Gestion de Projet Big Data & Développement d'applications Big Data

EDAH Kodjo
Consultant Systèmes d’Information, Big-Data

Objectifs

- Comprendre la notion et les spécificités du Big Data
- Connaître les technologies de l’écosystème Hadoop
- Connaître le langage python et utiliser les librairies de machine learning
- Savoir utiliser les outils de visualisation des données (Dataviz)

Partie 1 : Définition et les enjeux du big data

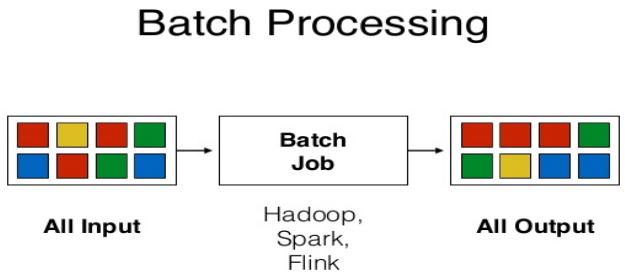
Partie 2 : Ecosystème hadoop

- Connaître les technologies de l'écosystème big data
- Connaître les outils big data

Terminologie big data

Batch : lot, traitement par lot, enchainement automatique de commande

Batch processing : Le batch processing permet de traiter de larges volumes de données



Terminologie big data

Cloud computing : mode de traitement des données d'un client, dont l'exploitation s'effectue par l'internet, sous la forme de services fournis par un prestataire

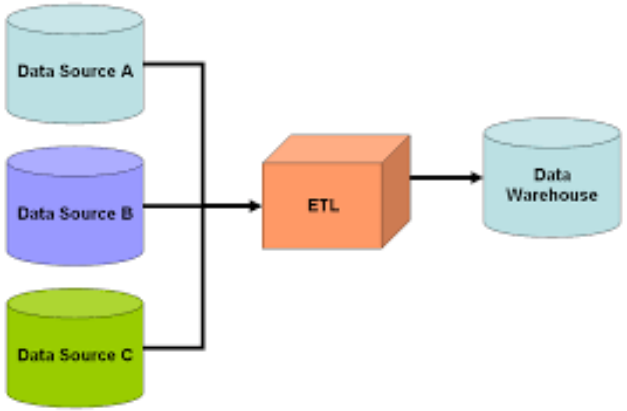
Un Data Lake : est un répertoire où sont stockées de nombreuses données d’entreprises au format brut

Terminologie big data

Les DataWarehouses : généralement utilisées pour le stockage de données conventionnelles, structurées et déjà formatées.

ETL : Extraire, Transformer et Load (charger)

In-memory Computing : Le computing in-memory est une technique permettant de transférer des ensembles de données complets vers la mémoire collective d'un cluster et d'éviter d'écrire des calculs intermédiaires sur le disque.

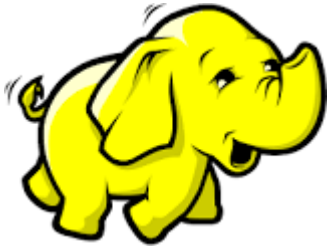


Framework Hadoop

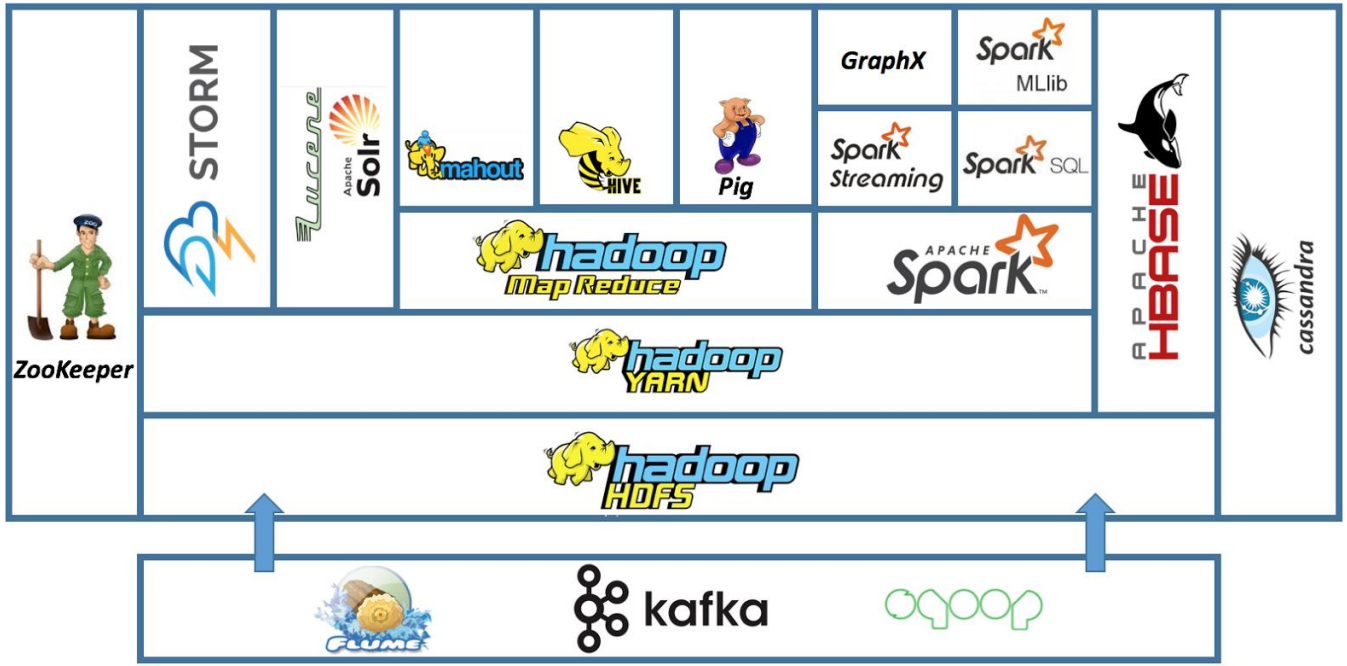
Hadoop : framework libre et open source écrit en Java destiné à faciliter la création d'applications distribuées (au niveau du stockage des données et de leur traitement) et échelonnables (scalables) permettant aux applications de travailler avec des milliers de nœuds et des pétaoctets de données. (Source wikipedia)

Caractéristiques de Hadoop

- ❑ **Traitement et stockage haute performance**
- ❑ **Résilience**
- ❑ **Haute disponibilité**
- ❑ **Scalabilité horizontale**
- ❑ **Optimisé pour la lecture en mode batch (**WORM)
- ❑ **Flexibilité**
- ❑ **Écosystème vaste et grandissant**



Framework Hadoop



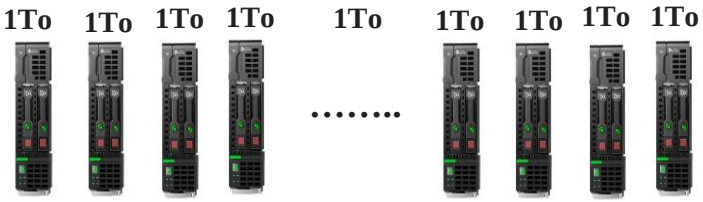
source : http://blog.newtechways.com/2017/10/apache-hadoop-ecosystem.html

