REPUBLIQUE DU SENEGAL

Un peuple –un but –une foi

MINISTERE DE L'ENSEIGNEMENT SUPERIEURE DE LA RECHERCHE ET DE L'INNOVATION

DIRECTION GENERALE DE L'ENSEIGNEMENT SUPERIEUR

Université Gaston Berger

UFR Institut Polytechnique de Saint-Louis

Étude comparative (pros & cons) sur les formats de fichiers parquet, orc, avro, apache arrow

Présenté par :

Bassirou SAGNA

ING 3 Info-Telecom

Sous la direction de :

Dr. Djibril MBOUP

Plan :

Introduction :

Dans le domaine du Big Data et de l'analyse des données, le choix de la bonne forme de fichier est une décision équivalant à de considérables implications sur les performances des systèmes, l'efficacité du stockage et la facilité d'utilisation des données. Les principaux formats de fichiers appréciés par les professionnels du Big Data sont Parquet, ORC, Avro et Apache Arrow. Tous ces formats ont leurs propres caractéristiques et présentent des avantages spécifiques. Chacun de ces formats est conçu pour répondre à des besoins particuliers et s'adapte différemment à diverses applications et environnements.

Ce format de fichier peut, en fait, faire une différence spectaculaire en termes de performance et d'efficacité du stockage et de la facilité d'utilisation. Parmi ces formats, les plus utilisés aujourd'hui sont Parquet, ORC, Avro et Apache Arrow ; avec des caractéristiques très différentes, chacun étant plus optimisé que d'autres en fonction des cas d'utilisation et des environnements.

Cette étude comparative vise à explorer en profondeur les avantages et les inconvénients de ces formats, en fournissant une analyse détaillée pour aider à mieux comprendre les différents types de formats de fichiers leurs utilités mais aussi la manière dont ils sont utilisés en terme de cas pratiques.

Pour une bonne compréhension de notre étude comparative, nous allons en premier lieu apporter une explication sur les formats de fichier paquet, ensuite faire une étude comparative en donnant une brève présentation, ensuite parler des avantages, des inconvénients, mais aussi donner des exemples sur ORC, Avro, et Apache Arrow, apres l'étude comparative nous allons essayer de montrer sur comment se fait le choix du fichier à choisir pour faire son analyse et enfin nous parlerons de la conclusion.

L'objectif de cette étude est de fournir une compréhension approfondie de chaque format de fichier, afin d'avoir un avis éclairé sur le choix du format le plus approprié pour les différents besoins spécifiques. Que ce soit pour optimiser les performances de requêtes analytiques, réduire les coûts de stockage, ou assurer une interopérabilité fluide entre divers systèmes.

Etude comparative :

Les structures standardisées des fichiers sont utilisées pour organiser et mémoriser efficacement les données numériques qui peuvent être lues par les ordinateurs. Ils ont une importance cruciale dans différentes sphères d'activité, notamment le stockage de données, le partage de l'information entre systèmes, et l'analyse à grande échelle des données. Chaque format est créé en fonction de certains buts définis tels que l'optimisation du rationnel espace disque, la rapidité d'accès aux informations ou la facilité d'interopérabilité de différents logiciels et plateformes.

Le choix des formats de fichiers adéquats peut avoir un impact significatif sur les performances des systèmes d'information en termes de Big Data. Les big data nécessitent des solutions de stockage et de traitement où non seulement minimise leur taille sur le disque dur mais permet aussi rapide opérations d'accès à ces dernières. Ces exigences sont prises en compte lors du développement des formats modernes avec leurs caractéristiques avancées en matière de compression, structuration ou encore indexation.

Dans cette etude comparative nous allons parler de différents formats en donnant une bref présentation ensuite parler de leurs avantages et inconvénient pour finir par donner un exemple de cas d'usage. Nous allons de ce fait parler de :

- ✓ **Parquet :** Parquet est un format de fichier de stockage en colonnes open-source pour l'écosystème Hadoop et Spark. Im a été conçu pour permettre le stockage et le traitement des données massives de manière efficace, Parquet permet de réduire la quantité de données lues à partir du disque tout en permettant l'optimisation des performances des requêtes analytiques. Il est largement utilisé dans les systèmes de Big Data pour son efficacité en termes de compression et de lecture

    - **Avantages :**

        - Stockage : Parquet stocke les données par colonnes plutôt que par lignes. Cela est d'autant plus efficace car cela signifie que lors de la lecture de certaines colonnes, seules les parties nécessaires du fichier sont lues, ce qui améliore les performances en d'autre terme Parquet facilite un accès rapide aux informations en réduisant ainsi considérablement les temps de lecture

- Compression : Le format en colonnes permet une meilleure compression performante des données car les données similaires sont stockées ensemble tout en permettant de minimiser l'espace de stockage nécessaire et de réduire les coûts associés. Parquet utilise plusieurs algorithmes de compression, comme Snappy et Gzip, pour réduire la taille des fichiers.

- Gestion : La flexibilité de Parquet lui offre la possibilité d'avoir une très bonne adaptabilité aux différents types de données.

