Big Data

ARCHITECTURES DE STOCKAGE, PROCESSUS DÉCISIONNEL, ET ANALYSE DE DONNÉES

32 HEURES - MASTER

PRÉSENTÉ PAR : PR NASRI MOHAMMED

Plan

1. ARCHITECTURES DE STOCKAGE DE DONNEES BIG DATA
   a. Hadoop HDFS
   b. SGBD NoSQL (Cassandra, Hbase, MongoDB, Elastic Search, etc…)·
   c. SGBD Orientés graphes (Neo4J)
2. PROCESSUS DECISIONNEL BIG DATA
   a. Traitement par lot (batch et micro processing) : Etude et Manipulation des Frameworks MapReduce et Spark.
   b. Traitement en temps réel (Stream processing) : Etude des flux de données dès son arrivé. Manipulation du Framework: Kafka Stream
   c. Analyse des données et Visualisation des données.

Introduction à Big Data

a. Big Data décrit des ensembles volumineux de données souvent non structurées.
b. Ces données ne peuvent pas être traités efficacement avec des outils et méthodes traditionnels.
c. Ce concept a émergé en réponse à l'explosion de la quantité de données générées par les activités numériques, les objets connectés et les interactions humaines sur Internet.

Introduction à Big Data

Caractéristiques principales 3V : Volume, Vitesse, Variété :

- **Volume** : Les données sont générées en quantités massives, allant des téraoctets aux pétaoctets. Les outils Facebook ou Google ou IOT génèrent de gros volumes de données.
- **Vitesse** : Les données sont produites et doivent être traitées à grande vitesse, souvent en temps réel.
- **Variété** : Les types de données varient et incluent des données structurées (bases de données relationnelles), semi-structurées (fichiers XML, JSON), et non structurées (vidéos, images, sons, texte libre).

Introduction à Big Data

Caractéristiques principales 3V + Véracité et Valeur:

- **Véracité** : La qualité et la fiabilité des données sont essentielles.
- **Valeur** : Le potentiel d'extraction d'insights précieux à partir de ces données est ce qui donne au Big Data tout son intérêt. Sans valeur exploitable, le Big Data n'a que peu d'utilité.

Introduction à Big Data

Pourquoi c’est intéressent :

- Santé : suivi des patients et analyses en temps réel.
- Secteur public : analyse de données pour améliorer les politiques.
- Médias sociaux : analyse des interactions pour cibler la publicité.

Hadoop : Définition

- **Hadoop** (***H**igh **A**vailability **D**istributed **O**bject **O**riented **P**latform*) est un Framework « open source » de la fondation Apache, qui permet de stocker et traiter les données massives sur un cluster.
- Il repose sur deux composantes majeurs:
  - **HDFS** (*Hadoop Distributed File System*) : un système de fichier distribués sur un ensemble de nœuds (ou serveur) mais manipulés et vus de l'utilisateur comme un seul fichier.
  - **MapReduce** : paradigme de calcul distribué.

Généralités

HDFS est un système de fichiers distribué conçu pour stocker de très gros volumes de données sur un grand nombre de machines.

– Les utilisateurs manipulent les données comme si elles étaient stockées sur un unique ordinateur grâce à la couche d'abstraction fournie par HDFS.

– Dans un système distribué comme HDFS, chaque nœud d'un cluster correspond à un sous-ensemble du volume global de données du cluster. Pour augmenter ce volume global, il suffira d'ajouter de nouveaux nœuds.

– Réplication des données pour assurer la fiabilité et la tolérance aux pannes.

Fonctionnement

- Les fichiers sont physiquement découpés en blocs d'octets de grande taille pour optimiser les temps de transfert et d'accès ;
  - Par défaut, la taille est fixée à 64 Mo. Il est toutefois possible de monter à 128 Mo, 256 Mo, 512 Mo voire 1 Go.
  - Selon la taille d'un fichier, il lui faudra un certain nombre de blocs. le dernier bloc d'un fichier fait la taille restante.
  - Ces blocs sont ensuite répartis sur plusieurs machines, permettant ainsi de traiter un même fichier en parallèle.
  - La répartition de bloc permet aussi de ne pas être limité par la capacité de stockage d'une seule machine pour au contraire tirer parti de tout l'espace disponible du cluster de machines.

