Faculté des
Sciences Oujda
Département
Informatique

Master Ingénierie
Informatique
Semestre 2

Introduction au Big Data

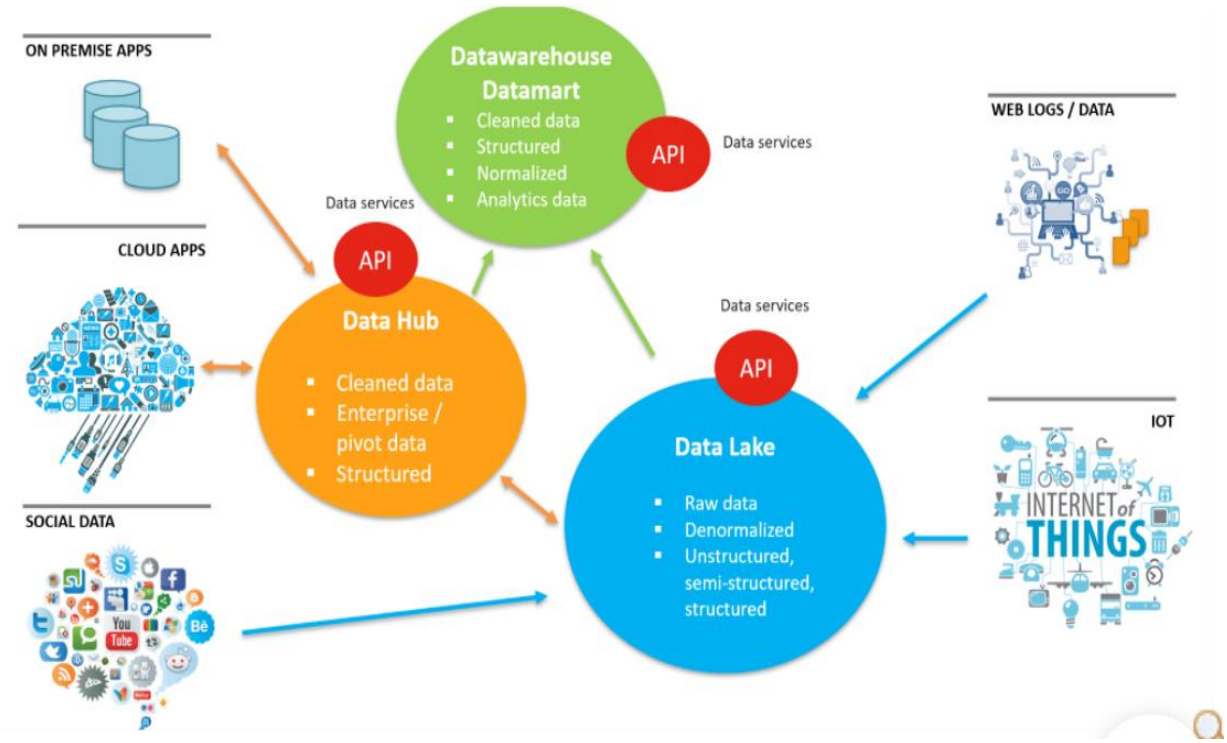
Pr ARRASSEN Ibtissam

2023-2024

Infos sur les données

Chaque jour nous générons plus de 2.5 trillions d'octets de données

- Plus de 90% des données dans le monde ont été crées au cours des dernières années
- 90% des données sont non structurées
- Sources des données:
  - Capteurs utilisés pour collecter les informations climatiques
  - Messages sur les médias sociaux
  - Images numériques et vidéos publiées en ligne
  - Enregistrements transactionnels d'achat en ligne
  - Signaux GPS de téléphones mobiles



Définition

Le Big Data (ou mégadonnées) représente les collectionsde données caractérisées par un volume, une vélocité et une variété ,

si grands que

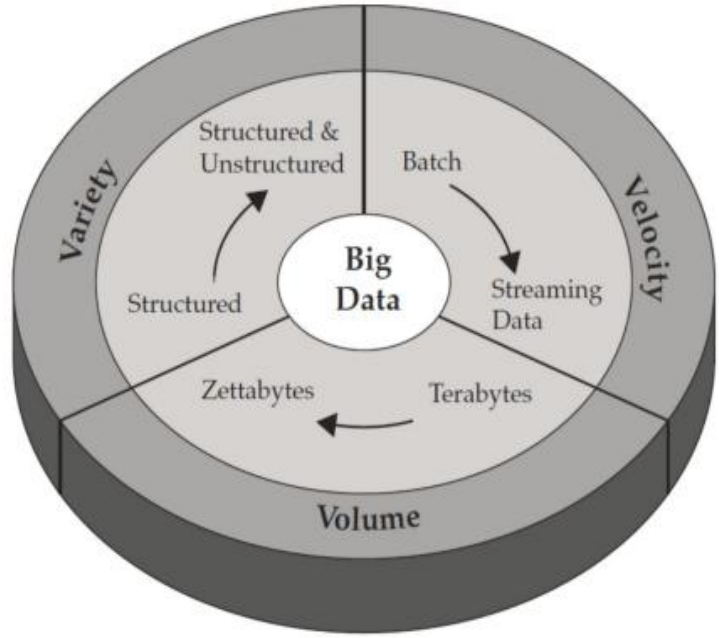
leur transformation en valeur utilisable requiert l'utilisation de technologies et de méthodes analytiques spécifiques."

Caractéristiques de ces données

● Volume - pas d'échantillonnage, on observe et mesure tout

● Vélocité - les données et les résultats sont souvent disponibles en temps réel

● Variété - puise dans les données textuelles, les photos, audio / vidéo et complète généralement les pièces manquantes en fusionnant plusieurs sources



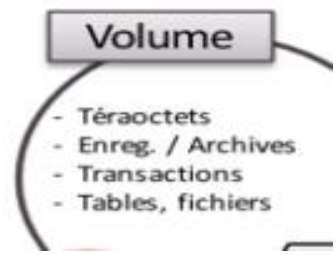
1) Volume

Aucune donnée n'est inutile. Certaines n'ont juste pas encore servi

Le prix de stockage des données a beaucoup diminué ces 40 dernières années:

• De $100 ,000 / Go (1980)

• À $0 .06 / Go (2021)



Problèmes:

• Comment **stocker** les données dans un endroit fiable, qui soit moins cher ?

