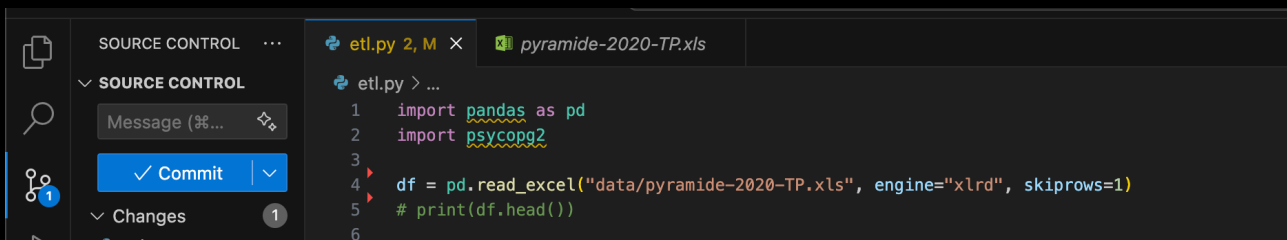


LAB : Mettre en place un ETL et visualiser les données

Étape 1 : Extraction des données

Pour ce lab, j'ai commencé par lire le fichier excel grâce à la bibliothèque Pandas que j'ai ajouté en tant que dépendance.

A screenshot of a code editor interface. On the left, there is a 'SOURCE CONTROL' panel with a 'Commit' button. The main editor area shows a Python file named 'etl.py' with the following code:

```
1 import pandas as pd
2 import psycopg2
3
4 df = pd.read_excel("data/pyramide-2020-TP.xls", engine="xlrd", skiprows=1)
5 # print(df.head())
6
```

J'ai passé l'en-tête du fichier qui ne m'était pas utile pour la suite du travail

Étape 2 : Transformation des données

Une fois les données extraites, j'ai effectué plusieurs transformations pour préparer le DataFrame pour l'insertion dans la base de données :

- Converti les années en entiers
- Supprimé les lignes vides
- Calculé la somme des hommes et des femmes pour chaque année et stockée dans une nouvelle colonne **Total**

```

df = df[df["Annee"].astype(str).str.isnumeric()]
df["Annee"] = df["Annee"].astype(int)

df = df.dropna()
df["Annee"] = df["Annee"].astype(int)
df["Nombre_Hommes"] = df["Nombre_Hommes"].astype(int)
df["Nombre_Femmes"] = df["Nombre_Femmes"].astype(int)

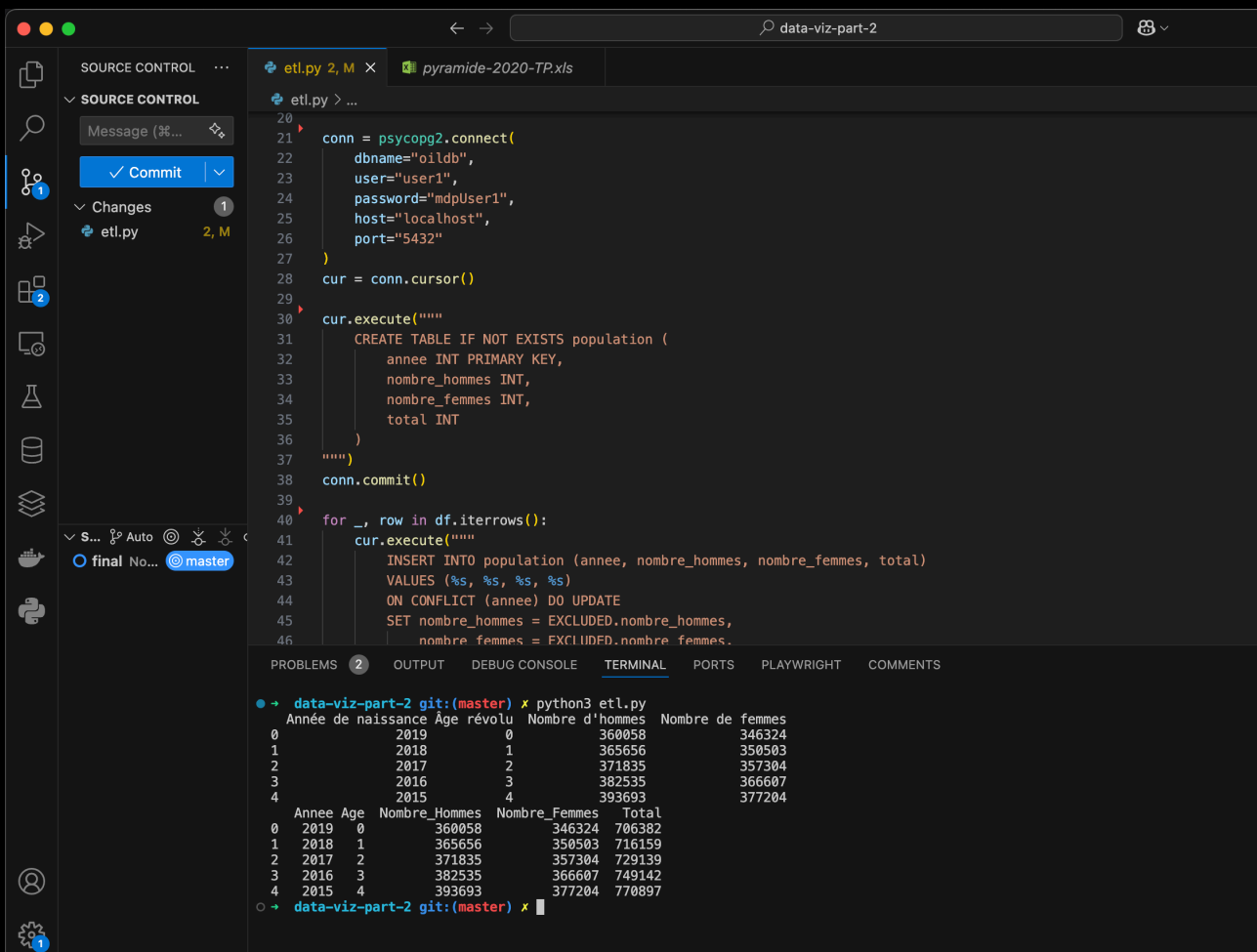
df["Total"] = df["Nombre_Hommes"] + df["Nombre_Femmes"]

print(df.head())

