# TP 2 SOL: Explorer et analyser des données avec Python sur un environnement Linux

## Objectifs :

- L'objectif de ce TP est de vous familiariser avec le processus d'exploration et de nettoyage de données en utilisant Python sur Linux. Vous travaillerez avec un ensemble de données pour identifier et corriger les incohérences, les valeurs manquantes, et les anomalies, puis vous appliquerez des techniques de base d'analyse de données.

## Modalités pédagogiques :

**Travail individuel**

**Durée : 1 jour**

> ⚠️ **Essayez d'utiliser ChatGPT comme dernière source d'information**

## Pré-requis :

- VsCode ou Pycharm installé sur linux
- Bibliothèques Python `pandas` et `matplotlib` installées. Vous pouvez les installer via pip :

```
pip install pandas matplotlib
```

- Un fichier CSV : `Mental Health Dataset.csv` à télécharger depuis le lien ci-dessous :

https://www.kaggle.com/datasets/bhavikjikadara/mental-health-dataset

## Consignes :

### Partie 1 : Chargement et Exploration des Données

- Utilisez `pandas` pour charger les données à partir du fichier CSV dans un DataFrame en suivant ces étapes :
  - Déplacez le .csv dans le même répertoire où se trouve votre projet python.
  - Utilisez le code suivant dans un script Python :

```
import pandas as pd

# Chemin relatif (le fichier est dans le même dossier que le script)
csv_file_path = 'Mental Health Dataset.csv'

# Chargement des données
data = pd.read_csv(csv_file_path)

# Aperçu
print(data.head())
```

  - Expliquez chaque ligne du script ci-dessus.
  - Expliquez les résultats affichés à l'écran.

pd : alias pour dire biblio pandas

read_csv : fonction de la biblio pandas pour lire un csv et le charge sous forme dataframe (tableau)

print : affiche les 5 1ère ligne du dataframe

print(df.head(10)) : pour afficher les 10 1ère ligne ; df c'est le nom de la variable data frame par ex

**Opérations de base:**

- Calculer le total des personnes (femmes et hommes) concernés par cette étude en se basant sur les filtres ci-dessous:
  - **Filtre 1** : basées aux USA
  - **Filtre 2**: Change Habits (No)
  - **Filtre 3**: Mood swings (High)
  - **Filtre 4**: Work Interest (Maybe)
  - **Filtre 5**: Social weakness (Yes)

```
import pandas ad pd

csv_file_path = 'Mental Health Dataset.csv

data = pd.read_csv(csv_file_path)

filtres = (
            (data['Country'] == 'United States') &
            (data['Changes_Habits'] == 'No') &
            (data['Mood swings'] == 'High') &
            (data['Work Interest'] == 'Maybe') &
            (data['Social weakness'] == 'Yes')
            )

filtered_data = data[filtres]

total_personnes = len(filtered_dat)

nb_femmes = len(filtered_data[filtered_data['Gender'] == 'Female']
nb_hommes = len(filtered_data[filtered_data['Gender'] == 'Male']


printf("Total des personnes correspondnats aux filtres:", total_personnes)
printf("Total des femmes correspondants aux filtres:", nb_femmes)
prontf("Total des hommes correspondants aux filtres:", nb_hommes)
```

**Explication** :

`data['Country']` ⇒ on dit à Python retourne uniquement les valeurs de la colonne Country avec le filtre sur USA

`filtered_data = data[filtres]` ⇒ on dit à Python de retourner les valeurs à partir de la variable filtres qu'on a défini plus haut;

`total_personnes = len(filtered_data)` : len c'est une fonction Python qui compte combien de lignes ont passé tous les filtres, c'est à dire elle nous donne le nombre de personne qui répond aux 5 filtres, par ex elle peut retourner une valeur : 43 personnes qui répondent exactement aux 5 filtres

## Partie 2 : Filtrage des données et calculs statistiques

**Dataset** :

StudentsPerformance.csv

- Pour effectuer des calculs statistiques sur une BDD, il faut tout d'abord filtrer les données selon certaines critères. Dans cette partie vous allez effectuer des opérations de statistique de base pour l'analyse de données:

1. **Tendances Centrales:**

- **Moyenne** : La somme des valeurs divisée par le nombre de valeurs.
- **Médiane** : La valeur centrale dans un ensemble de données ordonnées.
- **Mode** : La valeur la plus fréquente dans un ensemble de données
    - En se basant sur le script ci-dessous, adaptez-le pour calculer et afficher la moyenne:

```
import numpy as np
import pandas as pd
csv_file_path='Mental Health Dataset.csv'
data =  pd.read_csv(csv_file_path)
total_females = data[data['Gender'] == 'Female'].shape[0]
print(total_females)
52514
```

- Utilisez une liste de données de votre choix  pour calculer la Médiane et de mode avec des scripts Python, vous pouvez télécharger des Datasets sur `Kaggle` ou utiliser une autre source.