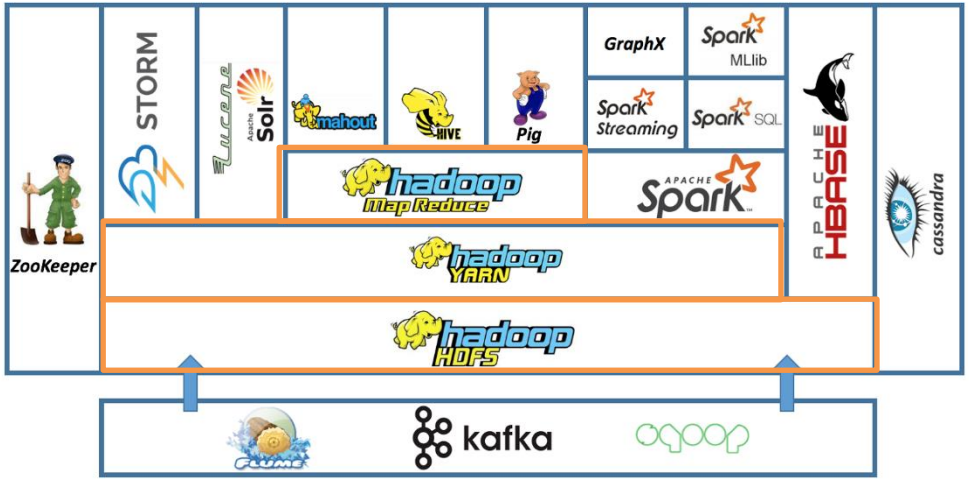
Module de base Hadoop : HDFS

Hadoop Distributed File System

Système de fichier

Stockage distribué



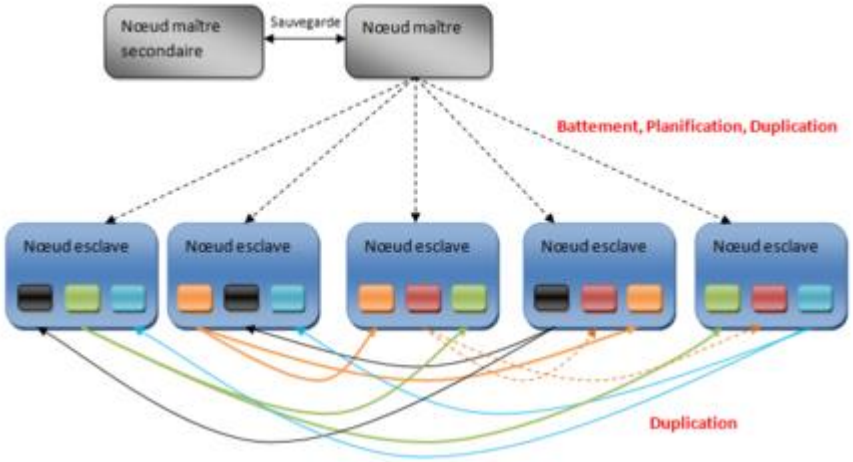


HDFS : Hadoop Distributed File System (Composants)

Master

Namenode

- Nœud de noms
- Gère l'espace de noms (namespace) l'arborescence du système de fichiers et les métadonnées des fichiers et des répertoires.
- Centralise la localisation des blocs de données répartis dans le cluster

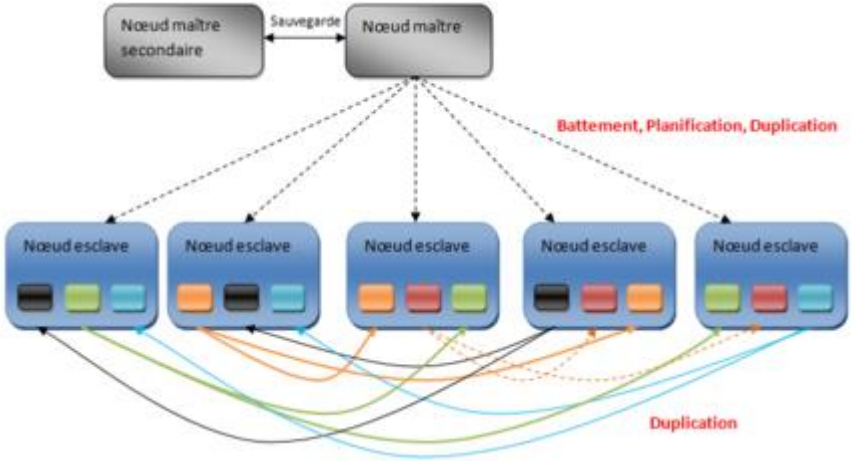


HDFS : Hadoop Distributed File System (Composants)

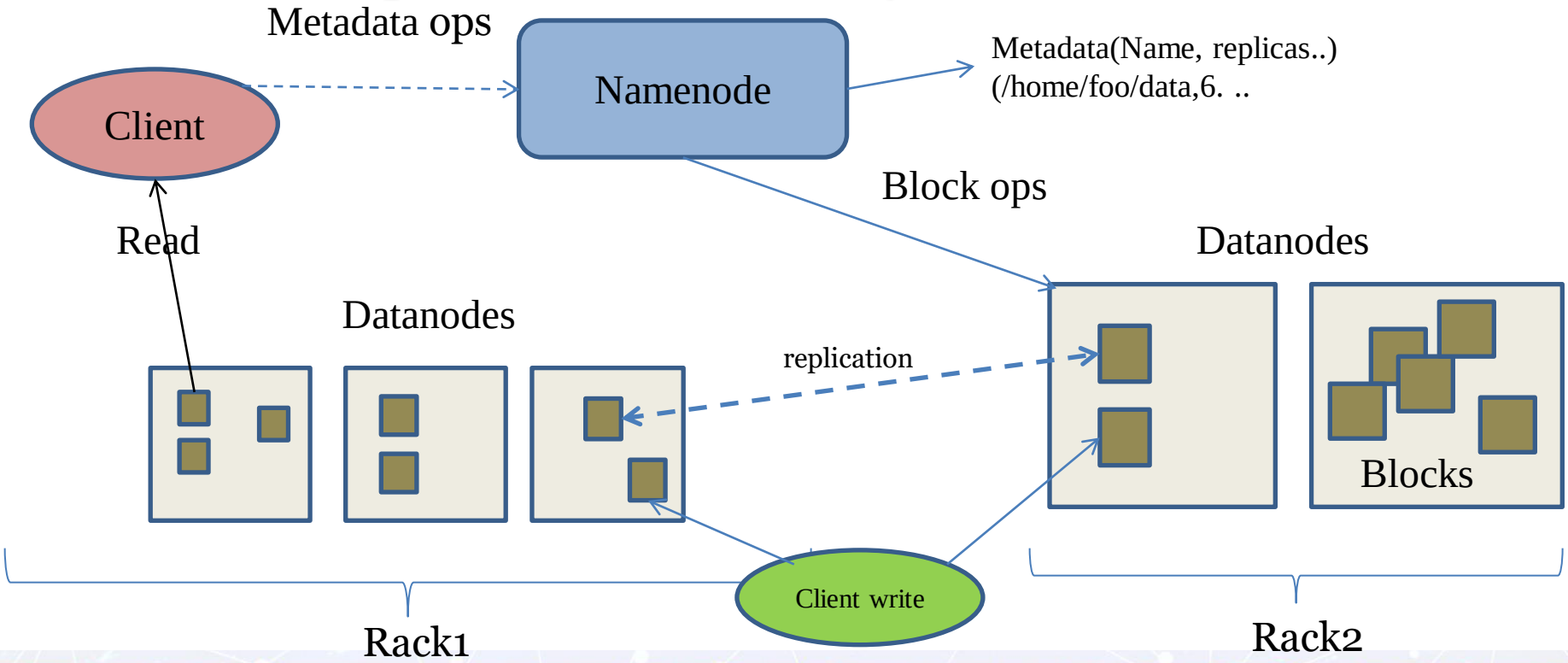
Datanode

Slaves

- Nœud de donnée
- Stocke et restitue les blocs de données.



HDFS : Hadoop Distributed File System



HDFS : les fonctions du NameNode (1/2)

- C'est le démon maître qui maintient et gère les DataNodes (nœuds esclaves)
- Il enregistre les métadonnées de tous les fichiers stockés dans le cluster
  - l'emplacement des blocs stockés
  - la taille des fichiers
  - les autorisations
  - la hiérarchie
- Deux fichiers sont associés aux métadonnées:
  - FsImage: il contient l'état complet de l'espace de noms du système de fichiers depuis le début du NameNode.
  - EditLogs: il contient toutes les modifications récentes apportées au système de fichiers par rapport à la FsImage la plus récente.

HDFS : les fonctions du NameNode (2/2)

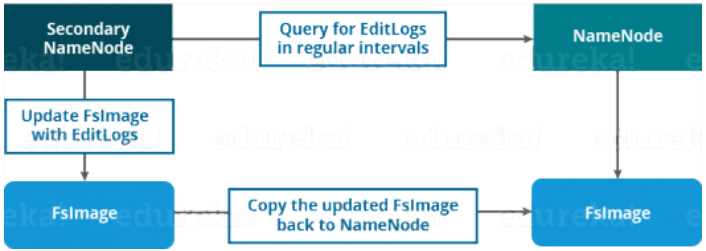
- Enregistre chaque modification apportée aux métadonnées du système de fichiers. Par exemple, si un fichier est supprimé dans HDFS, le NameNode l'enregistrera immédiatement dans EditLog.
- Reçoit régulièrement un Heartbeat et un rapport des blocs de tous les DataNodes du cluster pour s'assurer que les DataNodes sont actifs.
- Il garde un enregistrement de tous les blocs dans HDFS et dans quels nœuds ces blocs sont situés.
- Prends en charge le facteur de réplication de tous les blocs
- En cas d'échec du DataNode, le NameNode choisit de nouveaux DataNodes

HDFS : les fonctions du DataNode

- Démons ou processus esclaves qui s'exécutent sur chaque machine esclave.
- Les données réelles sont stockées sur les DataNodes (par block).
- Les DataNodes exécutent les requêtes de lecture et d’écriture de bas niveau à partir des clients du système de fichiers.
- Ils envoient périodiquement des Heartbeat et blockreport au NameNode