- Compatibilité : Intégré nativement à l'écosystème Hadoop, Parquet profite d'une large adoption ce qui lui permet d'être compatible avec de nombreux outils de Big Data comme Apache Hive, Apache Drill, Apache Impala, Apache Spark et Presto. Cette compatibilité facilite son intégration dans des workflows de données complexes.

- Optimisation : En raison de son format de stockage en colonnes, Parquet permet des lectures rapides et efficaces, ce qui est crucial pour les analyses de grandes quantités de données.

- **Inconvénients :**

  - Complexité : L'écriture de données dans le format Parquet peut être plus complexe et consommatrice en raison de la nécessité de gérer la structure en colonnes.

  - Latence : Parquet est optimisé pour les lectures de grandes quantités de données. Pour des transactions de lecture ou écriture de petites tailles, il peut introduire une latence plus élevée par rapport à d'autres formats autrement dit sur de simple requête, Parquet peut s'avérer moins performant.

  - Taille des Fichiers : La gestion de schémas complexes peut avoir un impact sur les performances, nécessitant une attention particulière lors de la conception des structures de données.

- Difficile d'appliquer les mises à jour, sauf si vous supprimez et recréez à nouveau le fichier.
- Non lisible par l'homme

- **Cas d'Usage:**
  - Parquet est idéal pour les tâches analytiques nécessitant des lectures sélectives de colonnes sur des dataset assez conséquent c'est-à-dire volumineux pour être plus performant.

✓ **ORC :** Le format ORC (Optimized Row Columnar) est un format de fichier optimisé pour le stockage en colonnes, spécialement conçu pour l'écosystème Hadoop. Il s'impose comme un format de fichier performant et flexible et permet d'améliorer les performances de stockage et de traitement dans Hadoop, ORC permet une compression de données efficace et des opérations de lecture et écriture rapides. Ce format est particulièrement adapté pour les grands strusture et volume de données où l'efficacité et la vitesse sont cruciales.

- **Avantages :**
  - Compression : ORC utilise des techniques de compression avancées, telles que Zlib et Snappy, ce qui permet de réduire de manière significative la taille des fichiers originales jusqu'à 75% et, par conséquent, les coûts de stockage.
  - Optimisation : En stockant les données en colonnes, ORC permet des opérations de lecture plus rapides et plus efficaces, surtout pour les requêtes analytiques qui nécessitent l'accès à un sous-ensemble de colonnes.
  - Tolérance : Grâce à sa structure interne robuste, ORC offre une meilleure tolérance aux pannes et une récupération plus facile des données corrompues.
  - Types de Données : Il prend en charge une large gamme de types de données mais aussi les types complexes comme les structures imbriquées, les cartes et les listes.

- **Inconvénients :**
  - Impossibilité d'ajouter des données sans avoir à recréer le fichier.
  - Compatibilité limitée.
  - Consommation : la consommation de mémoire lors des opérations de lecture et d'écriture nécessite plusieurs ressources, ce qui peut être un inconvénient pour les systèmes avec des ressources limitées.
  - Assez complexe.
- **Cas d'usage :**
  - ORC est souvent utilisé dans les environnements Hadoop et Hive où la performance et la compression sont cruciales.

✓ **Avro :** Avro est un framework développé dans le cadre du projet Apache Hadoop. Il s'agit d'un format de stockage basé sur des lignes qui est largement utilisé comme processus de sérialisation. AVRO stocke son schéma au format JSON, ce qui le rend facile à lire et à interpréter par n'importe quel programme. Les données elles-mêmes sont stockées au format binaire en les rendant compactes et efficaces. Une caractéristique clé d'AVRO est liée au fait qu'il gère facilement l'évolution des schémas. Il joint des métadonnées à leurs données dans chaque enregistrement.

- **Avantages :**
  - Avro est un système de sérialisation de données.
  - Il est divisible (AVRO a un marqueur de synchronisation pour séparer le bloc) et compressible.
  - Avro est un bon format de fichier pour l'échange de données. Il dispose d'un stockage de données qui est très compact, rapide et efficace pour l'analyse.
  - Il prend fortement en charge l'évolution du schéma (à différents moments et indépendamment).
  - Il supporte le batch et et particulière approprié au streaming.
  - Les schémas sont définis en JSON, facile à lire et à interpréter.
  - Les données sont toujours accompagnées d'un schéma qui permet un traitement complet des données.

- **Inconvénients :**
  - En étant binaire, les données ne sont pas lisibles par un humain.
  - Non intégré à tous les langages.
  - Compatibilité : Avro est moins couramment utilisé au sein de l'écosystème Hadoop que Parquet et ORC, ce qui peut limiter son intégration.
- **Cas d'usage** : Avro est particulièrement adapté pour les échanges de données et la sérialisation dans les systèmes distribués et les pipelines de traitement de données.

✓ **Arrow :** Apache Arrow est une bibliothèque pour le traitement des données en mémoire. Elle vise à accroître la vitesse et les performances des tâches de traitement des données en fournissant une représentation des colonnes en mémoire. Contrairement à d'autres formats de fichiers qui se concentrent sur le stockage persistant, la conception d'Arrow améliore l'efficacité des calculs et des échanges de données en mémoire entre différents systèmes et langages de programmation. Cela se produit sans qu'il soit nécessaire de convertir ou de sérialiser les données.

