  ... -63.08290     NaN     NaN
36699  975      Saint-Pierre-et-Miquelon  97500  ...  1.71819     NaN     NaN

[36700 rows x 12 columns]
```

Gr ce au tableau que j'obtiens en sortie, je sais que la ville ayant la plus petite altitude de France est Quimper. D'ailleurs en explorant un peu les donn es suivantes on peut en venir   la conclusion que la Bretagne a une faible altitude.

17) Trier le tableau en fonction des valeurs d'un descripteur d'un fichier CSV, et afficher le r sultat directement dans le terminal Python (altitude minimale d croissante) :

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
tri_alt_min=info_villes.sort_values(by=["alt_min"], ascending=False)
print(tri_alt_min)
```

Sortie :

```
analyse.py x
1 import pandas
2 info_villes=pandas.read_csv("./csv/villes_virgule.csv")
3 tri_alt_min=info_villes.sort_values(by=["alt_min"], ascending=False)
4 print(tri_alt_min)

Shell x
>>> %Run analyse.py
      dep      nom      cp  ...      lat  alt_min  alt_max
30114   73      Val-d'Isère  73150  ...  45.45000   1785.0   3599.0
30127   73  Bonneval-sur-Arc  73480  ...  45.36670   1759.0   3642.0
1798    5      Saint-Véran   5350  ...  44.70000   1756.0   3175.0
29970   73      Bessans   73480  ...  45.31670   1673.0   3754.0
1847    5  Molines-en-Queyras   5350  ...  44.73330   1625.0   3160.0
...     ...     ...     ...     ...     ...     ...
36695  976      Sada   97640  ... -12.84860      NaN      NaN
36696  976      Tsingoni  97680  ... -12.78970      NaN      NaN
36697  971  Saint-Barthélemy  97133  ...  17.91670      NaN      NaN
36698  971      Saint-Martin  97150  ... -63.08290      NaN      NaN
36699  975  Saint-Pierre-et-Miquelon  97500  ...   1.71819      NaN      NaN

[36700 rows x 12 columns]
```

Grâce au tableau que j'obtiens en sortie, je sais que la ville avec la plus importante altitude minimale de France est Val-d'Isère. D'ailleurs en explorant un peu les données suivantes [Savoie (73) et Hautes-Alpes (5)] on peut en venir à la conclusion que évidemment ce sont les départements montagneux qui comportent le plus de villes à « hautes-altitudes ».

18) Trier le tableau en fonction des valeurs d'un descripteur d'un fichier CSV, et afficher le résultat directement dans le terminal Python (densité de population décroissante) :

Code :

```
import pandas
info_villes=pandas.read_csv("./csv/villes_virgule.csv")
tri_dens_max=info_villes.sort_values(by="dens", ascending=False)
display=tri_dens_max.loc[[35912],['nom','dens']]
print(display)
```