HDFS : les fonctions du Secondary NameNode

- Lit constamment tous les systèmes de fichiers et métadonnées de la RAM du NameNode et les écrit sur le disque dur ou le système de fichiers.
- Effectue la combinaison des EditLogs avec FsImage du NameNode.
- Télécharge les EditLogs à partir du NameNode à intervalles réguliers et l'applique à FsImage. Le nouveau FsImage est recopié dans le NameNode, qui est utilisé chaque fois que le NameNode est démarré la prochaine fois



HDFS : les blocks et la replication

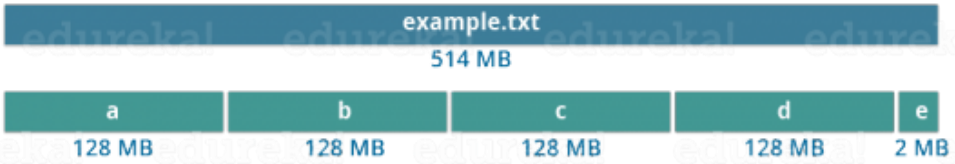
- Le plus petit emplacement continu sur le disque dur où les données sont stockées

- En général chaque système de fichiers stocke un fichier en une collection de **block**

https://www.edureka.co/blog/apache-hadoop-hdfs-architecture/

HDFS : les blocks et la replication

- Cas **hdfs** : taille fixe de 128 Mo par défaut
  - **Choix important de la taille des blocks**
  - **Trop petit => trop de données dans le namenode**
  - **Trop gros => gaspillage d'utilisation du disque si la plupart des fichiers sont plus petits**



https://www.edureka.co/blog/apache-hadoop-hdfs-architecture/

HDFS : les blocks, la réplication et rack awareness

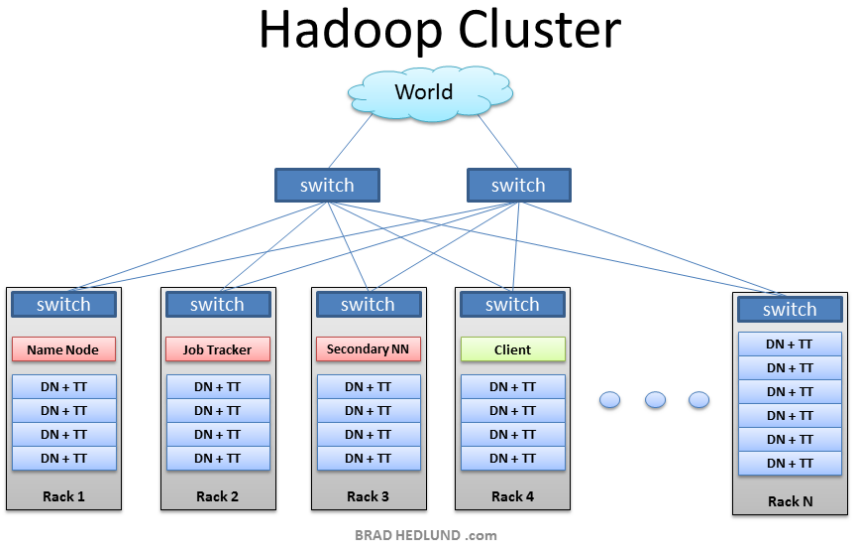
- Les blocks sont stockés dans les DataNodes de manière distribuée
- Les blocks sont répliqués pour fournir une tolérance aux pannes
- Le facteur de réplica par défaut est 3
  - Chaque block doit être stocké 3 fois
  - **Si plus namenode demande aux datanodes une suppression**
  - **Si moins, le namenode demande aux datanodes un ajout**

HDFS : les blocks, la réplication et rack awareness

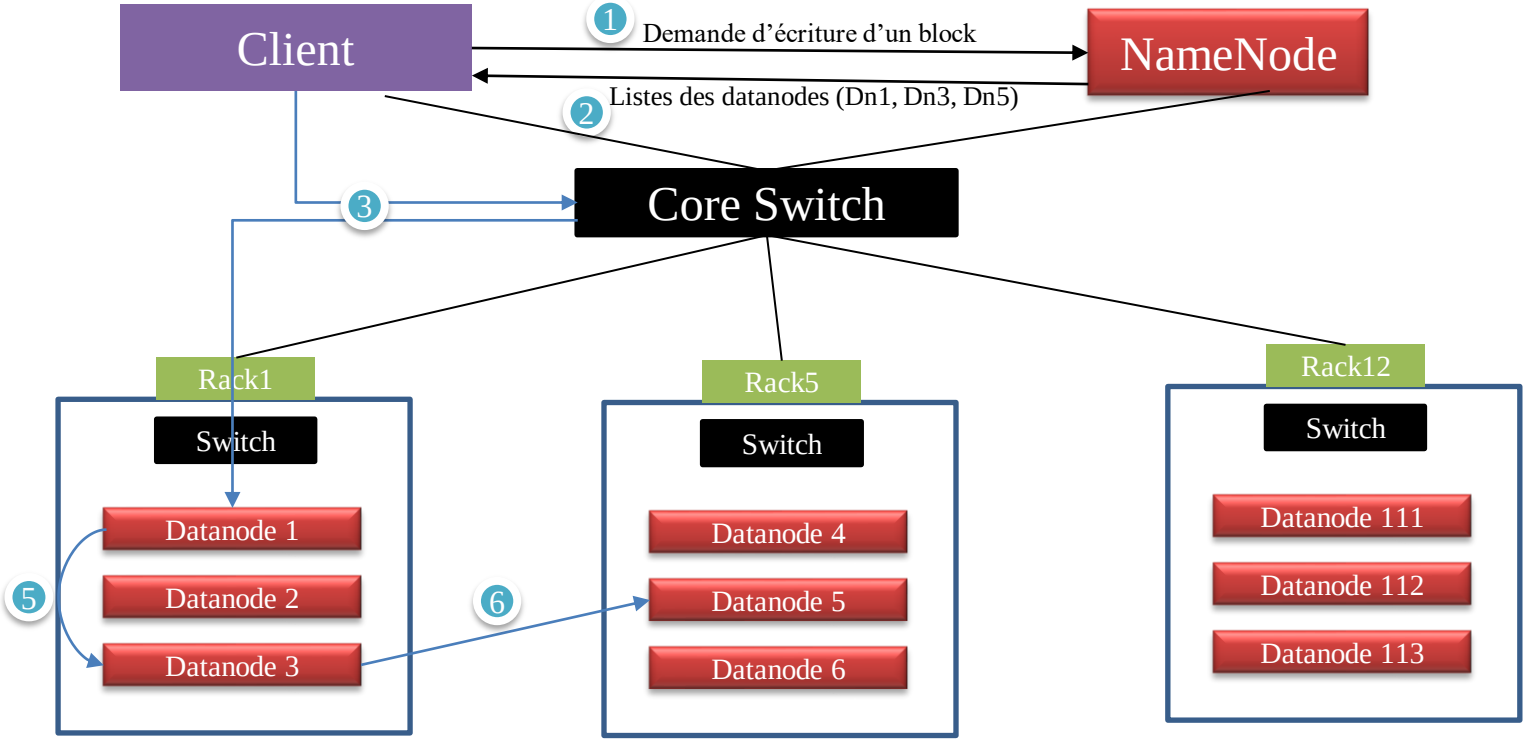
- Rack : Ensemble de machine de machine connecté à l'aide du même switch