Fonctionnement

– Enfin, pour garantir une tolérance aux pannes, les blocs de chaque fichier sont répliqués, de manière intelligente, sur plusieurs machines.

– Par défaut, chaque bloc est copié sur 3 machines différentes (c'est configurable).

– Chaque fichier se trouve donc en plusieurs exemplaires et à différents endroits.

Architecture des machines

Dans Hadoop, l'architecture de stockage est une architecture Master-Slave. Un cluster HDFS est constitué de machines jouant différents rôles exclusifs entre eux :

1. **NameNode (master node) :**

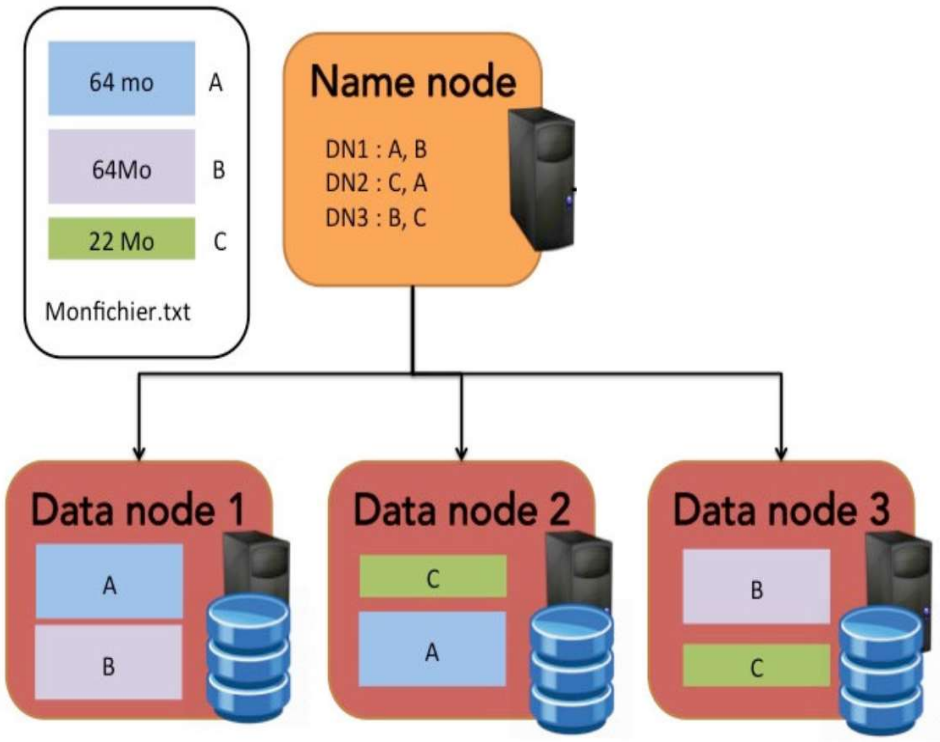
- C'est un service central qui s'occupe de gérer l'état du système de fichier.
- Il maintient **l'arborescence du système de fichiers** et les **métadonnées** de l'ensemble des fichiers et répertoires d'un systèmes Hadoop.

2. **DataNodes (slave nodes) :**

- Nœuds esclaves du Cluster (*il y a un DataNode pour chaque machine pour stocker les données*).
- Dédiés à la gestion des opérations de stockage locales (*manipulation, suppression et réplication de blocs*) sur instruction du NameNode.
- Ils sont sous les ordres du NameNode. Ils sont sollicités par ce dernier lors des opérations de lecture/écriture.