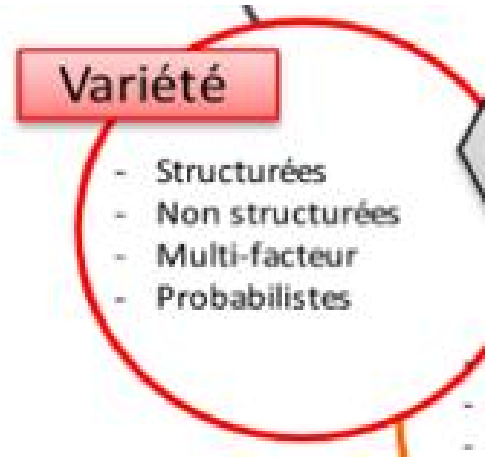
• Comment parcourir ces données et en **extraire** des informations facilement et rapidement ?

2) Variété (Variety)

Dans des **BD** ou dans des **DWH,** les données doivent respecter un format prédéfini pour être stocker.

La plupart des données existantes sont **non structurées ou semi-structurées**

Données sous plusieurs formats et types



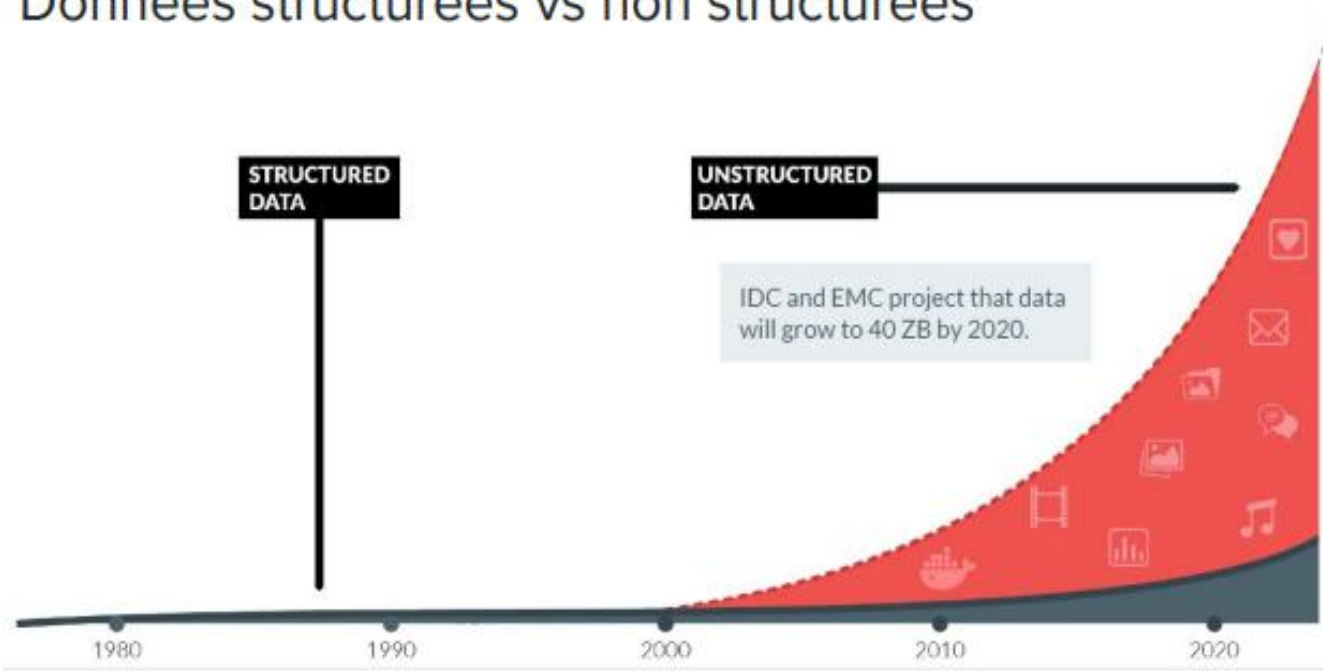
3) Vitesse (Velocity)

Rapidité d'arrivée des données

Vitesse de traitement

Les données doivent être stockées à l'arrivée, parfois même des Teraoctets par jour

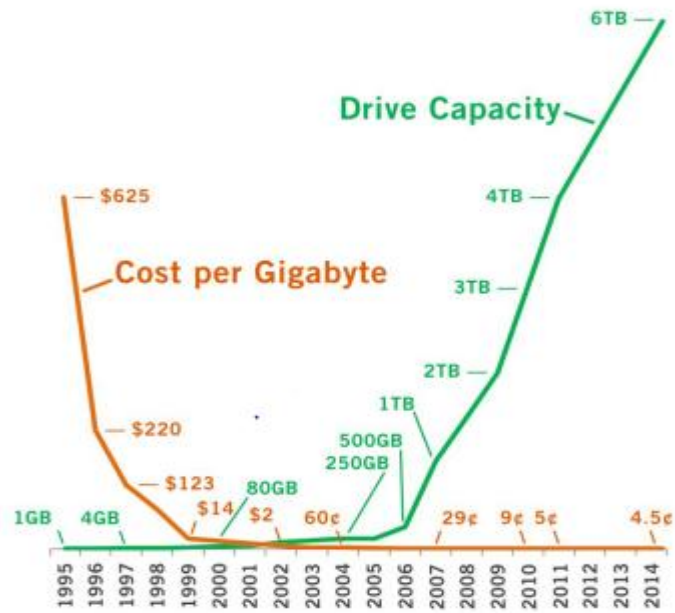
Données structurées vs non structurées



Big Data - Capacité de stockage

- Entre 2000 et 2006, la capacité des disques a augmenté par 10x alors que le prix par Gb a chuté du même ratio.

- Une augmentation de 100x à prix constant.