Calcul la **moyenne** de math des filles :

```
import pandas as pd

# Charger le fichier CSV
data = pd.read_csv('StudentsPerformance.csv')

# Filtrer uniquement les filles
females_data = data[data['gender'] == 'female']

# Calculer la moyenne des notes en math
moyenne_math_filles = females_data['math score'].mean()

print("Moyenne des notes en mathématiques chez les filles :", moyenne_math_filles)
```

La médiane : val centrale

```
import pandas as pd

# Charger le fichier CSV
data = pd.read_csv('StudentsPerformance.csv')

# Calcul de la médiane pour chaque type de score
mediane_math = data['math score'].median()
mediane_reading = data['reading score'].median()
mediane_writing = data['writing score'].median()

# Affichage des résultats
print("Médiane des notes en mathématiques :", mediane_math)
print("Médiane des notes en lecture :", mediane_reading)
print("Médiane des notes en écriture :", mediane_writing)
```

Calculer le mode :

- **Mode** : La valeur la plus fréquente dans un ensemble de données

```
import pandas as pd

# Charger le fichier CSV
donnees = pd.read_csv('StudentsPerformance.csv')

# Calcul du mode pour chaque type de score

mode_math = donnees['math score'].mode()
mode_reading = donnees['reading score'].mode()
mode_writing = donnees['writing score'].mode()

# Affichage des résultats
#La méthode .mode() retourne une Series pandas, qui peut contenir plusieurs valeurs si plusieurs scores sont les p
lus fréquents (égalité).
#.tolist() transforme la Series pandas en liste Python classique.


print("Mode des notes en mathématiques :", mode_math.tolist())
print("Mode des notes en lecture :", mode_reading.tolist())
print("Mode des notes en écriture :", mode_writing.tolist())
```

2. **Mesure de Dispersion:**

- **Variance** : c'est une mesure statistique qui décrit à quel point les valeurs d'un ensemble de données sont éloignées de la moyenne (aussi appelée l'espérance mathématique) de cet ensemble. En d'autres termes, elle vous aide à comprendre à quel point les données sont dispersées (éloignées) par rapport à la moyenne.

```
import pandas as pd

# Charger le fichier CSV
donnees = pd.read_csv('StudentsPerformance.csv')

# Calcul de la variance pour chaque type de score
variance_math = donnees['math score'].var()
variance_reading = donnees['reading score'].var()
variance_writing = donnees['writing score'].var()

# Affichage
print("Variance des notes en mathématiques :", variance_math)
print("Variance des notes en lecture :", variance_reading)
print("Variance des notes en écriture :", variance_writing)
```

résultats :

Variance des notes des maths: 229.91899799799796
Variance des notes de lecture: 213.16560460460462
Varaince des notes d ecriture: 230.907991991992

- Plus la variance est **grande**, plus les notes sont **variées, dispersées**.

- ➖ Plus elle est **petite**, plus les notes sont **regroupées autour de la moyenne**.

Les résultats montrent que les notes en mathématiques et en écriture présentent une

**dispersion plus importante** (avec une variance autour de 230) comparées aux notes en lecture (variance d'environ 213).

Cela signifie que les élèves ont des résultats **plus homogènes en lecture**, tandis que les performances sont **plus inégales** en mathématiques et en écriture.

Les notes en lecture sont plus stables, tandis que les notes en maths et en écriture sont plus dispersées.

