***BigData***

Les concepts de hadoop :

* Hadoop, qui est une plateforme de traitement distribué utilisée pour stocker et traiter de grandes quantités de données, également connue sous le nom de big data. Hadoop englobe un ensemble d'outils qui facilitent le stockage et le traitement de ces données massives.

Voici un aperçu de certains outils :

1. **Hadoop Distributed File System (HDFS) :**
   * HDFS est le système de fichiers distribué de Hadoop, conçu pour stocker de grandes quantités de données sur un cluster de machines.
   * Les données sont divisées en blocs et réparties sur plusieurs nœuds pour un stockage et une récupération efficace.
   * Il est inspiré du Google File System (GFS) et offre une tolérance aux pannes en répliquant les blocs de données.
2. **MapReduce :**
   * MapReduce est un modèle de programmation et de traitement de données pour Hadoop.
   * Il divise une tâche en deux phases : la phase Map, qui effectue des opérations de filtrage et de tri, et la phase Reduce, qui agrège les résultats.
   * Il permet un traitement parallèle sur de grandes quantités de données, ce qui est essentiel pour le traitement du big data.
3. **Pig :**

* Pig est plutôt une plateforme d'analyse de données pour Hadoop. Piglatin un langage de script conçu pour faciliter le traitement de données sur Hadoop.
* Il utilise des scripts Pig Latin pour décrire les opérations de traitement de données, offrant une abstraction plus élevée que MapReduce et réduisant le besoin de programmation complexe.

1. **Hive :** Hive est un entrepôt de données basé sur Hadoop qui permet aux utilisateurs d'écrire des requêtes similaires à SQL pour interroger et analyser les données. Il vise à simplifier le traitement des données pour les scientifiques qui n'ont pas nécessairement des compétences en programmation.
2. **Scoop :** Scoop est un outil utilisé pour déplacer les données entre Hadoop et des bases de données relationnelles. Il facilite l'importation de données depuis des sources externes dans Hadoop ou l'exportation de données générées par Hadoop vers des bases de données relationnelles.
3. **HBase :** HBase est une base de données NoSQL distribuée, conçue pour gérer de grandes quantités de données avec un accès en temps réel. Elle est souvent utilisée pour des applications nécessitant une lecture/écriture rapide de données massives.
4. **Flume :** Flume est un outil de collecte, d'agrégation et de déplacement de grands volumes de données de manière fiable et efficace. Il est souvent utilisé

pour l'ingestion de données dans Hadoop à partir de sources diverses.

***Ecosystème Hadoop :***

L'écosystème Hadoop englobe un ensemble d'outils et de frameworks qui s'intègrent à Hadoop pour résoudre divers problèmes liés au big data.

Cloudera et Hortonworks sont des sociétés qui ont créé des distributions Hadoop, intégrant ces outils pour une meilleure compatibilité et gestion.

Hbase :

Description : Hbase est une base de données Hadoop.

Hive :

Description : Hive fournit des outils de data warehousing pour extraire, transformer et charger (ETL) des données, ainsi que pour interroger ces données stockées dans des fichiers Hadoop.

Pig :

Description : Pig est un langage de haut niveau qui génère du code MapReduce pour analyser de grands ensembles de données.

Spark :

Description : Spark est un framework de calcul en cluster.

ZooKeeper :

Description : ZooKeeper est un service de configuration centralisé et un registre de noms pour les grands systèmes distribués.

Ambari :

Description : Ambari gère et surveille les clusters Hadoop via une interface utilisateur web intuitive. Cluster ensemble de machine interconnecte pour gerer la qte massive de donne

Avro :

Description : Avro est un système de sérialisation de données.

UIMA :

Description : UIMA est une architecture pour le développement, la découverte, la composition et le déploiement de l'analyse de données non structurées.

Yarn :

Description : Yarn est un système d'exploitation à grande échelle pour les applications Big Data.

MapReduce :

Description : MapReduce est un framework logiciel pour écrire facilement des applications qui traitent des quantités massives de données.

**Explication détaillée sur HDFS** :

*Qu'est-ce que HDFS (Hadoop Distributed File System) ?*

HDFS est un système de fichiers distribué utilisé dans le cadre d'Apache Hadoop, un écosystème open source pour le traitement distribué de données volumineuses. Voici une explication claire et compréhensible des points mentionnés :

*Gestion des grands ensembles de données :*

HDFS est conçu pour gérer d'énormes volumes de données sur du matériel de base.

Il fait partie intégrante d'Apache Hadoop aux côtés d'autres composants comme MapReduce et YARN.

*Objectifs de HDFS :*

**Récupération rapide des pannes de matériel** : Étant donné que les clusters HDFS peuvent avoir des milliers de serveurs, les pannes matérielles sont inévitables. HDFS est conçu pour détecter ces pannes et récupérer rapidement.

**Accès aux flux de données en continu** : HDFS privilégie le traitement par lots plutôt que l'utilisation interactive, offrant des taux élevés de traitement de données pour un accès complet aux ensembles de données.

**Hébergement** **de grands ensembles de données** : Il prend en charge des applications avec des ensembles de données allant de gigaoctets à téraoctets, offrant une grande bande passante de données agrégées et une mise à l'échelle à des centaines de nœuds.

**Portabilité** : HDFS est conçu pour être portable sur différentes plates-formes matérielles et compatible avec divers systèmes d'exploitation.

* HDFS se base sur le traitement par lots c.à.d. il effectue les opérations sont effectuées sur un ensemble complet de données à la fois, plutôt que sur des parties individuelles ou de petites quantités de données de manière interactive et en temps réel. Le traitement se fait de manière séquentielle .