Google et le Big Data

- Problématique de Google:

Le moteur de recherche de Google doit restituer, en **temps quasi-réel**,

le résultat de **3 millions** de recherches effectuées par minute

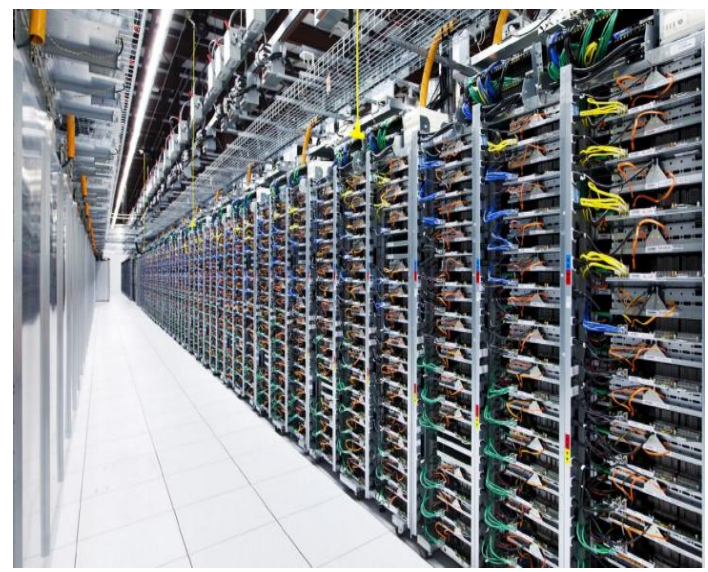
Solution

Google doit indexer toutes les pages web sur internet et rechercher dans chaque page les mots demandés par l'utilisateur

Solution Google pour résoudre les problèmes de stockage des données

- **Distribuer le stockage** des données et **paralléliser le traitement** de ces données sur plusieurs machines d'un Cluster

- en augmentant le nombre des machines du Cluster



• Google a décidé de construire un index inversé par mot-clé contenu dans chaque page web.

• Index inversé = (ensemble de mots-clés)

pour chaque mot recherché , on se serts de l'index pour identifier les pages où il se trouve.

• Exemple d'une page web contenant **5000 mots** :

Le moteur de recherche doit indexer les **5000 mots**.

→ Indexer **1000 pages** contenant chacune **5000 mots** implique

la construction d'un index de:

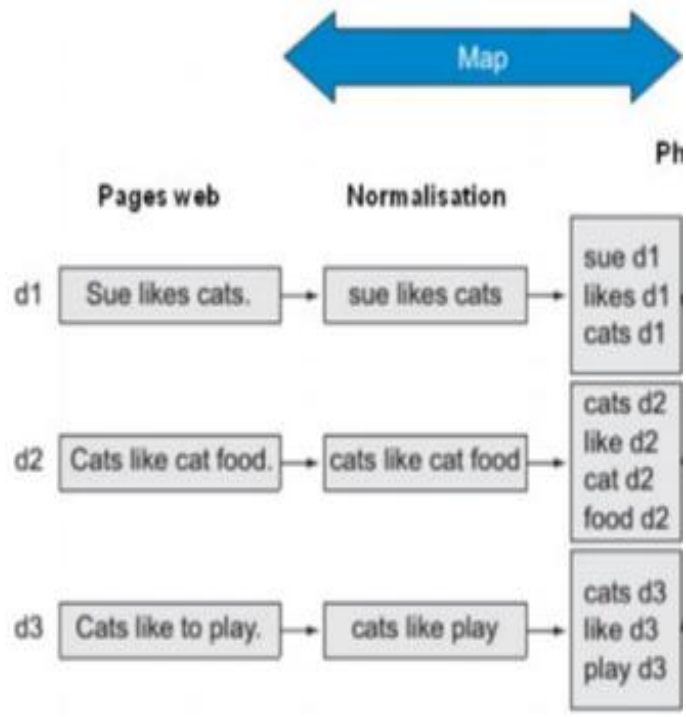
5×10^6 mots

Etape 1 : Map

Google a proposé une approche formée de 3 étapes :

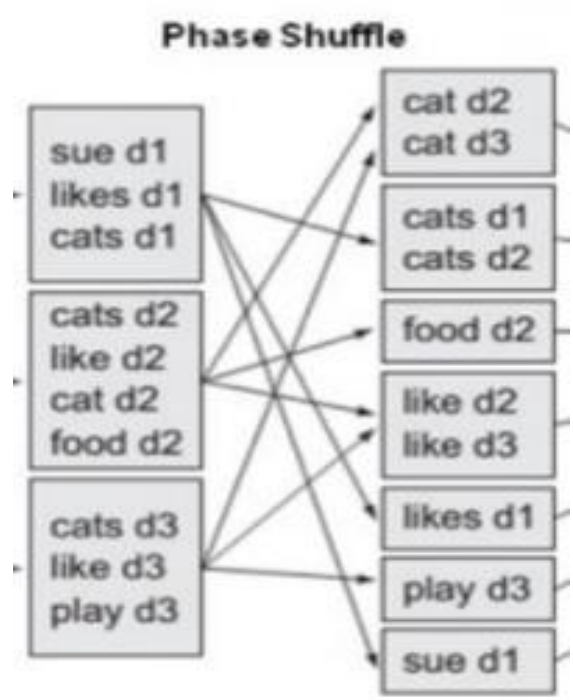
▪ chaque machine du Cluster attribue à chaque mot de la page web un indice = (titre de la page, adresse web, etc.).

▪ Cette tâche s'exécute, en même temps, sur toutes les machines du Cluster.