- **Écart-Type** : **la racine carrée de la variance.** Il s'agit d'une mesure **plus facile à interprét**er, car elle est exprimée dans la même unité que les données originales !

```
import pandas as pd

# Charger le fichier CSV
donnees = pd.read_csv('StudentsPerformance.csv')

# Calcul de l'écart-type pour chaque type de score
ecart_type_math = donnees['math score'].std()
ecart_type_reading = donnees['reading score'].std()
ecart_type_writing = donnees['writing score'].std()

# Affichage
print("Écart-type des notes en mathématiques :", ecart_type_math)
print("Écart-type des notes en lecture :", ecart_type_reading)
print("Écart-type des notes en écriture :", ecart_type_writing)
```

Résultat :

écart type des notes des maths: 15.16308009600945
Écart-type des notes en lecture : 14.600191937252216
Écart-type des notes en écriture : 15.19565701086965

### 3. La corrélation :

C'est un coefficient qui permet de voir si les données sont cohérentes par rapport à d'autres données, par exemple, admettons que vous voulez analyser le temps passé à étudier en dehors des heures de formation et comparer ce temps avec les notes obtenues:

**Coeff=1 : Plus vous révisez chez vous mieux sont vos notes ⇒ Corrélation +**

**Coeff = -1 : Plus vous révisez pire sont vos notes ⇒ Corrélation -**

**Coeff = 0 : Pas de corrélation**

- En se basant sur ce script calculer le coefficient de corrélation sur un dataset de votre choix :

```
#Ajoutons une autre colonne pour l'exemple de corrélation
df['Autre_Valeurs'] = [15, 25, 25, 35, 45, 55]
#Calcul de la corrélation
correlation = df['Valeurs'].corr(df['Autre_Valeurs'])
print(f"Corrélation: {correlation}"
```

On va faire un test  pour comparer deux colonnes numériques pour voir **si elles évoluent ensemble** (corrélation). Par exemple :

Est-ce que les élèves qui ont une bonne **note en lecture** ont aussi une bonne **note en écriture** ?

```
import pandas as pd

# Charger le fichier CSV
donnees = pd.read_csv('StudentsPerformance.csv')

# Calcul du coefficient de corrélation entre lecture et écriture
correlation = donnees['reading score'].corr(donnees['writing score'])

# Affichage
print("Corrélation entre les notes de lecture et d'écriture :", correlation)
```

résultat :

Corrélation entre les notes de lecture et écriture des élèves: 0.9545980771462481

la corrélation est proche de 1 donc **plus un élève est bon en lecture, plus il est aussi performant en écriture**.

## Partie 3: Analyse des données

Faites une analyse sur les études de la santé mentale des femmes et des hommes en se basant sur les tableaux ci-dessous:

### 1- Statistiques pour les femmes :

- Donnez les scripts Python qui permettent de faire les calculs du total **des femmes** concernées par cette étude pour l'année 2014 seulement par pays et par mois.
- Remplissez le tableau par les valeurs trouvées.

| Pays | USA | Poland | Australia | Canada | South Africa | Sweden | Netherla |
|---|---|---|---|---|---|---|---|
| Mois | | | | | | | |
| Août | | | | | | | |
| Septembre | | | | | | | |
| Octobre | | | | | | | |
| Novembre | | | | | | | |

```
import pandas as pd

# Charger les données
data = pd.read_csv('Mental Health Dataset.csv')

# Convertir la colonne Timestamp en format datetime (en format vraie date)
# filtre sur les années dt.year
#filtre sur les mois dt.month
# coerce = S'il y a une valeur que je n'arrive pas à transformer en date,
# NE fais pas d'erreur. Mets simplement une valeur vide spéciale appelée NaT (Not a Time)



data['Timestamp'] = pd.to_datetime(data['Timestamp'], errors='coerce')

# Filtrer : femmes uniquement et année 2014
```

```
#str.lower = les caractères en miniscule pour éviter que les filtres soient
#bloqué à cause des maj


femmes_2014 = data[
    (data['Gender'].str.lower() == 'female') &
    (data['Timestamp'].dt.year == 2014)
]

# Extraire le mois
#dt.strftime('%B) = transforme la date en texte sous format "august, septemeber..)
#il existe plusieurs variantes :
# %b : Aug - Setp
# %m : 08
# %Y : 2014


femmes_2014['Month'] = femmes_2014['Timestamp'].dt.strftime('%B')

# Définir la liste des pays demandés
pays = ['United States', 'Poland', 'Australia', 'Canada', 'South Africa', 'Sweden', 'Netherlands']

# Filtrer sur les pays d'intérêt
femmes_filtrees = femmes_2014[femmes_2014['Country'].isin(pays)]

# Compter le nombre par Mois et Pays
resultat = (
    femmes_filtrees
    .groupby(['Month', 'Country'])
    .size()
    .unstack(fill_value=0)
    .reindex(index=['August', 'September', 'October', 'November'], fill_value=0)
    .reindex(columns=pays, fill_value=0)
)

# Affichage du tableau final
print(resultat)
```