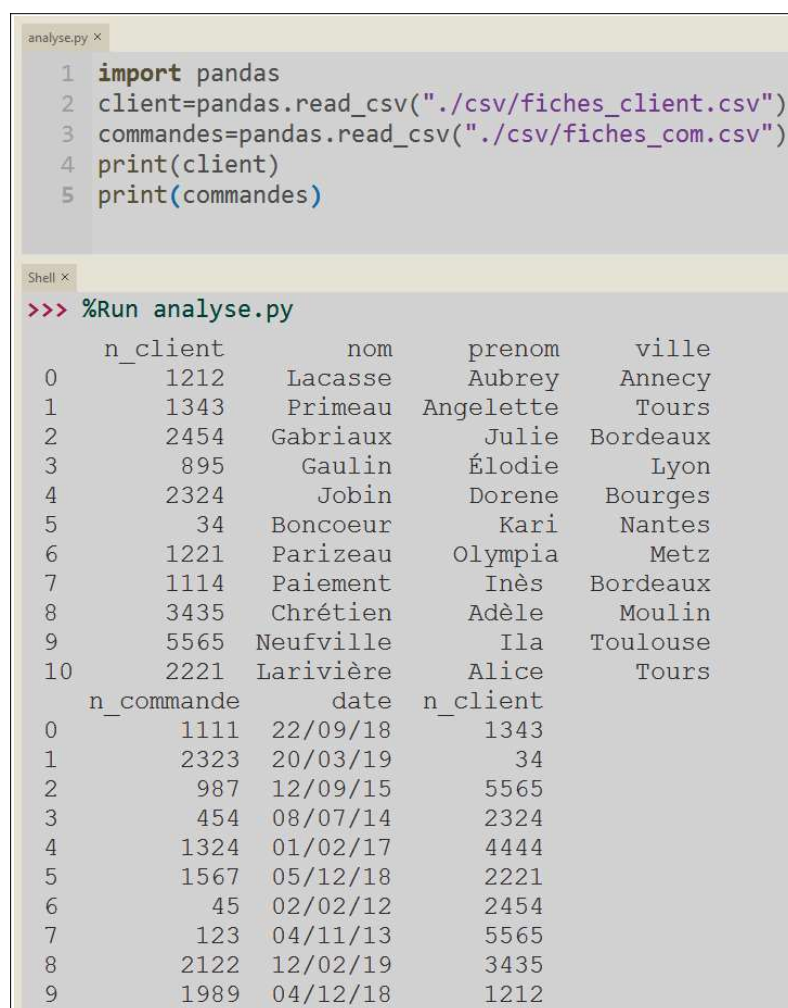
Grâce à ce code, j'obtiens en sortie, l'information que la ville la plus dense de France au moment de l'étude est Levallois-Perret.

19) Lecture simple de deux fichiers CSV directement dans le terminal Python :

Code :

```
import pandas
client=pandas.read_csv("./csv/fiches_client.csv")
commandes=pandas.read_csv("./csv/fiches_com.csv")
print(client)
print(commandes)
```

Sortie :



```
analyse.py x
1 import pandas
2 client=pandas.read_csv("./csv/fiches_client.csv")
3 commandes=pandas.read_csv("./csv/fiches_com.csv")
4 print(client)
5 print(commandes)
```

```
Shell x
>>> %Run analyse.py

   n_client  nom      prenom  ville
0      1212  Lacasse  Aubrey   Annecy
1      1343  Primeau  Angelette  Tours
2      2454  Gabriaux  Julie   Bordeaux
3       895   Gaulin  Élodie   Lyon
4      2324   Jobin   Dorene  Bourges
5        34  Boncoeur    Kari   Nantes
6      1221  Parizeau  Olympia  Metz
7      1114  Paiement    Inès  Bordeaux
8      3435  Chrétien  Adèle   Moulin
9      5565  Neufville    Ila  Toulouse
10     2221  Larivière  Alice   Tours

   n_commande  date      n_client
0          1111  22/09/18        1343
1          2323  20/03/19          34
2           987  12/09/15       5565
3           454  08/07/14       2324
4          1324  01/02/17      4444
5          1567  05/12/18       2221
6           45   02/02/12       2454
7           123  04/11/13       5565
8          2122  12/02/19       3435
9          1989  04/12/18       1212
```

J'obtiens bien en sortie les deux fichiers CSV mais dans deux tableaux différents alors qu'il y a lien (n_client) entre les clients et leur commandes respectives, je vais donc les fusionner.

20) Fusionner deux fichiers CSV, pour ne former qu'un seul tableau et afficher le résultat directement dans le terminal Python (première méthode):

Code :