Etape 2: Shuffle

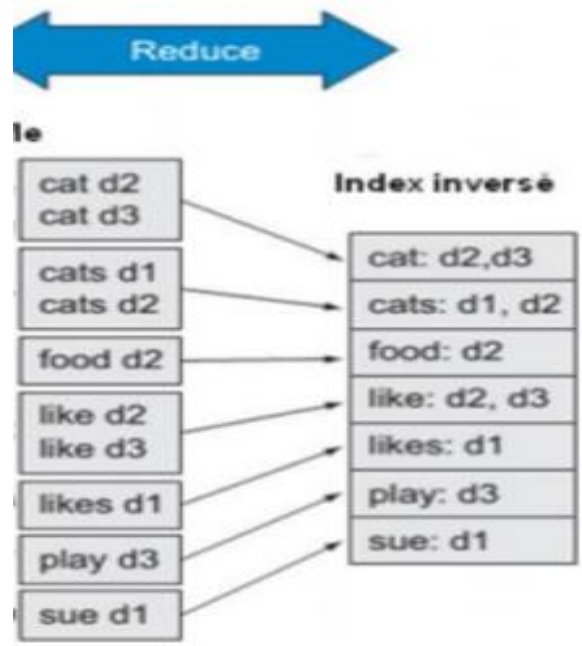
- trier par ordre **alphabétique** les mots auxquelles on a affecté un index.
- C'est une étape intermédiaire et permet de faciliter les tâches de la troisième étape.
- Exécutée sur chaque machine du Cluster



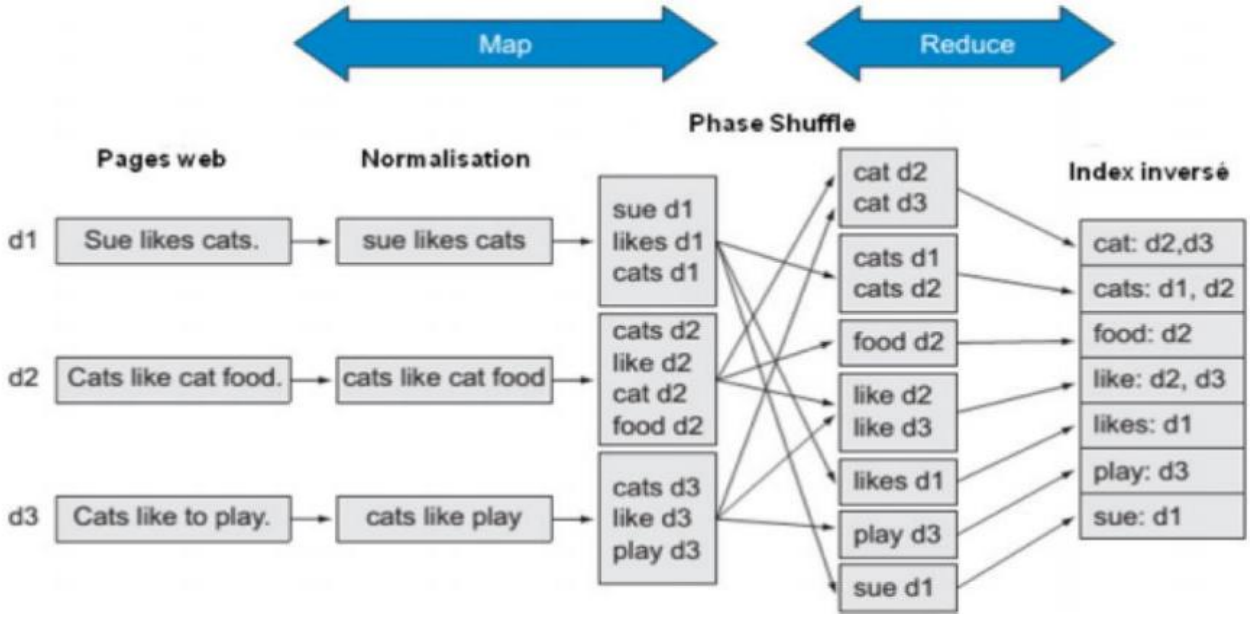
Etape 3 : Reduce

- Pour chaque mot dans l'ensemble des machines du cluster, on regroupe l'ensemble de ses index. Ainsi on construit l'index inversé.

Ce mode de traitement de données en trois étapes est appelé **MapReduce**



Approche conceptuelle de google



Création d'index

Hadoop platforme

en utilisant la solution fournie par Google, Doug Cutting et son équipe ont développé un projet Open Source appelé HADOOP.

Hadoop est une plateforme qui implémente le mode de traitement **MapReduce**

Le projet **Hadoop** consiste en deux grandes parties:

- Stockage distribué des données : HDFS (Hadoop Distributed File System) ;Diviser les données et les sauvegarder sur une collection de machines , appelée cluster

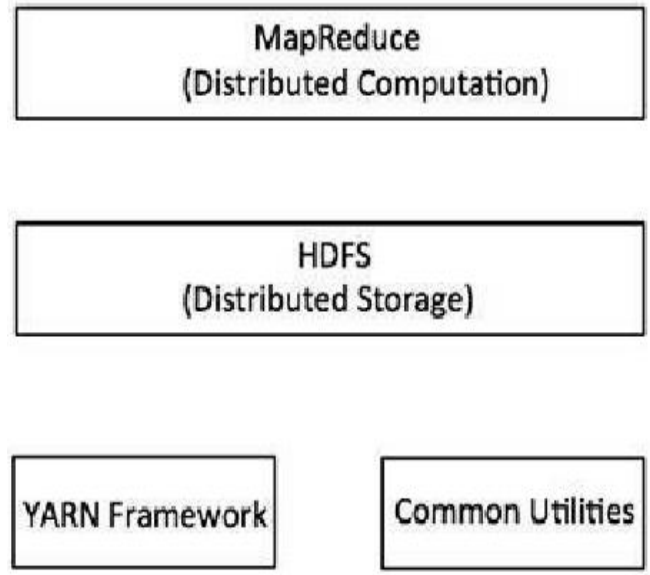
- Traitement parallèle des données (MapReduce/Yarn) directement là où elles sont stockées, plutôt que de les copier à partir d'un serveur distribué