Rack Awareness Algorithm

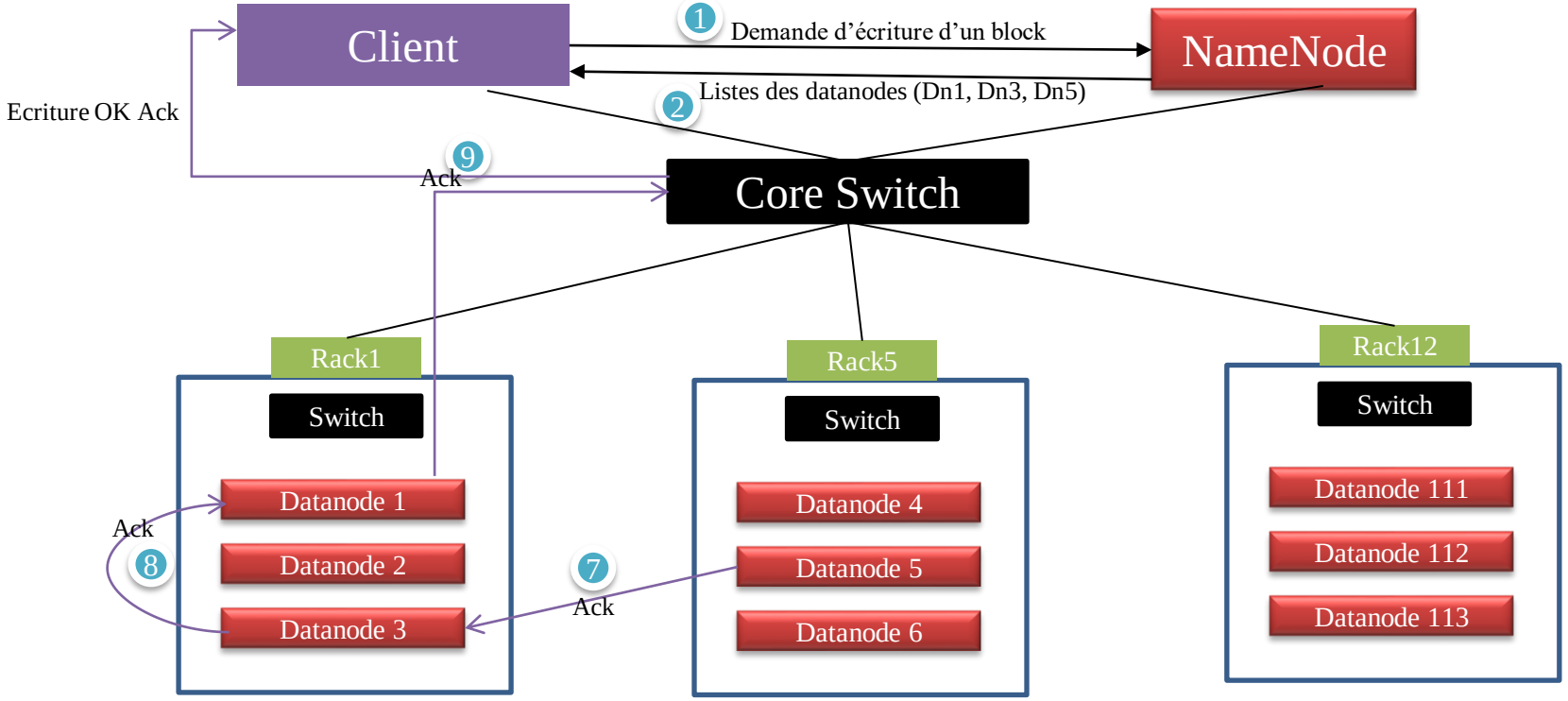
- Le premier réplica du bloc sera stocké sur un rack local.
- Le deuxième sera stocké sur un autre datanode dans le même rack.
- Le troisième réplica stocké sur un rack différent.



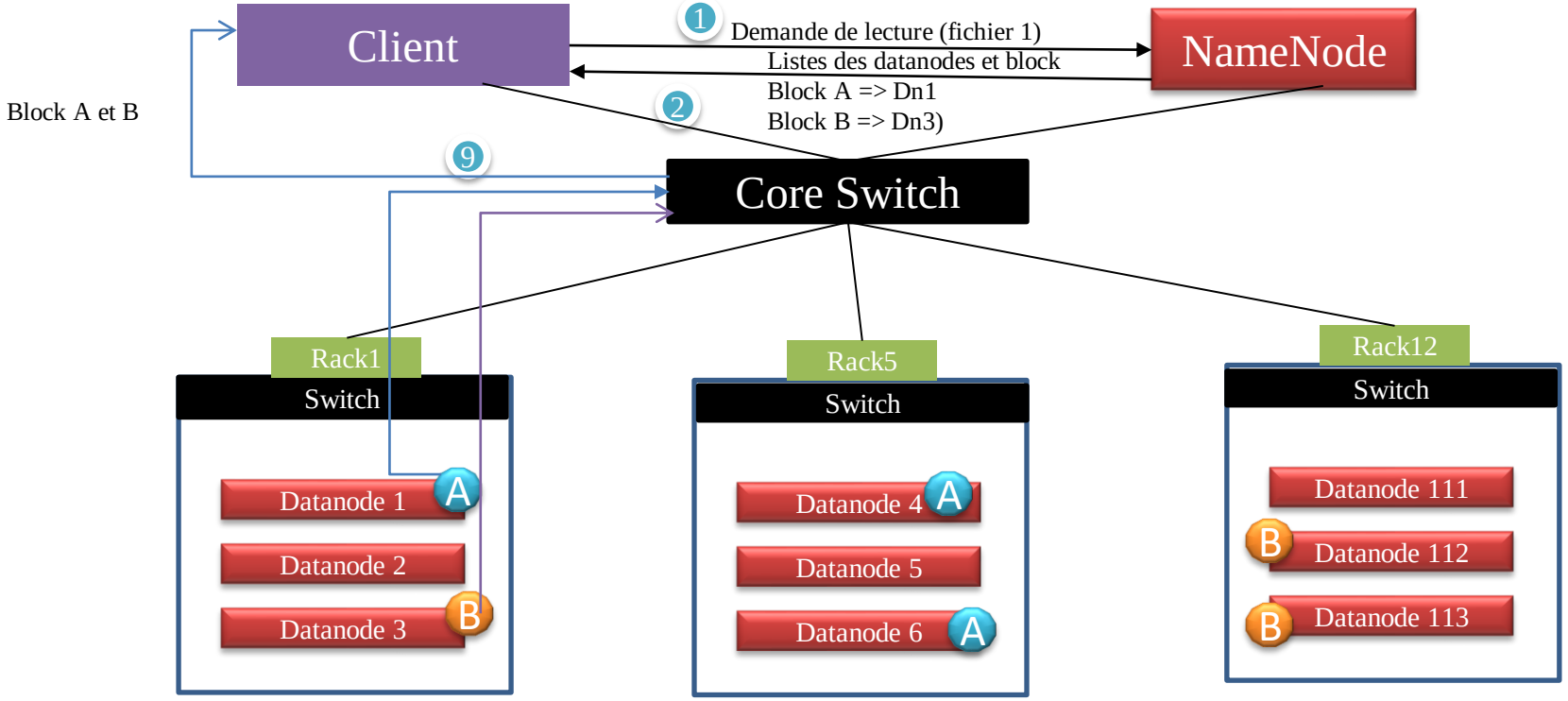
HDFS : Ecriture facteur replication 3 (1/2)



HDFS : Ecriture facteur replication 3 (2/2)



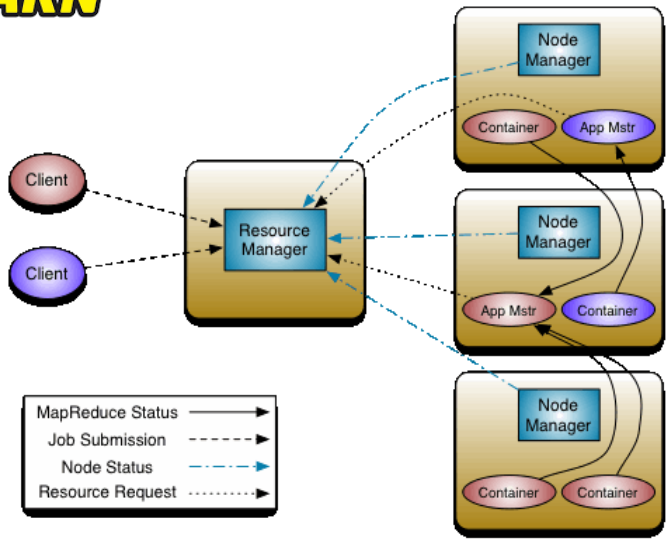
HDFS : Lecture d'un fichier divisé de deux blocks (A ,B)



Module de base Hadoop : Yarn

Yet Another Resource Negotiator

- Le gestionnaire de ressources et des tâches dans un cluster Hadoop
- **Resource Manager** = Scheduler + ApplicationManager
- **Node manager** : responsable des containers , surveille leur utilisation des ressources (processeur, mémoire, disque, réseau) et en rend compte au ResourceManager (Scheduler)



Source : https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/yarn_architecture.gif

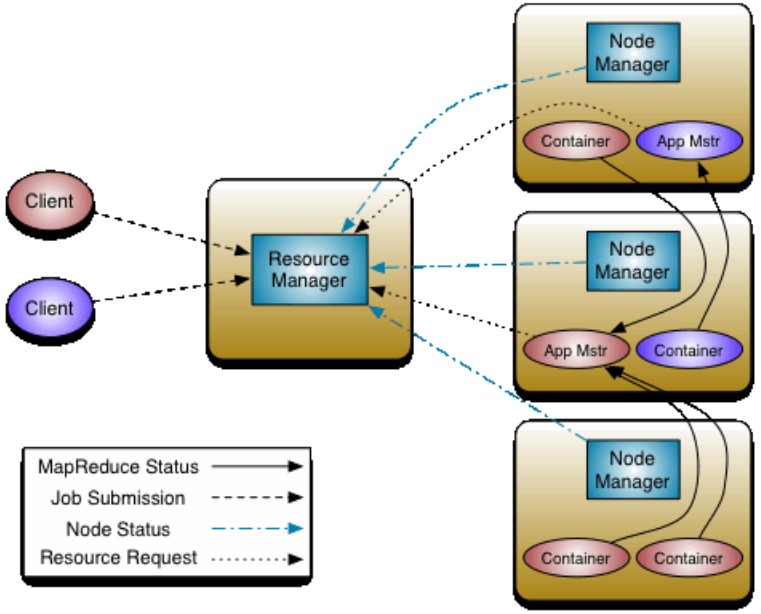
Module de base Hadoop : YARN

❑ Scheduler

- ❑ Allocation de ressources
- ❑ Planification basée sur les ressources demandées par l'application en se basant sur le Container (Mémoire, CPU, Disk, Réseau ….)