Architecture des machines

- Les ***datanodes*** contiennent des blocs. Les mêmes blocs sont dupliqués *sur différents* datanodes, en général 3 fois. Cela assure :
  - fiabilité des données en cas de panne d'un datanode,
  - accès parallèle par différents processus aux mêmes données.
- Le ***namenode*** sait à la fois :
  - sur quels blocs sont contenus les fichiers,
  - sur quels datanodes se trouvent les blocs voulus.



Ecriture sur HDFS

Si on souhaite écrire un fichier dans HDFS, on utilise un client Hadoop (*une ou plusieurs machines jouant le rôle d'un point d'accès au cluster pour s'y connecter et travailler*).

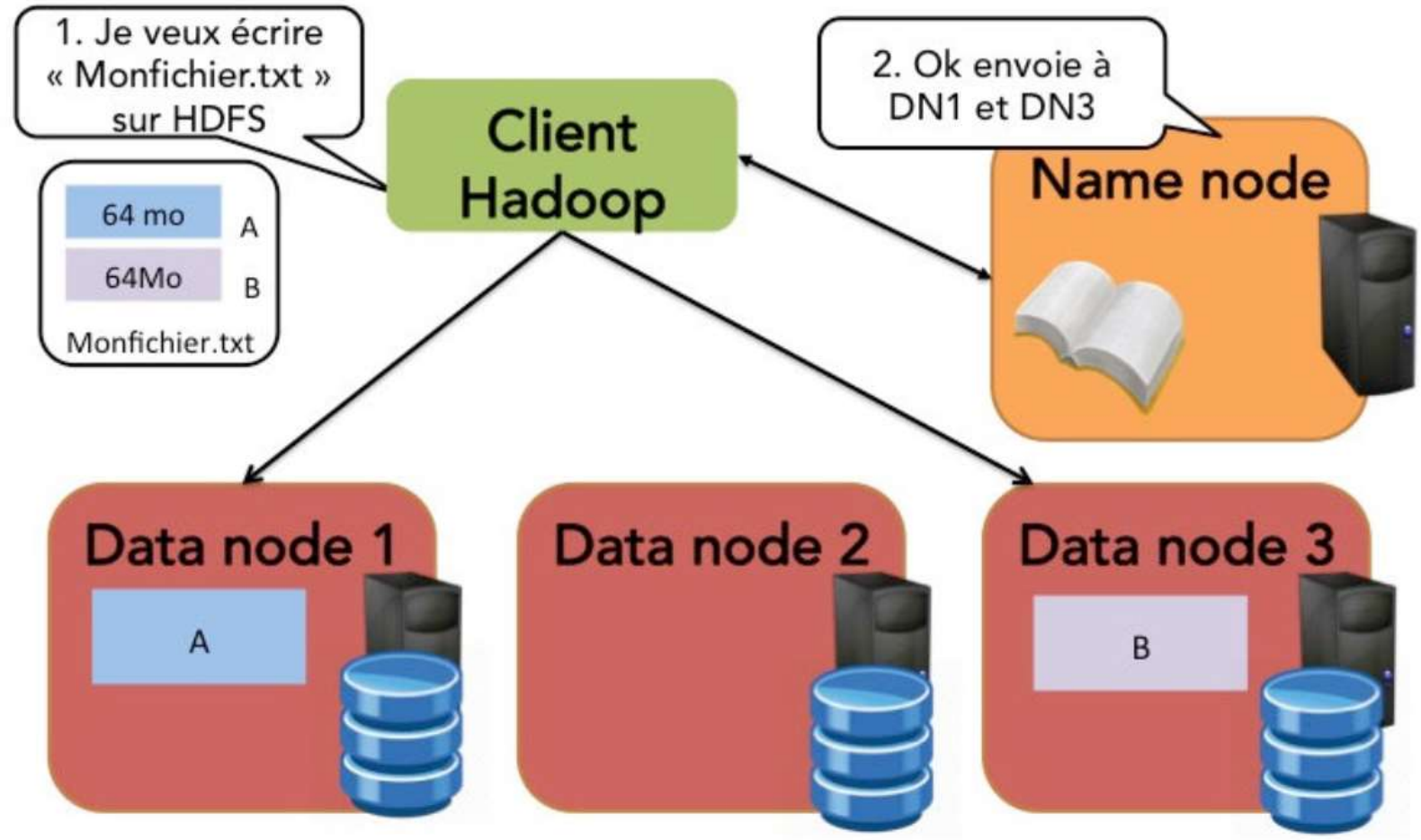
– Le fichier sera divisé en bloc de 64 Mo (*ou autre selon la configuration*). Il va ensuite annoncer au NameNode : « je souhaite stocker ce fichier au sein de HDFS ».