Hadoop Architecture

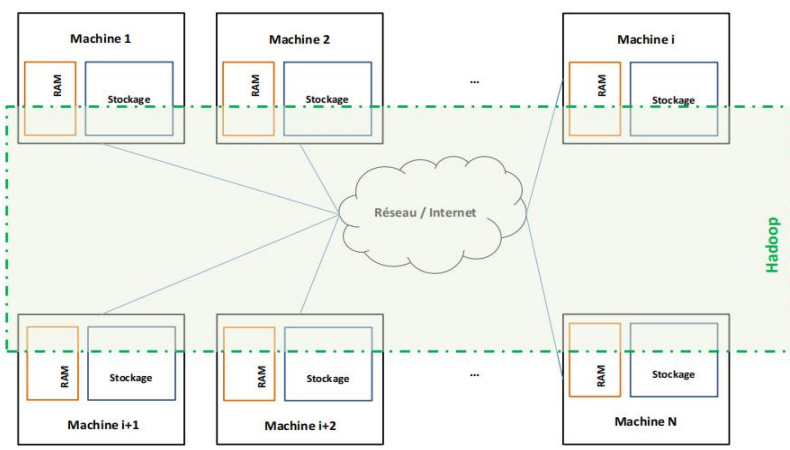
À la base, Hadoop a deux couches principales à savoir:

(a) Couche de traitement / calcul (MapReduce),

(b) Couche de stockage (Hadoop Distributed File System).



19



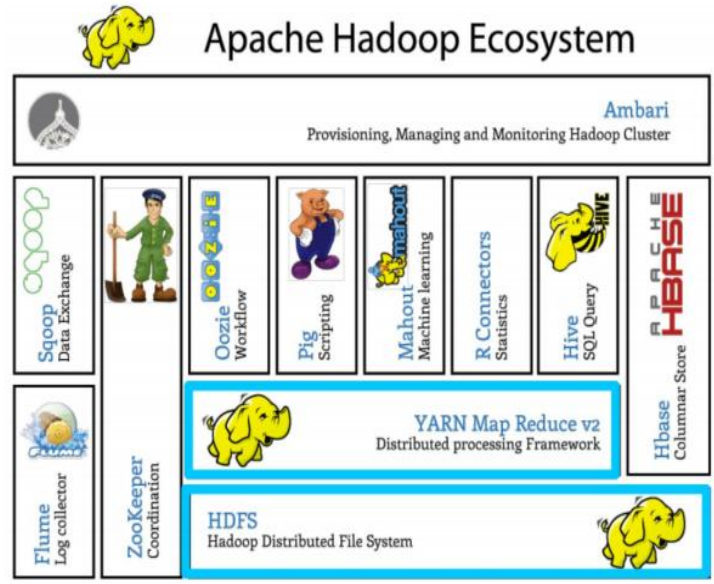
Principe :

- **Diviser** les données et les sauvegarder sur une collection de machines (cluster)
- **Traiter** les données directement là où elles sont stockées, plutôt que de les copier à partir d'un serveur distribué

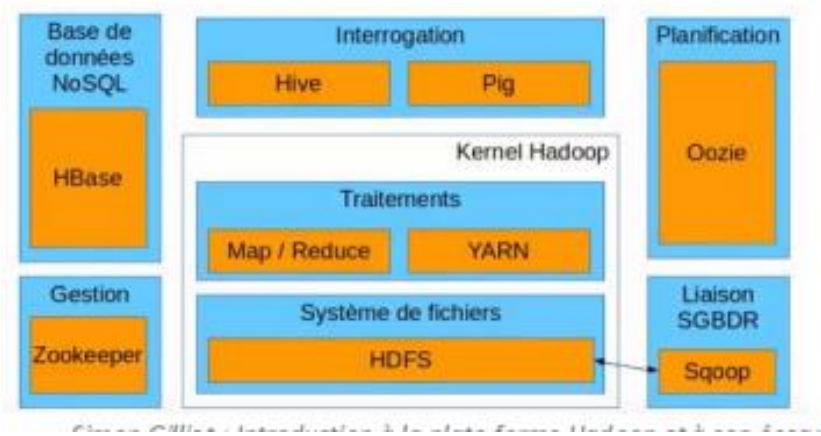
NB• Il est possible d'ajouter des machines au cluster, au fur et à mesure que les données augmentent

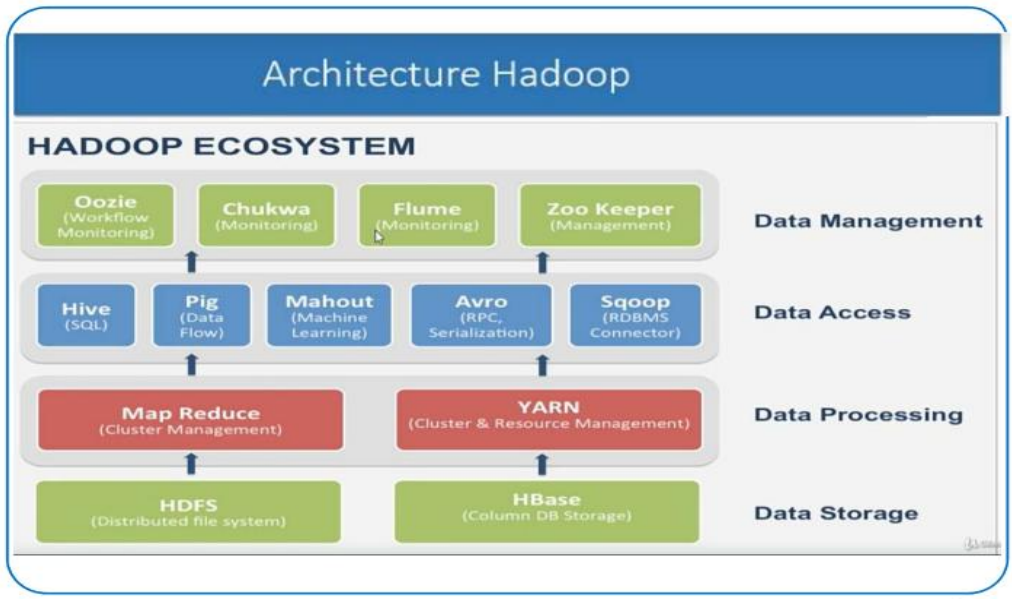
En plus des briques de base plusieurs outils existent pour:

- L'extraction et le stockage des données de/sur HDFS
- La simplification des opérations de traitement des données
- La gestion et coordination de la plateforme
- La gestion du cluster