❑ ApplicationsManager

- ❑ Accepter la soumission des jobs
- ❑ Négocier le premier Container pour exécuter l'applicationMaster et fournir un service pour redémarrer l'applicationMaster en cas d'échec
- ❑ L'applicationMaster doit négocier les ressources (Container) avec le Scheduler, suivre l'état d'avancement des jobs et les status



Source : https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/yarn_architecture.gif

https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html

Module de base Hadoop : HDFS

Mode de fonctionnement

- Standalone Mode
- Pseudo-distributed Mode ou Single Node Cluster
- Fully-Distributed Mode ou (Multi-Node Cluster)

Module de base Hadoop : HDFS

Mode de fonctionnement

❑ Standalone Mode

- ➢ Installation sur un seul système
- ➢ Tous les composants dans la même JVM
- ➢ Mode de configuration par défaut
- ➢ Aucune configuration requise
- ➢ Utilisation pour apprentissage ou debbugage

Module de base Hadoop : HDFS

Mode de fonctionnement

❑ Pseudo-distributed Mode

- ➢ Simulation du cluster
- ➢ Composants indépendants (JMV indépendant)
- ➢ Exécution dans une seule machine
- ➢ Modification des fichiers de configuration (mapred-site.xml, core-site.xml, hdfs-site.xml, yarn-site.xml .)
- ➢ Utilisation pour apprentissage ou debbugage

Module de base Hadoop : HDFS

Mode de fonctionnement

- **Fully-distributed Mode**

  - **Les nœuds master sur des machines séparées (namenode, resource manager)**
  - **Les nœuds esclave sur d'autres machines (datanode, nodemanager)**
  - **Modification des fichiers de configuration (mapred-site.xml, core-site.xml, hdfs-site.xml, yarn-site.xml .)**
  - **Mode de production**

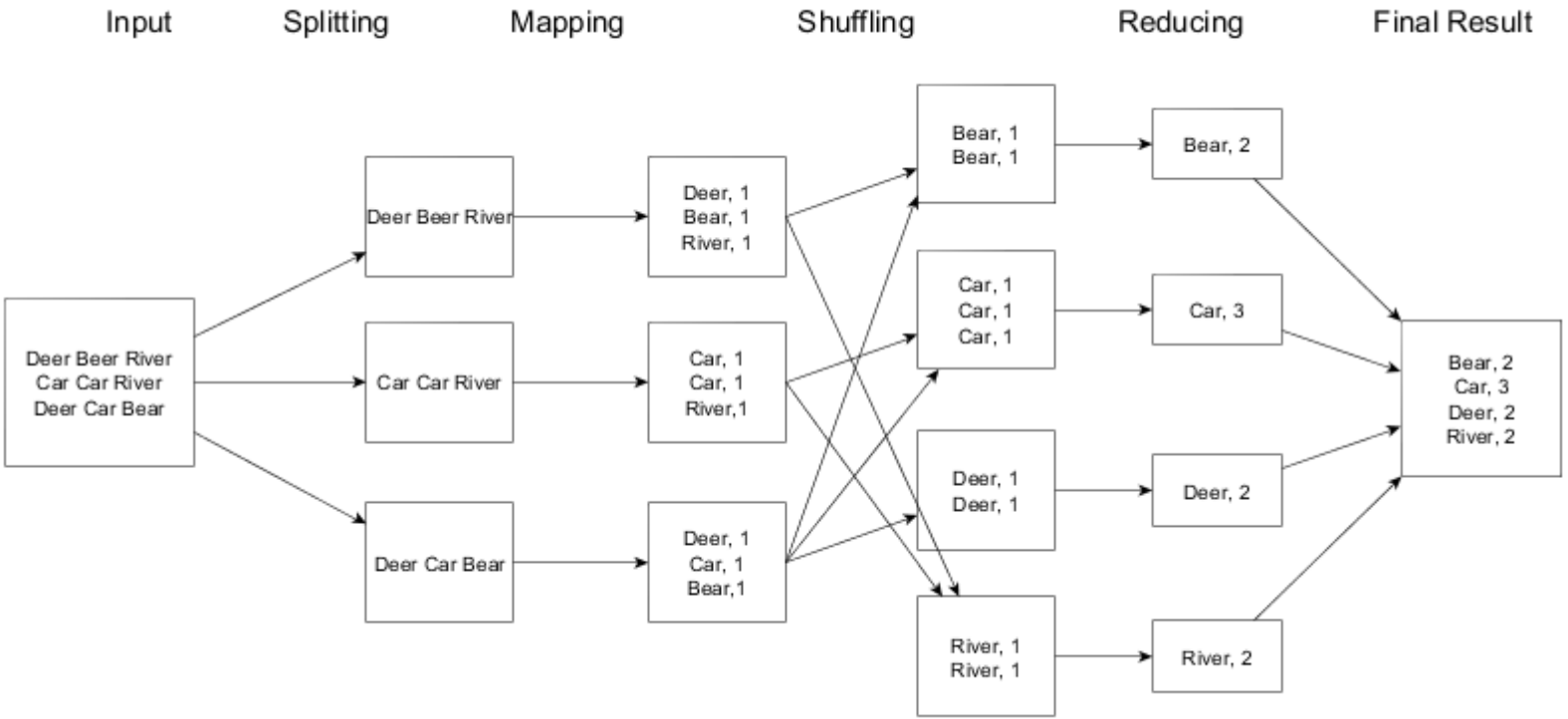
Module de base Hadoop : Map Reduce

- Modèle de programmation pour permettre d'appliquer un calcul (Map) et d'agréger les résultats (Reduce).

- Map : A partir d'un ensemble de données, la fonction map va générer un autre ensemble de données sous forme de tuples (paire de clé/valeur).

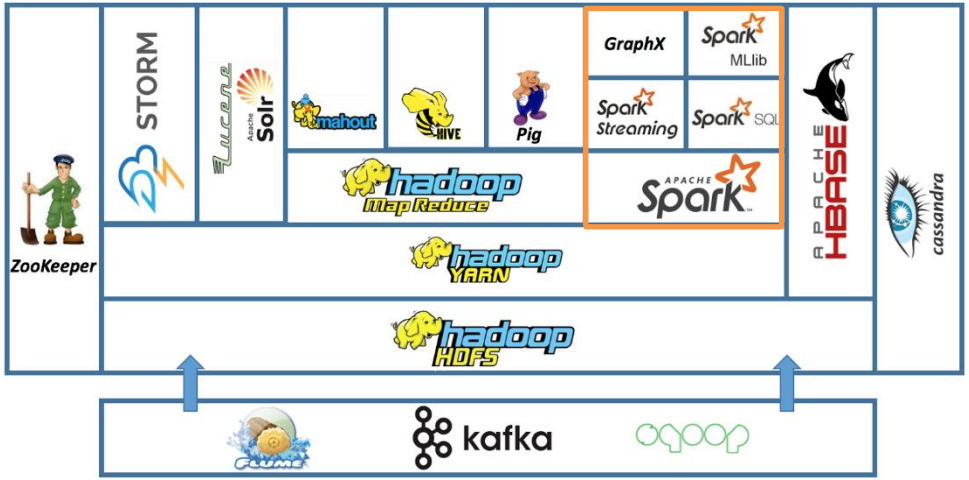
- Reduce : A partir des résultats issus de la fonction map, les tuples sont combinés pour être réduits en un plus petit ensemble de tuples.