```

Étape 3 : Chargement des données dans la BDD

J'ai créé une table **population** dans la base de données **PostgreSQL** et inséré les données. En cas de conflit (si l'année existe déjà par exemple), j'ai mis à jour les valeurs existantes.



The screenshot shows a VS Code editor with a Python script named `etl.py` and its output in the terminal. The script connects to a PostgreSQL database, creates a table named `population`, and inserts data from a DataFrame. The terminal output shows the execution of the script and the resulting data.

```

20
21 conn = psycopg2.connect(
22     dbname="oildb",
23     user="user1",
24     password="mdpUser1",
25     host="localhost",
26     port="5432"
27 )
28 cur = conn.cursor()
29
30 cur.execute("""
31     CREATE TABLE IF NOT EXISTS population (
32         annee INT PRIMARY KEY,
33         nombre_hommes INT,
34         nombre_femmes INT,
35         total INT
36     )
37 """)
38 conn.commit()
39
40 for _, row in df.iterrows():
41     cur.execute("""
42         INSERT INTO population (annee, nombre_hommes, nombre_femmes, total)
43         VALUES (%s, %s, %s, %s)
44         ON CONFLICT (annee) DO UPDATE
45         SET nombre_hommes = EXCLUDED.nombre_hommes,
46             nombre_femmes = EXCLUDED.nombre_femmes.

```

The terminal output shows the execution of the script and the resulting data:

```

data-viz-part-2 git:(master) x python3 etl.py
Année de naissance Âge révolu Nombre d'hommes Nombre de femmes
0 2019 0 360058 346324
1 2018 1 365656 350503
2 2017 2 371835 357304
3 2016 3 382535 366607
4 2015 4 393693 377204

```

Année	Age	Nombre_Hommes	Nombre_Femmes	Total
0	2019	0	360058	346324
1	2018	1	365656	350503
2	2017	2	371835	357304
3	2016	3	382535	366607
4	2015	4	393693	377204

Étape 4 : Visualisation avec Grafana

J'ai lancé le service Grafana branché sur le port 3000, ajouté PostgreSQL en tant que data source puis créé un dashboard pour visualiser l'évolution de la population.



Ci-dessus la query exécutée.



Enfin, sur ce graphique, l'évolution de la population à travers le temps (Homme et femme).