écosystème Hadoop





Système de fichiers HDFS (Hadoop File System)

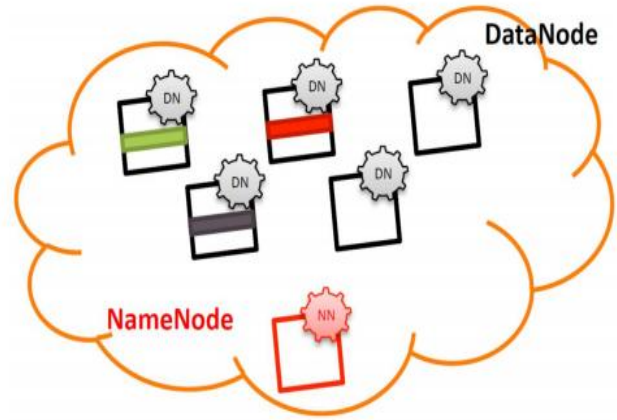
- HDFS est un système de fichiers écrit en **Java**, **distribué**, e**xtensible** et **portable**

- Permet de **stocker** de très gros volumes de données sur un **grand nombre** de machines (noeuds) équipées de disques durs → Cluster

Chaque fichier est décomposé en blocs de taille fixe. Par défaut 64 Mo. Chaque bloc est enregistré dans un noeud différent du cluster

DataNode :
▪ Démon sur chaque nœud du cluster

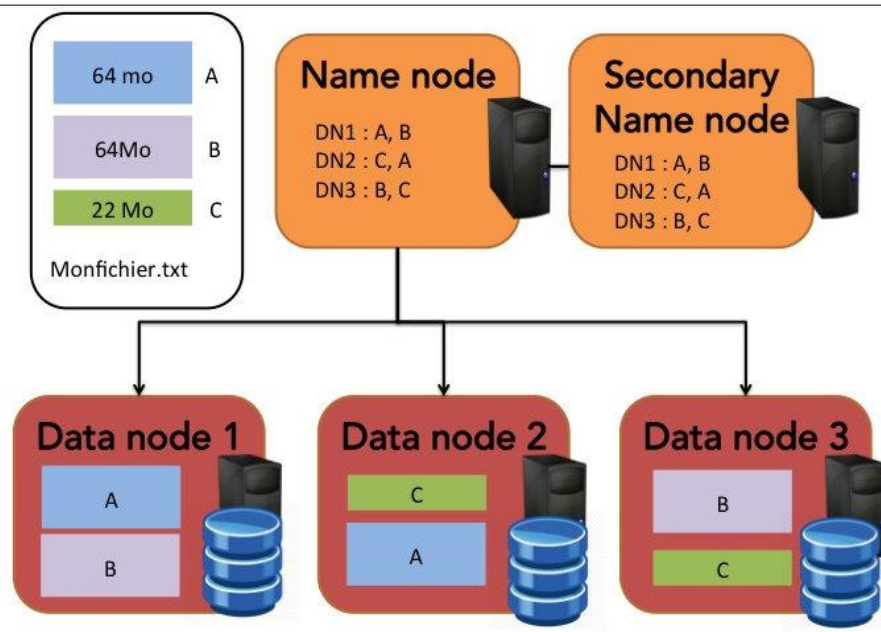
NameNode:
▪ Démon s'exécute sur une machine séparée
▪ sait comment sont décomposés les fichiers et sur quels **Datanodes** sont stockés ces blocs.



Exemple**:**

un fichier de **150 Mo** est décomposé en trois blocs : 2 blocs de **64 Mo** et 1 bloc de **22 Mo**.

Ces blocs sont répartis sur les différents data nodes.



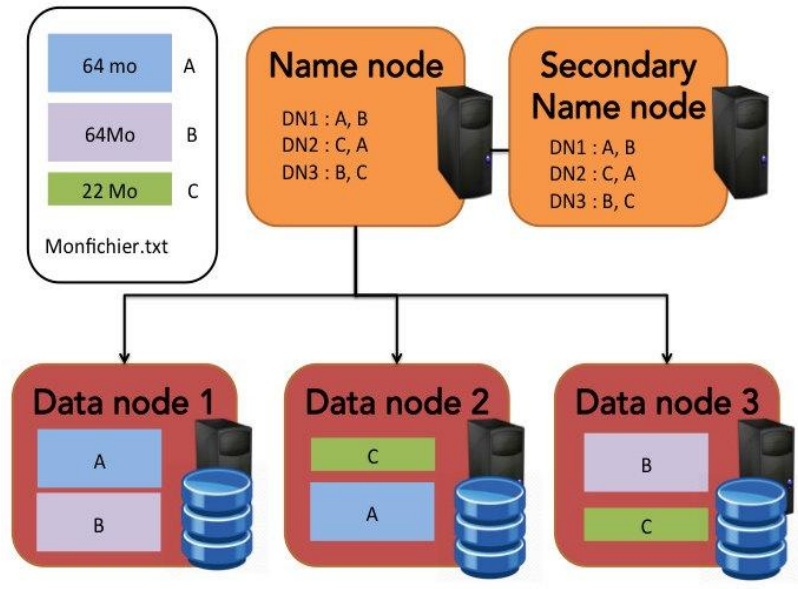
les **DN** ont très peu d'intelligence et ils ne servent qu'à stockér les données.
les **NN** stockent Les adresses des blocs ainsi que les noms des fichiers.