– Le NameNode va indiquer par exemple qu'il faut stocker le bloc 1 sur le DataNode 1 et le bloc 2 sur le DataNode 3.

Le principe est assez simple :

1. Le client indique au NameNode qu'il souhaite écrire un bloc.
2. Le NameNode indique le DataNode à contacter.
3. Le client envoie le bloc au DataNode.
4. Les DataNode répliquent les blocs entre eux.
5. Le cycle se répète pour le bloc suivant.

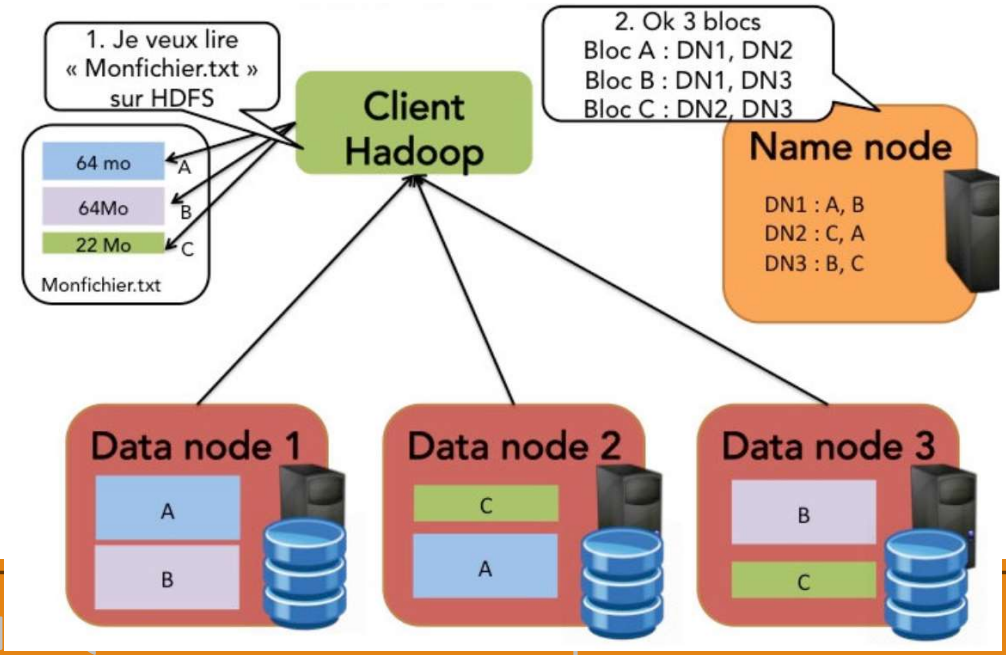
Ecriture sur HDFS



Lecture depuis HDFS

Si on souhaite lire un fichier dans HDFS, le processus est le suivant :

1. Le client indique au NameNode qu'il souhaite lire un fichier.
2. Le NameNode indique sa taille ainsi que les différents DataNodes contenant les blocs.
3. Le client récupère chacun des blocs sur l'un des DataNodes.
4. Si le DataNode est indisponible, le client en contacte un autre.



Atelier pratique

Durant le lab. 1, vous allez :

1. Installer une architecture HDFS basé sur un NameNode et 2 DataNodes.
2. Puis pratiquer les base d'écriture et de lecture dans et depuis HDFS.

Le lien du lab. 1 :

https://github.com/nasri-lab/bigdata/blob/main/labs/lab1-en.md