S1J3 : Manipulation des données.

Objectif :

Apprendre à manipuler et transformer des données tabulaires, car en tant que Data Engineer, vous serez souvent amené à travailler avec des ensembles de données bruts nécessitant des nettoyages, des restructurations ou des agrégations avant de les charger dans des bases ou pipelines.

---

1. Les bases de Pandas.

Créer et lire des DataFrames.

Comprendre ce qu'est un DataFrame et comment il représente des données tabulaires (lignes et colonnes).

Charger des fichiers CSV, Excel, JSON dans un DataFrame

```
import pandas as pd
df = pd.read_csv("data.csv")
```

Comprendre les formats de données (`.head()` , `.info()` , `.describe()`).

Explorer et sélectionner des données.

Accéder à des colonnes ou des lignes spécifiques avec `.loc[]` et `.iloc[]` .

```
# Sélectionner une colonne
df['colonne']

# Sélectionner des lignes
df.loc[0]  # Par index
df.iloc[0]  # Par position
```

Comprendre les types de données.

Vérifier et convertir les types (`df.dtypes` , `astype()`).

---

2. Nettoyage des données.

En tant que Data Engineer, une grande partie de votre travail sera de traiter des données "sales".

Traiter les valeurs manquantes

Vérification : `df.isnull().sum()`

Remplissage ou suppression :

```
df.fillna(0, inplace=True)  # Remplir par 0
df.dropna(inplace=True)     # Supprimer les lignes avec NaN
```

Supprimer ou renommer des colonnes.

```
df.drop('colonne_a_supprimer', axis=1, inplace=True)
df.rename(columns={'ancien_nom': 'nouveau_nom'}, inplace=True)
```

Détecter et supprimer les doublons.

```
df.drop_duplicates(inplace=True)
```

---

3. Transformation des données.

Un Data Engineer prépare les données dans le bon format pour les pipelines ou l'analyse.

Créer de nouvelles colonnes.

```
df['nouvelle_colonne'] = df['colonne1'] + df['colonne2']
```

Filtrer les données.

```
df_filtre = df[df['colonne'] > 10]
print(df_filtre)
```

Appliquer des transformations.

Appliquer une fonction personnalisée :

```
df['colonne'] = df['colonne'].apply(lambda x: x * 2)
```

---

4. Agrégation et regroupement.

Ces opérations permettent de résumer ou structurer les données pour une analyse approfondie.

GroupBy pour l'agrégation.

```
df_grouped = df.groupby('colonne_categorie').mean()
```

Pivot Table.

Utile pour restructurer les données.

```
df.pivot_table(index='colonne1', columns='colonne2', values='colonne3', aggfunc='sum')
```

---

5. Sauvegarde des données.

Une fois les données nettoyées et transformées, vous devrez souvent les sauvegarder dans des formats spécifiques.

Exporter un DataFrame

```
df.to_csv("fichier_propre.csv", index=False)
df.to_excel("fichier_propre.xlsx", index=False)
```

Exercice pratique (Projet mini).

Voici un exercice simple pour consolider les concepts :

Charger un fichier CSV (par exemple, un fichier contenant des informations de ventes).

Nettoyer les données :

Supprimez les doublons.

Remplissez les valeurs manquantes.

Transformer les données :

Ajoutez une colonne "revenu total" (quantité * prix unitaire).

Filtrez les ventes où le revenu est supérieur à un seuil.

Agrégation :

Trouvez le revenu total par mois.

Exporter le résultat dans un nouveau fichier CSV.

Pourquoi c'est important pour un Data Engineer ?

Nettoyage et préparation des données : Les données brutes doivent souvent être nettoyées et réorganisées avant d'être utilisées dans des pipelines ou des analyses.

Manipulations complexes : Pandas vous permet d'effectuer rapidement des manipulations qui seraient fastidieuses avec SQL ou d'autres outils.

Étape essentielle dans les pipelines ETL : Les transformations dans Pandas constituent souvent une étape intermédiaire avant de charger les données dans une base ou un entrepôt.