facteur de réplication

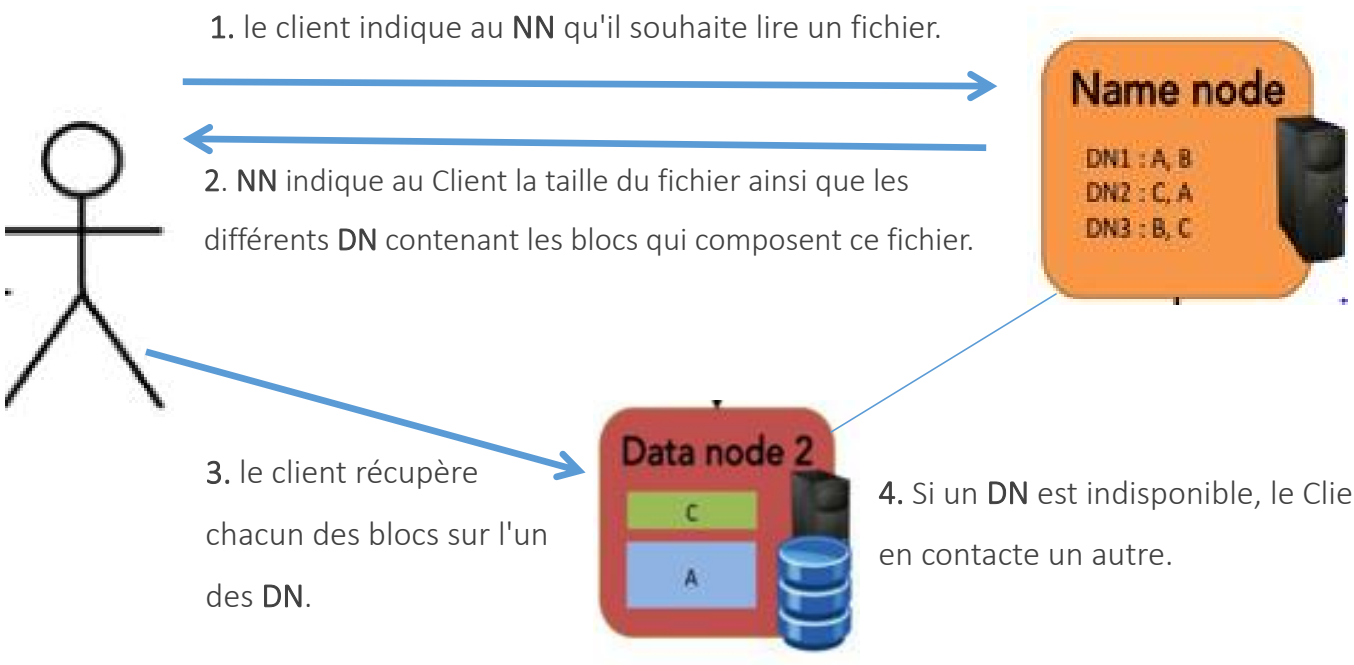
Hadoop réplique chaque bloc 2 fois (par défaut)

- Il choisit 2 noeuds au hasard, et place une copie du bloc dans chacun d'eux

- Si le noeud est en panne, le NN le détecte, et s'occupe de répliquer encore les blocs qui y étaient hébergés pour avoir toujours 2 copies stockées



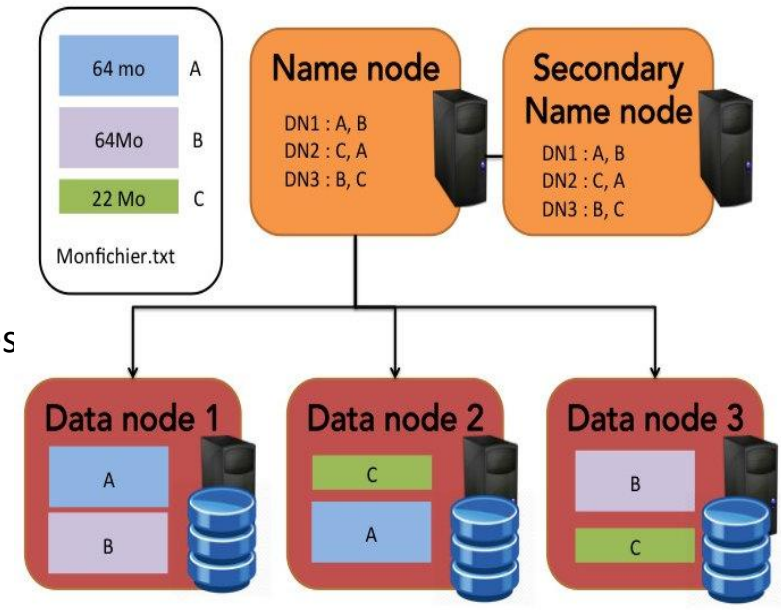
Exemple



Si le NameNode a un problème ?

- Si c'est un problème d'accès (réseau), les données sont temporairement Inaccessibles
- Si le disque du **NN** est défaillant, les données seront perdues à jamais

l 'existance d'un NameNode Secondaire est nécessaire



les fichiers de configuration de Hadoop
Core-site.xml and hdfs-site.xml:

The core-site.xml file informs Hadoop daemon where NameNode runs in the cluster.

It contains the configuration settings for Hadoop Core such as I/O settings that are common to HDFS and MapReduce.

```
<property>
        <name>hadoop.tmp.dir</name>
        <value>/data/hdfs/tmp<</value>
        <description>Where Hadoop will place all of its working files</description>
</property>
<property>
        <name>fs.defaultFS</name>
        <value>hdfs://master:9000</value>
        <description>Where HDFS NameNode can be found on the network
        </description>
</property>
```

les fichiers de configuration de Hadoop
Core-site.xml and hdfs-site.xml

the **hdfs-site.xml** file contains the configuration settings for HDFS daemons:

- **NameNode**,
- **Secondary NameNode**, and
- **DataNodes**.

in **hdfs-site.xml** we can specify **default block replication** and **permission checking** on HDFS.

The actual number of replications can also be specified when the file is created. The default is used if replication is not specified in create time.

The three main hdfs-site.xml properties are:

- **dfs.name.dir** gives the location where NameNode stores the metadata (FsImage and edit logs). And also specify where DFS should locate – on the disk or in the remote directory.
- **dfs.data.dir** gives the location of DataNodes where it stores the data.
- **fs.checkpoint.dir** is the directory on the file system. On which secondary NameNode stores the temporary images of edit logs. Then this EditLogs and FsImage will merge for backup.

T

Hadoop: trois modes de fonctionnement

• **Mode local (standalone)** : Hadoop fonctionne sur une seule machine et tout s'exécute dans la même JVM (Java Virtual Machine). En mode local le système de gestion de fichiers utilisé est celui du système hôte (Exemple Linux: ext3, ext4 ou xfs) et non HDFS.