Le paradigme Map-Reduce : Exemple words count



Module Hadoop : Spark

- Un moteur d'analyse unifié pour le traitement de données à grande échelle (distribué)
- RDD : Resilient Distributed Datasets
  - collection de données calculée à partir d'une source et conservée en mémoire vive (si possible)
  - répartis sur le cluster de données permettant la récupération complète de données en cas de perte



Spark : caractéristiques

❑ Rapide

- ✓ 100 fois plus rapide
- ✓ 2014 Dayton GraySort Contest : Objectif trié 100To de données
  - ➢ Spark 206 machines en 23 minutes
  - ➢ Hadoop Map Reduce : 2100 machines en 72 min (tenant du titre)

Source :. ©CAMERON SPENCER/GETTY IMAGES

❑ Facile à utiliser

- ❑ Plus de 80 opérateurs
- ❑ **Scala**, R, Python, Java

```
df = spark.read.json("logs.json")
df.where("age > 21")
  .select("name.first").show()
```

Spark's Python DataFrame API
Read JSON files with automatic schema inference

Spark : caractéristiques

APACHE SPARK

SPARK SQL	SPARK Streaming	SPARK Graph X	SPARK MLlib
Exécuter des requêtes en langages SQL pour charger et transformer des données.	Spark Streaming offre à son utilisateur un traitement des données en flux	permet de traiter les informations issues de graphes	une bibliothèque d'apprentissage automatique

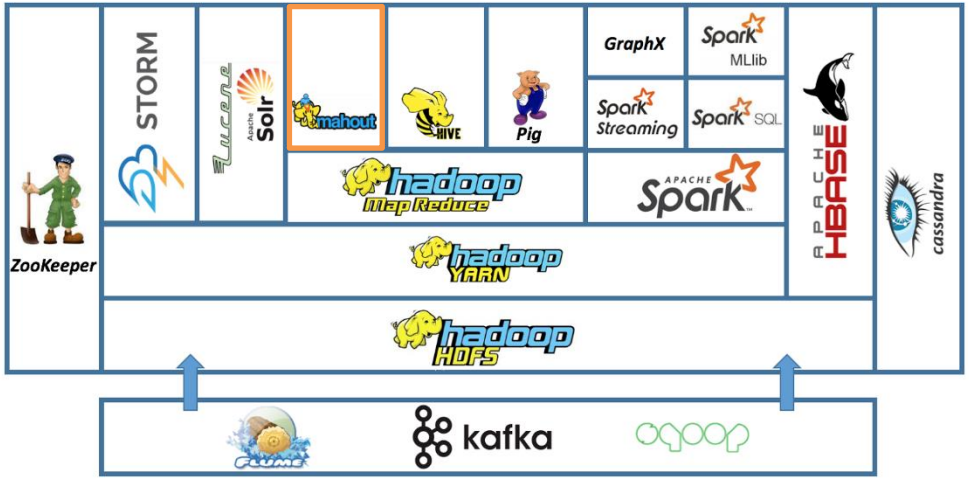
Spark : caractéristiques

- Hadoop, Apache Mesos, Kubernetes, de manière autonome ou dans le cloud.
- Il peut accéder à diverses sources de données.

Module Hadoop : Mahout

- créer des implémentations d'algorithmes d'apprentissage automatique distribués

- Implémentation
  - ✓ Recommandation
  - ✓ Classification
  - ✓ Clusterisation

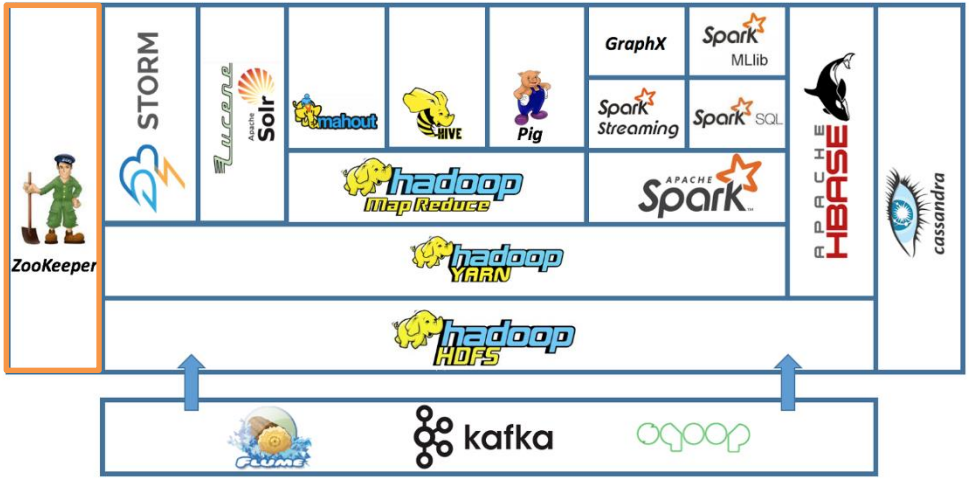


https://www.tutorialspoint.com/mahout/index.htm

Ecosystème Hadoop : Zookeeper

- ZooKeeper est un service centralisé de gestion des informations de configuration, de nommage (naming), de synchronisation distribuée et de service de groupe

Apache Zookeeper



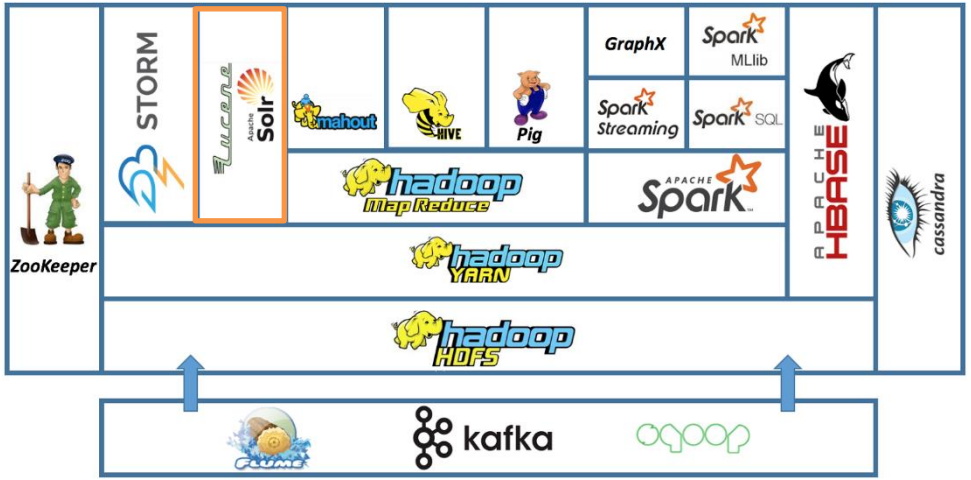
https://cwiki.apache.org/confluence/display/ZOOKEEPER/ProjectDescription

https://blog.engineering.publicissapient.fr/2015/02/24/introduction-et-demystification-de-zookeeper/

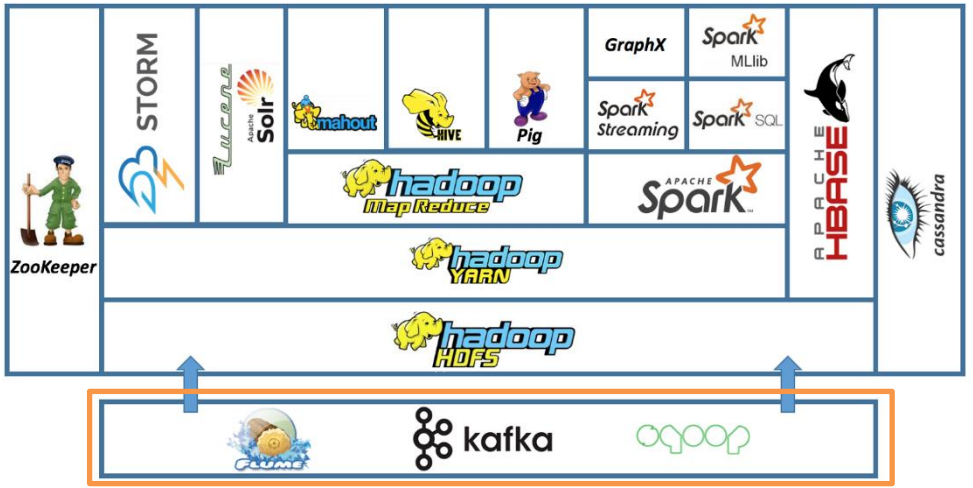
Ecosystème Hadoop: recherche et l'indexation

