PARTIE 1 :

Objectif : construire une data pipeline permettant de traiter les données des médicaments et leurs publications dans des journaux.

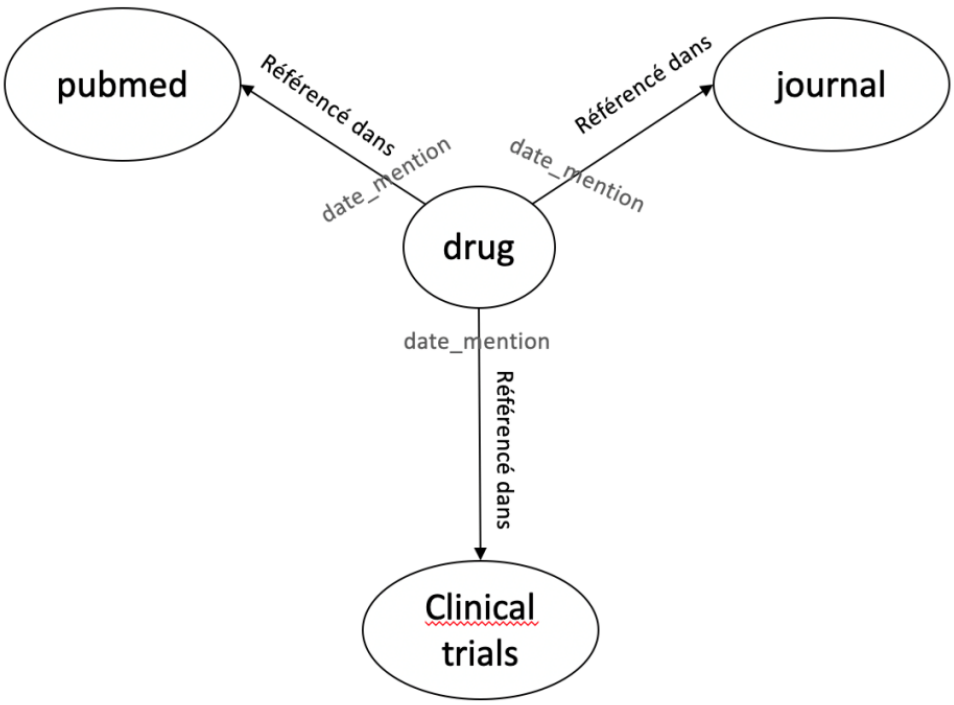
Travaux réalisés :

I. Pipeline

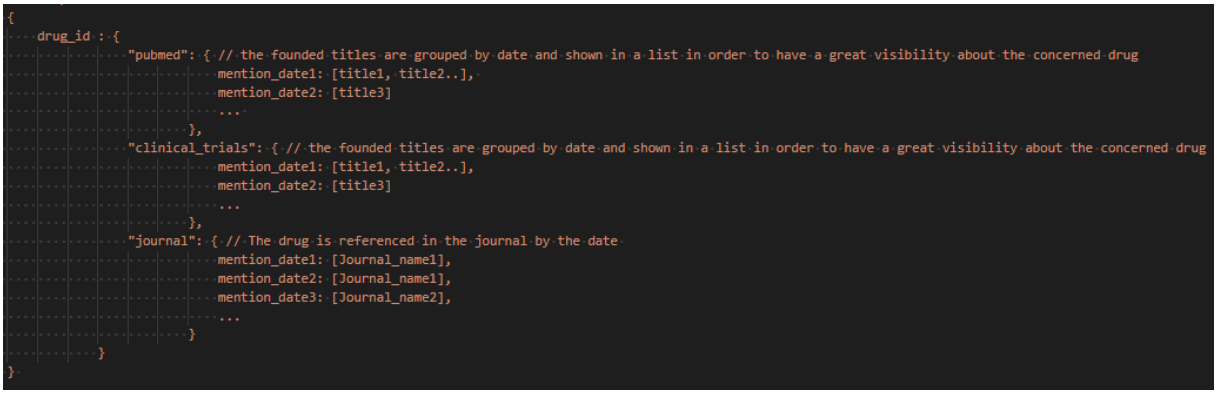
Le projet a été développé en utilisant python comme langage de programmation. Les dépendances du projet ont été mentionnées dans le fichier requirements.txt.

Le pipeline conçu pour ce projet est composé de :

1. Job1 : Extraction des données
    - Extraire les données en partant de plusieurs fichiers et différents formats (JSON / CSV…).
    - Concaténer les données
2. Job2 : Nettoyage des données
    - Supprimer les lignes avec des valeurs manquantes
    - Normaliser le format de la colonne date dans les tables Clinical trials et Pubmed
    - Supprimer les charactères ignorés suite à un défaut d'encodage de la source
3. Job3 : Manipulation des données
    - En se basant sur le graphe suivant :



- Je boucle sur la table des médicaments et je cherche le nom du médicament dans les titres des publications dans Pubmed et Clinical trials.
- Les clés principales de notre dictionnaire de sortie sont drug_ids. En second niveau, nous avons les entités liées aux médicaments qui sont Pubmed, Clinical_trials et journal. Dans pubmed et clinic_trials, nous retrouverons dans le JSON les titres des publications référencées par date. Enfin, nous aurons en sortie les journaux associées à ces publications référencées également par date.
- Cette modélisation a été traduite par le schéma JSON suivant :



II. Ad-hoc

La partie ad hoc est une sorte d'exploitation de la sortie du pipeline de données. elle est basée sur la sortie JSON et consiste à renvoyer le journal qui mentionne le plus de médicaments différents.

III. Pour aller plus loin

Quels sont les éléments à considérer pour faire évoluer votre code afin qu'il puisse gérer de grosses volumétries de données (fichiers de plusieurs To ou millions de fichiers par exemple) ?

Afin de gérer de grosses volumétries de données, je propose passer au stockage/calcul distribués des données avec Hadoop ou bien Spark.

Pourriez-vous décrire les modifications qu'il faudrait apporter, s'il y en a, pour prendre en considération de telles volumétries ?

En utilisant Apache Spark, on remplace Pandas Dataframe par PySpark Dataframe, il y a une légère différence entre les méthodes de chacun de ses dataframes :

Pandas	Spark
Pandas.DataFrame()	Spark.createDataFrame()
Pandas.read_csv	Spark.read.csv()
Pandas.concat([df1,df2])	df1.union(df2)
to_datetime()	to_timestamp()
DataFrame.str.replace()	DataFrame.na.replace()
df[‘col’].str.contains('anystring_to_match')	df['col'].like('anystring_to_match')
df.groupby()	df.groupBy()

PARTIE 2 :

Objectif : Réaliser des requêtes SQL claires et facilement compréhensibles.

1ère requête :

```
SELECT date, SUM(prod_price*prod_qty) AS ventes FROM transaction

WHERE date BETWEEN '2019-01-01' AND '2019-12-31'

GROUP BY date;
```

2ème requête :

```
SELECT transaction.client_id,
        SUM(CASE WHEN product_nomenclature.product_type = 'MEUBLE' THEN
transaction.prod_price*transaction.prod_qty ELSE 0 END) AS ventes_meubles,
        SUM(CASE WHEN product_nomenclature.product_type = 'DECO' THEN
transaction.prod_price*transaction.prod_qty ELSE 0 END) AS ventes_deco

FROM transaction

JOIN product_nomenclature ON transaction.prod_id =
product_nomenclature.product_id

WHERE transaction.date BETWEEN '2019-01-01' AND '2019-12-31'

GROUP BY transaction.client_id;
```