• **Mode pseudo-distribué:** Hadoop fonctionne sur une seule machine, mais chacun des daemons principaux Hadoop tel que hdfs, yarn, MapReduce, etc., s'exécutera comme un processus Java distinct(s'exécute dans sa propre JVM). Le système de fichier utilisé est HDFS dans ce mode. Ce mode est utile pour le développement.

• **Mode totalement distribué:** c'est le mode d'exécution réel d'Hadoop. le système de fichiers distribué et les daemons fonctionnent sur des machines différentes

avec **au moins deux machines ou plus** en cluster

TP avec HADOOP

TP1 : Hadoop installation in Standalone Mode

Exemple : **wordcount**

Compter le nombre total de mots contenus dans les fichiers donnés dans le dossier «**input**»

```
arrassen@arrassen-VirtualBox:~$ cd output
arrassen@arrassen-VirtualBox:~/output$ ls
part-r-00000  _SUCCESS
arrassen@arrassen-VirtualBox:~/output$ cat part-r-00000
"AS     2
"Contribution"  1
"Contributor"   1
"Derivative     1
"Legal  1
"License"       1
"License");     1
"Licensor"      1
"NOTICE"        1
"Not    1
"Object"        1
"Source"        1
"Work"  1
"You"   1
"Your") 1
"[]"    1
```

TP avec HADOOP

TP 2 : Hadoop Distributed File System (HDFS) et API Java

Exemple : purchase.txt

Afficher le nom de la ville dans laquelle il y a eu le plus grand nombre de paiement par carte bancaire.

```
hadoop@node-master:~/TP_Hadoop/Parcours_Data$ make
hadoop jar projet.jar
2020-12-13 16:12:46,295 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your
platform... using builtin-java classes where applicable



Parcours du fichier: storage/purchases.txt
Villes avec le plus grand nombre de paiement par CB :
{Rochester=22758, Fort Wayne=22758}
```

Commandes hdfs

Instruction	Fonctionnalité
hdfs dfs –ls	Afficher le contenu du répertoire racine
hdfs dfs –put file.txt	Upload un fichier dans hadoop (à partir du répertoire courant de votre disque local)
hdfs dfs –mkdir myinput	Créer un répertoire
hdfs dfs –get file.txt	Download un fichier à partir de hadoop sur votre disque local
hdfs dfs –tail file.txt	Lire les dernières lignes du fichier
hdfs dfs –cat file.txt	Affiche tout le contenu du fichier
hdfs dfs –mv file.txt newfile.txt	Renommer (ou déplacer) le fichier
hdfs dfs –rm newfile.txt	Supprimer le fichier

Projets à réaliser

Vous allez travailler sur le fichier « **purchases.txt** » existant sur votre cluster et don la structure est donnée ci-après.

Date	Heure	Ville	Achat	Prix	Type de paiement

En utilisant MapReduce, pour chacun des objectifs ci-après dédiez un projet :

1) Objectif 1 :

Sélectionner la ville avec le plus grand nombre de ventes.

2) Objectif 2 :

Afficher le montant total des ventes par ville.

3) Objectif 3 :

Sélectionner la ville avec le plus grand nombre de ventes des articles de type « Children's Clothing ».

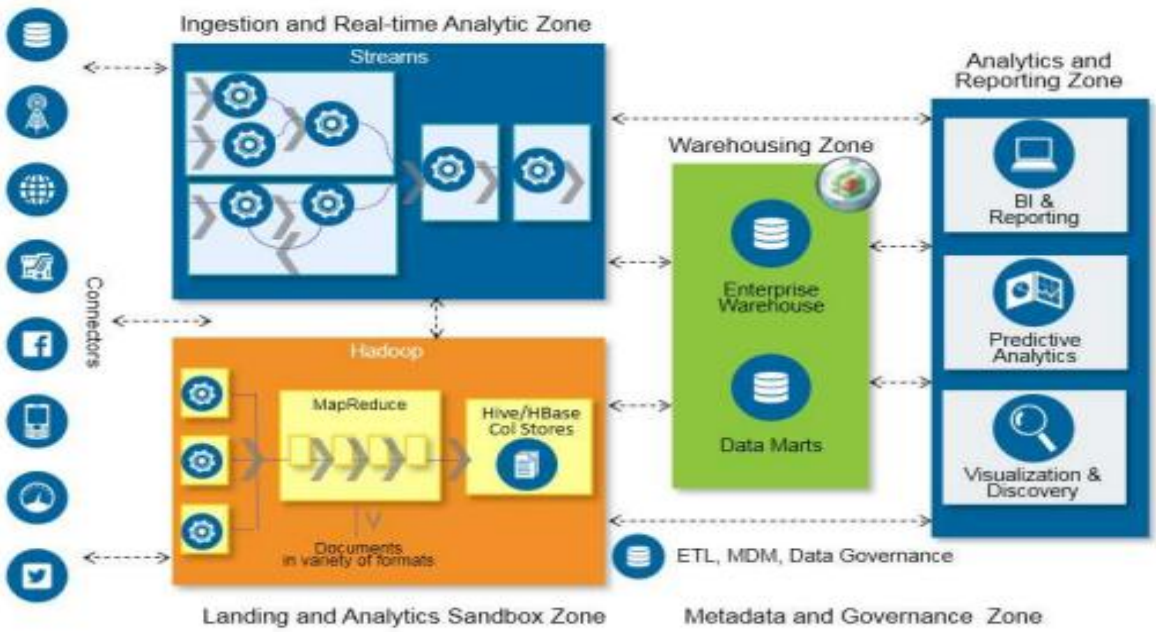
4) Objectif 4 :

Afficher le montant total des transactions effectuées par chaque type de carte bancaire.

5) Objectif 5 :

Sélectionner le nom de la ville avec le plus grand nombre de paiements effectués par carte bancaire du type « MasterCard ».

HADOOP et DWH



Course Guide Volume 1: IBM BigInsights Foundation v4.0 (Course code DW613 ERC 1.0)