- **Avantages :**
  - Vitesse de traitement : Arrow est conçu pour être très rapide. Il exploite au maximum l'accès à la mémoire et réduit la nécessité de changements de format coûteux entre les différentes étapes du traitement des données.
  - Collaboration : Arrow facilite le partage des données entre différents langages de codage et systèmes sans avoir à les emballer et à les déballer, ce qui est souvent le cas avec d'autres formats de stockage.
  - Disposition des colonnes : Comme Parquet, Arrow utilise une structure en colonnes. Cela permet d'améliorer considérablement l'analyse des données lorsque vous avez besoin de certaines colonnes.
  - Nombreux langages : Arrow dispose de bibliothèques pour plusieurs langages de codage, notamment Python, Java C++ et R. Il est donc facile à utiliser dans divers contextes de développement.
- **Inconvénients :**
  - Utilisation de la mémoire : Arrow est optimisé pour le traitement en mémoire, ce qui signifie qu'il peut utiliser beaucoup de mémoire lorsqu'il s'agit de données très volumineuses.

- Manque de persistance : Contrairement à Parquet, Arrow n'est pas conçu pour le stockage de données à long terme. Il sert à traiter et à transférer les données en mémoire.
- Complexité de l'intégration : L'intégration d'Arrow dans une configuration existante peut nécessiter des changements importants si les systèmes actuels ne prennent pas en charge la représentation de la mémoire en colonnes.

- **Cas d'usage :** Arrow est idéal pour les applications nécessitant un traitement rapide en mémoire, comme les calculs analytiques en temps réel et l'ETL (Extract, Transform, Load) en mémoire.

Choix du fichier à choisir :

Le choix du format de fichier le plus approprié pour vos données est une décision cruciale qui impacte directement la gestion efficace. Divers formats présentent des caractéristiques et des avantages distincts, rendant la sélection tributaire de vos besoins spécifiques. Pour ce faire Le choix du format de fichier est crucial car il influence plusieurs aspects clés du traitement et de la gestion des données :

- ✓ **Performance** : Certains formats sont optimisés pour des requêtes rapides et efficaces, ce qui est essentiel pour les analyses de données en temps réel.
- ✓ **Efficacité de Stockage** : Les techniques de compression varient selon les formats, impactant directement les coûts de stockage.
- ✓ **Interopérabilité** : La facilité avec laquelle les données peuvent être échangées entre différents systèmes et applications dépend du format choisi.
- ✓ **Évolutivité** : La capacité à gérer les changements de schéma de données sans affecter les opérations courantes est un facteur déterminant pour des systèmes évolutifs.

Dès lors que nous connaissons les différents aspects qui influent sur les choix du format de fichier nous pouvons entre autre ajouter comment choisir le format de fichier le plus approprié en suivant ces différentes étapes :

- ✓ Évaluer les Besoins de Performance
- ✓ L'Efficacité de Stockage
- ✓ Assurer l'Interopérabilité
- ✓ Gestion des Schémas
- ✓ Analyser les Cas d'Utilisation Spécifiques

Le choix du format de fichier doit être basé sur une évaluation rigoureuse des besoins de performance, d'efficacité de stockage, d'interopérabilité et d'évolutivité. En prenant en compte ces facteurs, on pourra faire le choix sur le format qui pourra permettre d'optimiser les opérations de traitement et de gestion de données.

Conclusion :

Le choix du format de fichier approprié dépend des besoins spécifiques du projet, y compris les exigences de performance, d'interopérabilité et de gestion des données. Parquet et ORC sont recommandés pour les analyses ad hoc et les entrepôts de données massifs, offrant une compression efficace et des performances optimisées. D'autre part, Avro est idéal pour les échanges de données entre systèmes différents et pour la sérialisation des données, tandis qu'Apache Arrow est préféré pour les opérations en mémoire nécessitant des performances élevées. En comprenant les subtilités de chaque format et en tenant compte de leurs points forts et faiblesses, les professionnels peuvent optimiser leurs infrastructures de données et améliorer la rentabilité de leurs solutions analytiques.

Cette étude comparative des formats Parquet, ORC, Avro et Apache Arrow, nous a permis de connaitre les avantages et inconvénients dans le domaine de l'analyse de données et d'applications Big Data.

Dans le cadre de cette étude, nous avons eu à examiner les divers aspects qui influencent le choix du format de fichier. De plus, notre étude a souligné l'importance de la gestion des schémas et de l'évolutivité dans le choix du format de fichier.

En conclusion, le choix du format de fichier doit être guidé par une compréhension approfondie des exigences spécifiques du projet, des caractéristiques distinctes de chaque format, et de leurs forces et faiblesses respectives. En faisant des choix éclairés basés sur cette étude comparative, les professionnels du Big Data peuvent optimiser leurs infrastructures de données, améliorer l'efficacité des analyses, et maximiser la rentabilité de leurs solutions analytiques.