. *Exemple illustratif de HDFS :*

Imaginez un fichier contenant les numéros de téléphone de tout le monde aux États-Unis.

HDFS divise ce répertoire et stocke des parties sur différents serveurs du cluster.

Chaque partie est répliquée sur deux serveurs supplémentaires pour garantir la disponibilité en cas de panne.

La redondance facilite la répartition des tâches sur l'ensemble du cluster, améliorant l'extensibilité.

La localisation des données est optimisée, crucial lors du travail avec d'énormes ensembles de données.

En résumé, HDFS permet de stocker et de gérer de grandes quantités de données sur un cluster distribué, garantissant la résilience, la disponibilité élevée et l'extensibilité pour le traitement de données à grande échelle.

*Type de nodes dans HDFS :*

NameNode :

* Rôle : Il agit comme le gestionnaire principal du système de fichiers.
* Fonction : Stocke les métadonnées du système de fichiers, y compris la structure, les autorisations et l'emplacement physique des blocs de données.
* Responsabilité : Gère l'espace de noms, suit les opérations de lecture/écriture et assure la redondance des métadonnées pour la tolérance aux pannes.

Imaginez-le comme le chef de fichier qui sait où se trouvent tous les morceaux de données dans votre énorme collection de fichiers.

DataNodes :

* Rôle : Ils stockent réellement les blocs de données.
* Fonction : Chaque DataNode est responsable de la gestion des blocs de données qui lui sont attribués.
* Responsabilité : Stocke et récupère les blocs de données sur demande, rapporte périodiquement au NameNode pour indiquer les blocs qu'il détient.

Ce sont comme les gardiens physiques des morceaux de données, stockant réellement les informations que le NameNode gère.

**Nodes dans le cadre de MapReduce**

JobTracker :

* Rôle : Il coordonne et gère l'exécution des jobs MapReduce.
* Fonction : Reçoit les jobs des clients, les planifie en fonction de la disponibilité des ressources et surveille l'exécution des tâches.
* Responsabilité : Optimise l'utilisation des ressources, gère les échecs de tâches, planifie les tâches sur les TaskTrackers.

C'est un chef d'orchestre qui distribue des tâches à accomplir pour traiter de gros ensembles de données (jobs MapReduce).

TaskTracker :

* Rôle : Il exécute effectivement les tâches MapReduce.
* Fonction : Exécute les tâches map et reduce sur les données locales ou télécharge les données nécessaires.
* Responsabilité : Communique avec le JobTracker pour recevoir des tâches, signale l'état d'avancement et signale les éventuelles défaillances.

Ce sont les membres de l'orchestre qui effectuent réellement le travail assigné par le JobTracker, chacun sur une partie des données.

*En résumé, le NameNode gère où se trouvent les données, les DataNodes les stockent, le JobTracker organise le travail à effectuer, et les TaskTrackers réalisent ce travail sur les données. C'est une équipe bien coordonnée pour gérer et traiter de grandes quantités de données.*

YARN (Yet Another Resource Negotiator) traite les demandes de tâches et gère les ressources d'un cluster en environnement informatique distribué. Voici une explication simplifiée de la manière dont YARN gère les demandes de tâches et la gestion des ressources :

Soumission de la Tâche :

Lorsqu'un utilisateur soumet une tâche au cluster YARN, il inclut des informations sur l'application, telles que le type d'application, les besoins en ressources et le code ou script à exécuter.

Demande de Ressources :

Le ResourceManager de YARN reçoit la demande de tâche et négocie des ressources au nom de l'application. Le ResourceManager est l'autorité centrale qui gère et alloue des ressources dans l'ensemble du cluster.

Allocation du Gestionnaire d'Application :

YARN lance un Application Master (AM) pour la tâche soumise. L'AM est responsable de la négociation des ressources spécifiques à son application et de la gestion de son exécution.

Négociation des Ressources :

L'Application Master communique avec le ResourceManager pour demander les ressources nécessaires à ses tâches. Cette négociation inclut des détails tels que le nombre de conteneurs nécessaires, la quantité de mémoire requise et la puissance de traitement nécessaire.

Allocation des Conteneurs :

Le ResourceManager alloue des conteneurs à l'Application Master. Un conteneur est un ensemble de ressources (CPU, mémoire) sur un nœud où une tâche sera exécutée. Le ResourceManager prend en compte la disponibilité des ressources sur différents nœuds et prend des décisions d'allocation.

Exécution de la Tâche :

L'Application Master lance des tâches dans les conteneurs alloués sur les nœuds du cluster. Ces tâches peuvent faire partie d'une application plus vaste et effectuent le calcul réel.

Surveillance et Battements de Cœur :

Tout au long de l'exécution de la tâche, le ResourceManager et les NodeManagers (en cours d'exécution sur chaque nœud) échangent des messages de battement de cœur. Cette communication leur permet de surveiller la santé et la progression des conteneurs et des nœuds.

Libération des Ressources :

Une fois qu'une tâche ou un conteneur a terminé son exécution, le ResourceManager en est informé, et les ressources sont libérées pour être éventuellement réallouées à d'autres tâches.

Fin de la Tâche :

L'Application Master surveille la progression de la tâche entière et signale sa fin au ResourceManager une fois que toutes les tâches sont terminées.

En résumé, YARN agit en tant que gestionnaire de ressources et ordonnanceur dans un cluster Hadoop, traitant efficacement les demandes de tâches en négociant des ressources, en allouant des conteneurs, en gérant l'exécution des tâches et en assurant une utilisation efficace des ressources. Il offre un cadre flexible et scalable pour le traitement de différents types d'applications dans un environnement distribué.