## 2- Statistiques pour les hommes :

- Donnez les scripts Python qui permettent de faire les calculs du total **des hommes** concernés par cette étude pour l'année 2014 seulement par pays et par mois.
- Remplissez le tableau par les valeurs trouvées.

| Pays | USA | Poland | Australia | Canada | South Africa | Sweden | Netherla |
|---|---|---|---|---|---|---|---|
| Mois | | | | | | | |
| Août | | | | | | | |
| Septembre | | | | | | | |
| Octobre | | | | | | | |
| Novembre | | | | | | | |

## 3- Filtres:

- Donnez les scripts Python qui permettent d'effectuer des filtres en se basant sur les tableaux ci-dessous pour **l'année 2015 pour les hommes uniquement** et remplissez les cases par les valeurs trouvées.

**Self-Employed: (Yes)**

| Pays | USA | Poland | Australia | Canada | South Africa | Sweden | Netherla |
|---|---|---|---|---|---|---|---|
| Mois | | | | | | | |
| Août | | | | | | | |
| Septembre | | | | | | | |
| Octobre | | | | | | | |
| Novembre | | | | | | | |

```
import pandas as pd

# Charger les données
data = pd.read_csv('Mental Health Dataset.csv')

# Convertir la colonne Timestamp en format datetime
data['Timestamp'] = pd.to_datetime(data['Timestamp'], errors='coerce')

# Filtrer : hommes uniquement, année 2015, et self-employed == 'Yes'
hommes_2015 = data[
    (data['Gender'].str.lower() == 'male') &
    (data['Timestamp'].dt.year == 2015) &
    (data['self_employed'] == 'Yes')
].copy()

# Extraire le mois sous forme textuelle (ex : "August")
hommes_2015['Month'] = hommes_2015['Timestamp'].dt.strftime('%B')

# Liste des pays à analyser
pays = ['United States', 'Poland', 'Australia', 'Canada', 'South Africa', 'Sweden', 'Netherlands']

# Filtrer par pays et grouper par mois + pays
resultat = (
    hommes_2015[hommes_2015['Country'].isin(pays)]
    .groupby(['Month', 'Country'])
    .size()
    .unstack(fill_value=0)
    .reindex(index=['August', 'September', 'October', 'November'], fill_value=0)
    .reindex(columns=pays, fill_value=0)
)

# Afficher le tableau final
print(resultat)
```

**Self-Employed: (No)**

| Pays | USA | Poland | Australia | Canada | South Africa | Sweden | Netherla |
|---|---|---|---|---|---|---|---|
| Mois | | | | | | | |

| Pays | USA | Poland | Australia | Canada | South Africa | Sweden | Netherla |
|---|---|---|---|---|---|---|---|
| Août | | | | | | | |
| Septembre | | | | | | | |
| Octobre | | | | | | | |
| Novembre | | | | | | | |

**Family history: (Yes)**

| Pays | USA | Poland | Australia | Canada | South Africa | Sweden | Netherla |
|---|---|---|---|---|---|---|---|
| Mois | | | | | | | |
| Août | | | | | | | |
| Septembre | | | | | | | |
| Octobre | | | | | | | |
| Novembre | | | | | | | |

**Family history: (No)**

| Pays | USA | Poland | Australia | Canada | South Africa | Sweden | Netherla |
|---|---|---|---|---|---|---|---|
| Mois | | | | | | | |
| Août | | | | | | | |
| Septembre | | | | | | | |
| Octobre | | | | | | | |
| Novembre | | | | | | | |

## Livrables attendus

- Script Python complet avec :
  - chargement du dataset
  - Le filtrage
  - calculs statistiques (moyenne, médiane, mode, variance, corrélation)
- Fichier texte avec explication des résultats
- Tableaux de la section 3 : Analyse des données