Etude comparative (Avantages et Incovenients) sur les formats des fichiers parquet, orc, avro, apache arrow

REALISER PAR: Mohamed El Houssein Cheikh

Classe : ING3 INFO

1. Parquet

Avantages :

- **Colonnes** : Parquet est un format de stockage en colonnes, ce qui signifie qu'il est particulièrement efficace pour les requêtes analytiques qui n'ont besoin que d'un sous-ensemble des colonnes d'une table.
- **Compression** : Il offre des taux de compression élevés grâce à une compression basée sur les colonnes, réduisant la taille des données stockées.
- **Compatibilité** : Très bien intégré avec les écosystèmes Hadoop, Spark, Hive, et d'autres outils Big Data.
- **Efficacité des requêtes** : Parquet permet des lectures rapides pour les charges de travail analytiques, en accédant uniquement aux colonnes nécessaires, ce qui réduit le volume de données à lire.

Inconvénients :

- **Complexité de l'écriture** : La structure par colonnes peut rendre l'écriture des données plus complexe et plus lente, en particulier pour les charges de travail transactionnelles ou de flux.
- **Flexibilité limitée** : Parquet est moins flexible pour les données semi-structurées ou non structurées par rapport à des formats comme Avro.

2. ORC (Optimized Row Columnar)

Avantages :

- **Colonnes** : Comme Parquet, ORC est un format de stockage en colonnes, ce qui offre des avantages similaires en termes d'efficacité pour les requêtes analytiques.
- **Compression** : ORC offre une compression de haute qualité avec des options de compression avancées comme Zlib et Snappy, ce qui réduit l'espace de stockage nécessaire.
- **Optimisation pour Hadoop** : ORC est particulièrement optimisé pour Hadoop, notamment pour les systèmes utilisant Hive, offrant des performances de lecture/écriture optimales.

Inconvénients :

- **Ecriture lente** : L'écriture des données dans le format ORC peut être lente, ce qui peut être un inconvénient pour les cas d'utilisation nécessitant des écritures fréquentes ou en temps réel.
- **Format plus lourd** : ORC peut être plus lourd en termes de métadonnées, augmentant la taille globale des fichiers dans certains cas.

3. Avro

Avantages :

- **Format en ligne** : Avro stocke les données sous un format en ligne (row-based), ce qui le rend bien adapté aux scénarios où les lectures ou écritures en ligne sont fréquentes.
- **Schéma intégré** : Avro stocke le schéma avec les données, permettant une forte interopérabilité et facilitant l'évolution des schémas.
- **Compact** : Le format binaire d'Avro est compact et offre une bonne compression, réduisant l'utilisation de la bande passante et du stockage.
- **Interopérabilité** : Avro est très bien intégré avec divers langages de programmation et systèmes, ce qui le rend polyvalent pour de nombreuses applications.

Inconvénients :

- **Moins efficace pour les analyses** : Comme Avro est basé sur les lignes, il est moins efficace pour les requêtes analytiques massives, où seuls quelques champs sont nécessaires.
- **Complexité des schémas** : La gestion et l'évolution des schémas peuvent devenir complexes, en particulier dans des environnements de données massives.

4. Apache Arrow

Avantages :

- **Format en mémoire** : Apache Arrow est conçu pour le traitement de données en mémoire, ce qui le rend extrêmement rapide pour les opérations analytiques et les traitements en temps réel.
- **Interopérabilité** : Arrow permet l'interopérabilité entre différents langages et systèmes, en facilitant le partage des données en mémoire sans copie supplémentaire.
- **Performance** : Optimisé pour le traitement par vecteurs, Arrow offre des performances supérieures pour les calculs analytiques et les transformations de données en mémoire.
- **Support de GPU** : Arrow est bien adapté pour les opérations sur GPU, ce qui permet d'accélérer les traitements analytiques massifs.

Inconvénients :

- **Pas de stockage natif** : Arrow est principalement un format en mémoire et n'est pas conçu pour le stockage persistant, nécessitant d'autres formats pour le stockage à long terme.
- **Support limité pour les données complexes** : Le support des types de données complexes dans Arrow est moins mature comparé à d'autres formats comme Avro ou Parquet.

Conclusion

Le choix du format dépendra fortement du cas d'utilisation spécifique :

- **Parquet et ORC** sont idéaux pour les requêtes analytiques sur de grands volumes de données où seules certaines colonnes sont nécessaires.
- **Avro** est mieux adapté pour les charges de travail transactionnelles ou de streaming où les écritures fréquentes et l'évolution des schémas sont critiques.
- **Apache Arrow** est le choix idéal pour les analyses en mémoire haute performance, en particulier dans les environnements de calculs distribués ou utilisant des GPU.