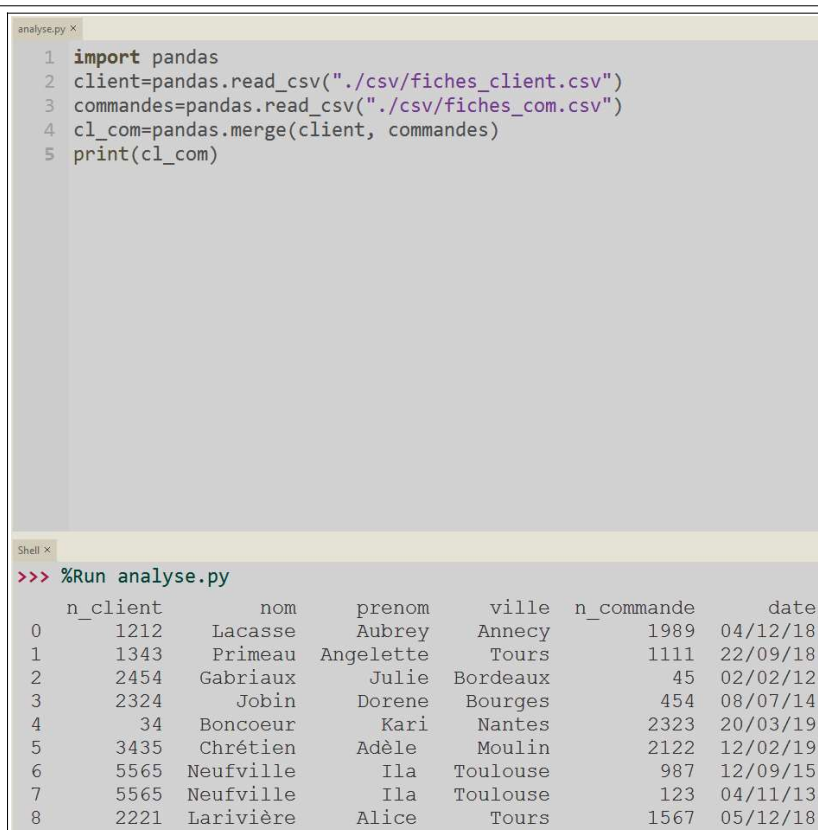
```
import pandas
client=pandas.read_csv("./csv/fiches_client.csv")
commandes=pandas.read_csv("./csv/fiches_com.csv")
cl_com=pandas.merge(client, commandes)
print(cl_com)
```

21) Fusionner deux fichiers CSV, pour ne former qu'un seul tableau et afficher le résultat directement dans le terminal Python (deuxième méthode):

Code :

```
import pandas
client=pandas.read_csv("./csv/fiches_client.csv")
commandes=pandas.read_csv("./csv/fiches_com.csv")
com_cl=pandas.merge(commande, client)
print(com_cl)
```

Sortie :



```
analyse.py x
1 import pandas
2 client=pandas.read_csv("./csv/fiches_client.csv")
3 commandes=pandas.read_csv("./csv/fiches_com.csv")
4 cl_com=pandas.merge(client, commandes)
5 print(cl_com)

Shell x
>>> %Run analyse.py
   n_client  nom      prenom  ville  n_commande  date
0      1212  Lacasse  Aubrey   Annecy         1989  04/12/18
1      1343  Primeau  Angelette  Tours         1111  22/09/18
2      2454  Gabriaux   Julie  Bordeaux          45  02/02/12
3      2324   Jobin   Dorene  Bourges          454  08/07/14
4         34  Boncoeur    Kari   Nantes        2323  20/03/19
5      3435  Chrétien  Adèle   Moulin        2122  12/02/19
6      5565  Neufville   Ila  Toulouse          987  12/09/15
7      5565  Neufville   Ila  Toulouse          123  04/11/13
8      2221  Larivière  Alice   Tours         1567  05/12/18
```


22) On constate que Mme Élodie Gaulin (n° de client 895) bien que présente dans le tableau "client", est absente du tableau "cl_com", je pense que cela s'explique par le fait que son numéro client "n_client" qui est 895, n'apparaît pas dans le tableau des commandes, en théorie cela signifie que l'on ne peut pas fusionner cette cliente avec le tableau des commandes, en pratique cela signifie tout simplement que Mme Élodie Gaulin n'a pas passée de commandes !

23) On constate aussi qu'il n'y a aucune trace de la commande n° 1324 du 01/02/2017 dans le tableau "cl_com", cela s'explique par le fait que si on regarde bien, cette commande a été passée par le client n°4444 sauf que ce client n'apparaît pas dans le tableau "client" donc impossible d'obtenir des informations sur lui.