**Question 1: 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) ?**

Pour faire évoluer le code afin qu'il puisse gérer de grosses volumétries de données, tels que des fichiers de plusieurs téraoctets (To) ou des millions de fichiers, on devrait tenir compte de plusieurs éléments :

1. Optimisation de la Mémoire

* Évitez de charger l'ensemble des données en mémoire si cela n'est pas nécessaire. Utilisez la lecture en streaming (par exemple, avec des itérateurs) pour traiter les données par petits morceaux.
* Utilisez des structures de données efficaces pour minimiser l'utilisation de la mémoire.

2. Parallélisme et Distribution ( Diviser le traitement en tâches indépendantes pouvant être exécutées en parallèle pour accélérer le traitement).

3. Compression de Données ( Utiliser des formats de données compressés pour réduire la taille des fichiers de données sans perte de qualité).

4. Indexation et Partitionnement

5. Scalabilité ( Concevoir le code pour qu'il puisse évoluer horizontalement en ajoutant des ressources ou des noeuds de calcul au fur et à mesure que les données augmentent).

**Question 2 : Pourriez-vous décrire les modifications qu’il faudrait apporter, s’il y en a, pour prendre en considération de telles volumétries ?**

Les modifications à apporter, pour prendre en considération de telles volumétries sont:

* Lecture en Streaming: Au lieu de charger l’intégralité des données en mémoire à partir des Data Frames drugs\_df, pubmed\_df et clinical\_trials\_df, il conviendra de lire et traiter les données par petits morceaux à la fois, ce qui réduirait considérablement la consommation de mémoire.
* Utiliser des formats de données compressés, tels que le Parquet, pour stocker et lire les données. Cela réduira la taille des fichiers de données sur le disque.
* Diviser les tâches en sous-tâches plus petites et parallèles pour tirer parti de la capacité de calcul disponible.
* Créer des index dans le DataFrames pour accélérer les recherches.
* Mettre en place des mécanismes de surveillance et de journalisation pour suivre les performances en temps réel et diagnostiquer les problèmes.