Solr

Lucene

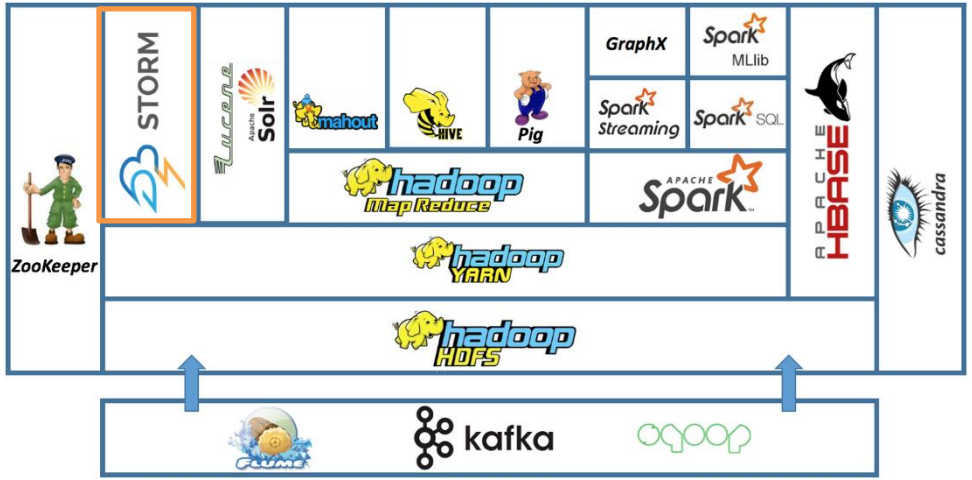


Ecosytème Hadoop : Service ingestion de données



Ecosystème Hadoop : Storm

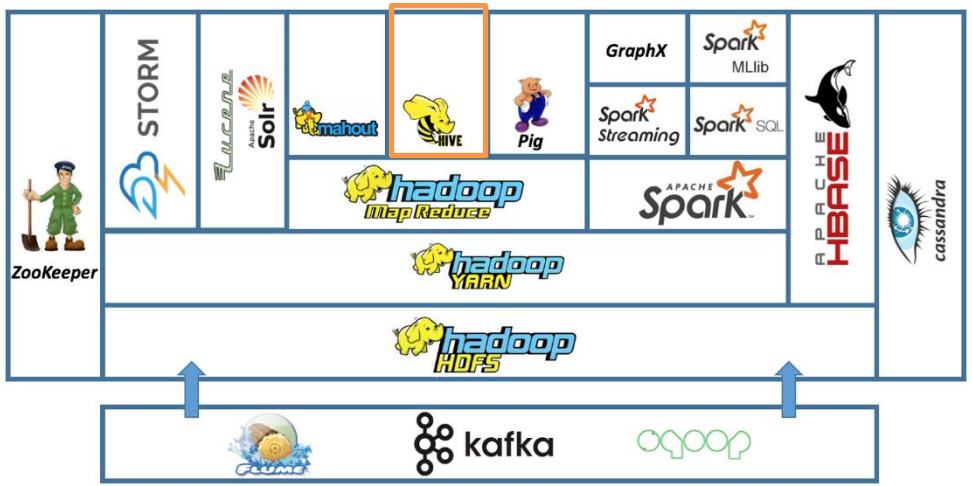
- Framework de calcul de traitement de flux distribué



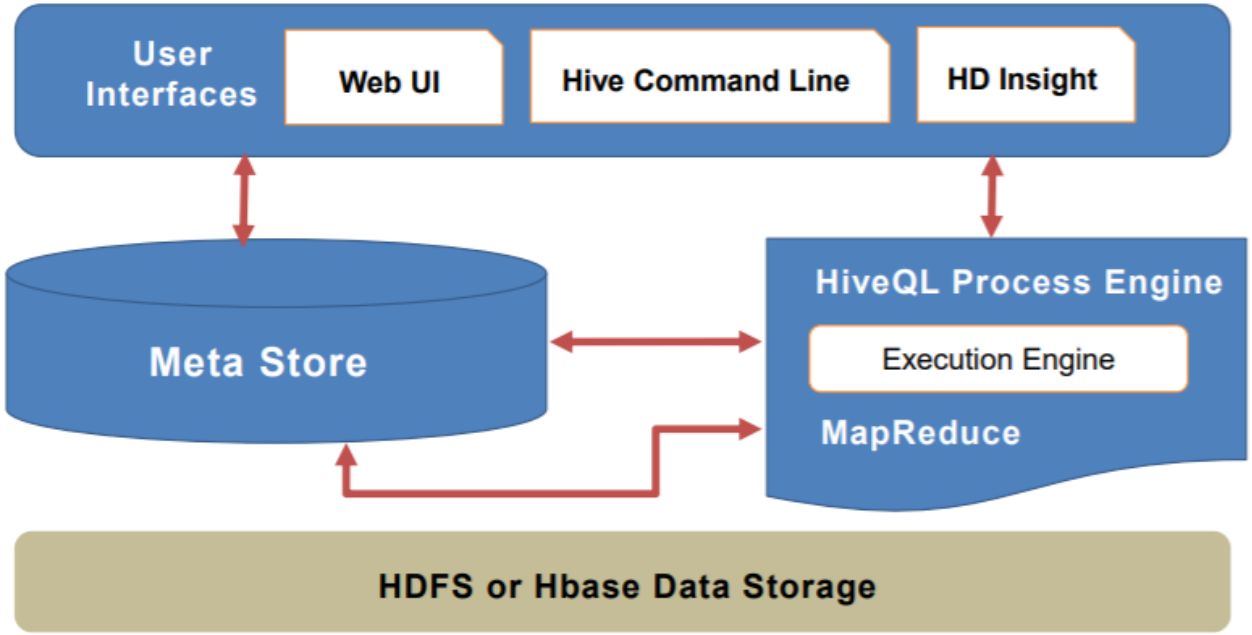
Hive

- Apache Hive est une infrastructure d'entrepôt de données intégrée sur Hadoop permettant l'analyse, le requêtage via un langage proche syntaxiquement de SQL ainsi que la synthèse de données (wiki)

- ✓ Écrit en Java avec une infrastructure Data Warehouse
- ✓ Traite les données structurées dans Hadoop
- ✓ Exécute des requêtes proches de la syntaxe SQL
- ✓ Orienté OLAP



Hive : architecture



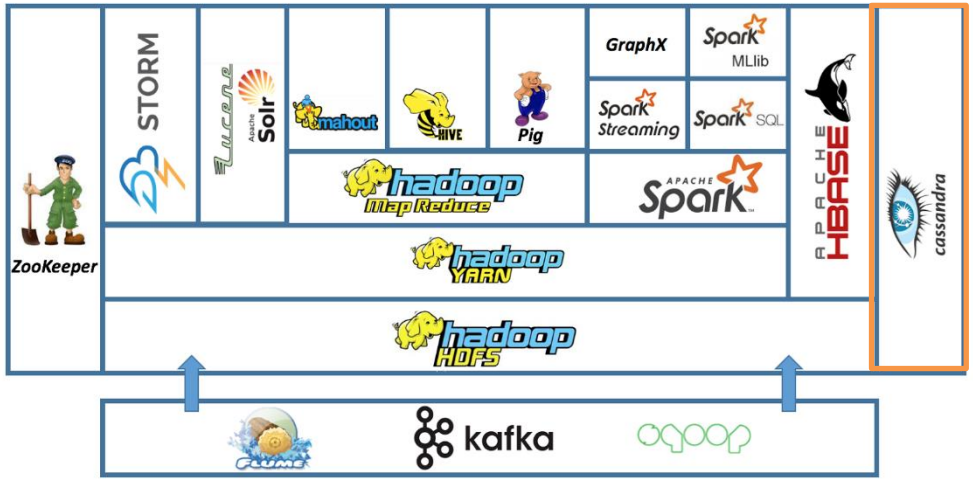
Hive : hive vs base relationnelle

	Bases de données	Hive
Langage	SQL	HiveQL / HQL
Transactions	Oui	Non
Update/Delete	Oui	Non
Latence	Faible	Elevée
Volume de données	Teraoctet	Petaoctet

Cassandra

- Système de gestion de base de données (SGBD) de type NoSQL conçu pour gérer des quantités massives de données sur un grand nombre de serveurs, assurant une haute disponibilité en éliminant les points individuels de défaillance.

- ✓ Réplication asynchrone sans master
- ✓ CQL (pour Cassandra Query Language)
- ✓ Faible latence pour les opérations de tous les clients
- ✓ Architecture distribuée et architecture relationnelle est orientée colonne
- ✓ Structuration en paires clé-valeur de type **eventually consistent**



Ecosystème Hadoop : Ambari

- Système de planification à usage général pour les travaux Hadoop multi-étapes, Système de planification de workflow pour gérer les jobs Hadoop

- Mise à disposition, monitoring et maintenance de cluster

Apache Ambari

Ecosystème Big data : ELK

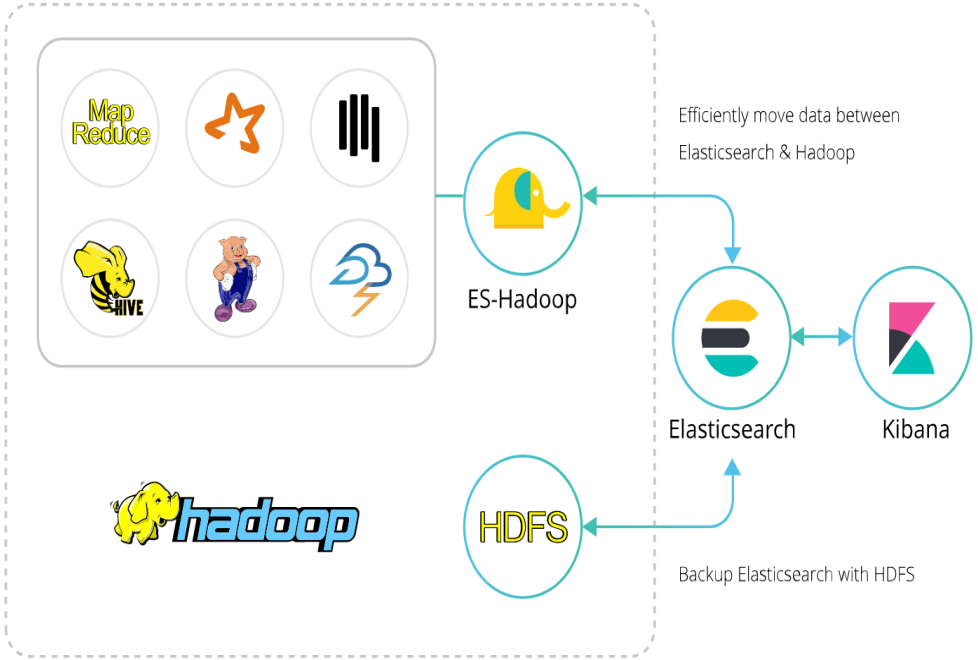
- Elasticsearch est un moteur de recherche et d'analyse RESTful distribué
  - centralise le stockage de vos données
  - assure une recherche ultra-rapide
  - scalables

- Logstash est un pipeline côté serveur destiné au traitement des données
  - Ingérer des données provenant d'une multitude de sources
  - Transformer
  - Envoyer vers votre système de stockage préféré (Elasticsearch …..)

- Kibana est une interface utilisateur qui vous permet de visualiser vos données Elasticsearch et de naviguer dans la Suite Elastic

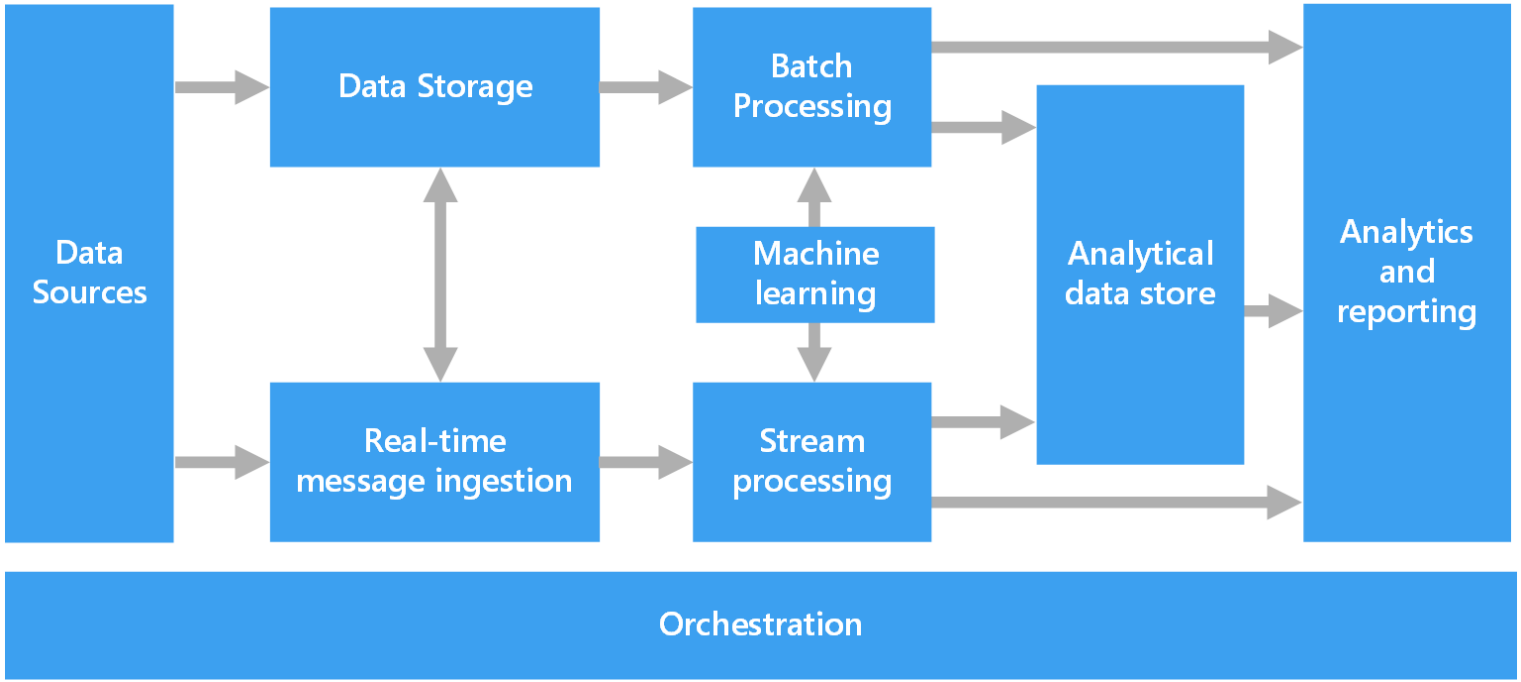
Ecosystème big data : ES-Hadoop

- ES-Hadoop permet d'indexer vos données Hadoop dans la Suite Elastic
- Pallier aux insuffisances d'analyse temps réel d'hadoop



https://www.elastic.co/fr/what-is/elasticsearch-hadoop

Exemple d'architecture Big data

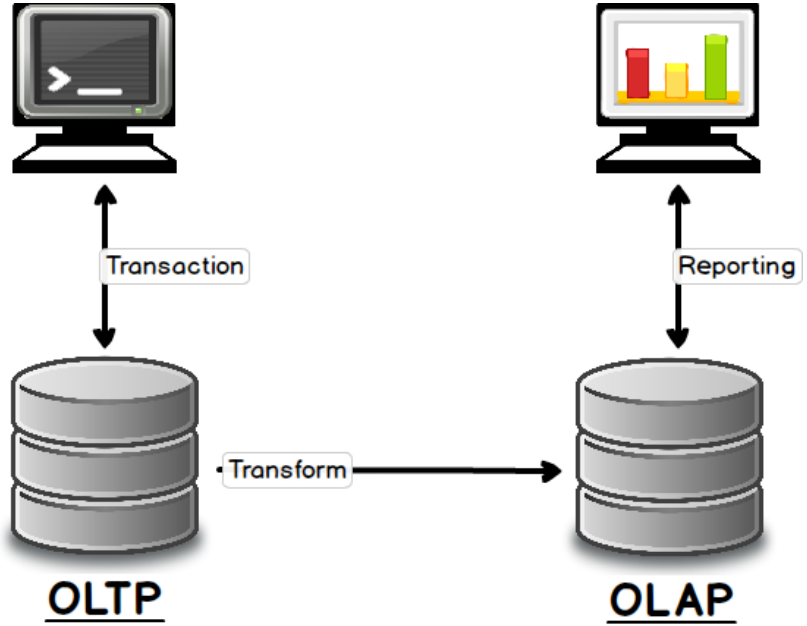


https://docs.microsoft.com/fr-fr/azure/architecture/data-guide/big-data/

OLTP vs OLAP

OnLine Transaction Processing

OnLine Analytical Processing



OLTP vs OLAP

	OLTP	OLAP
Définition	C'est un système transactionnel en ligne qui sert à effectuer des modifications dans une base de données.	C'est un système de récupération de données et d'analyse de données en ligne.
Transaction	OLTP a des transactions courtes.	OLAP a des transactions longues.
Les données	OLTP et ses transactions constituent la source originale de données.	Différentes bases de données OLTP deviennent la source de données pour OLAP.
Intégrité	La base de données OLTP doit maintenir la contrainte d'intégrité des données.	La base de données OLAP n'est pas fréquemment modifiée. Par conséquent, l'intégrité des données n'est pas affectée.
Normalisation	Les tables dans la base de données OLTP sont normalisées (3NF).	Les tables dans la base de données OLAP ne sont pas normalisées.
Requêtes	Des requêtes plus simples.	Des Requêtes plus complexes

Questions ?

Merci