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 outils de collecte, de traitement et d’exploitation des données
- Savoir utiliser les outils de visualisation des données (Dataviz)
- Piloter et maîtriser les risques des projets

Partie 2 : Outils de collecte, de traitement et d’exploitation des données

Partie 2 : Outils de collecte, de traitement et d’exploitation des données

- Les types de bases de données
- Les propriétés des bases de données
- L’exploitation des données
- TP : Hadoop MapReduce

Les types de bases de données (1/2)

- **Les bases de données hiérarchiques**
  - Arborescence (1parent/ 1fils)
- **Les bases de données réseau**
  - Graphe (n parent/ m fils)
- **Les bases de données orienté Object**
  - Programmation Object
- **Les bases de données relationnelles**

https://www.lebigdata.fr/base-de-donnees

Les types de bases de données (2/2)

- **Les bases de données NoSQL**
- **Les bases de données distribuées**
- **Les bases de données orientée graph**

https://www.lebigdata.fr/base-de-donnees

Stockage de la donnée : **Relationnelles (SQL)**

Relationnelles (SQL) : stockage en table

- MySql
- SQL Server
- PostgreSql
- Oracle
- SqLite …….

Stockage de la donnée : **Non relationnelles (No SQL)**

Non relationnelles (No SQL) : Bases de type **documents (stockage Json/xml)**

```
{
  "name" : "IRIS",
  "address"  : "7-8 des impasses, Paris",
  "filieres" : [
    "Information",
    "Management"
  ]
}
```

CouchDB

Requétage plus complet

Flexibilité

Evolutif au cours du temps

Duplication des données

Cohérence pas forcément assurée

mongoDB

https://www.oracle.com/fr/database/base-donnees-relationnelle-difference-non-relationnelle.html

Stockage de la donnée : les bases de données

Base de données de colonnes : stocke les données par colonne et non par ligne (adapté pour **l'OLAP**)

cassandra

ClickHouse

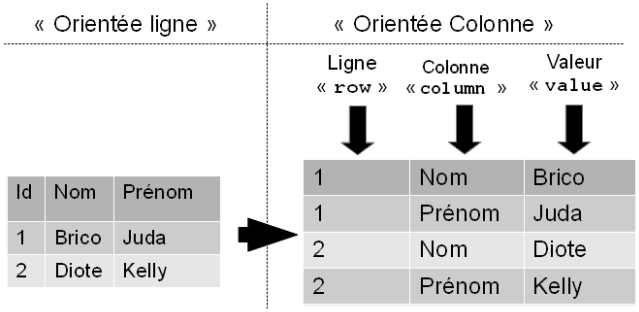
Capacité de stockage accrue

Accès rapide aux données

Efficace surtout pour des données de même type et similaires

Requêtage limité

APACHE HBASE



https://www.oracle.com/fr/database/base-donnees-relationnelle-difference-non-relationnelle.html

Stockage de la donnée : les bases de données

Bases de données en cache : stocke les données en mémoire vive (RAM)

redis

ACTIVEPIVOT
In-memory analytical database

MEMCACHED

https://www.oracle.com/fr/database/base-donnees-relationnelle-difference-non-relationnelle.html

Stockage de la donnée : les bases de données

Bases de données graphe: basées sur la théorie des graphes, sont gérées par noeuds, relations et propriétés.

Source : neo4J

+

Adapté à la gestion de données relationnelles

Architecture modelable

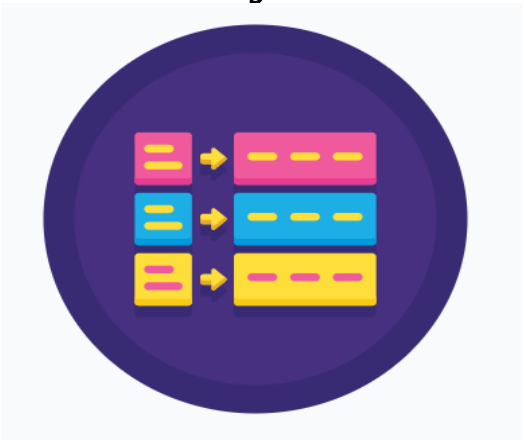
-

Architecture limitée à certains cas

https://www.oracle.com/fr/database/base-donnees-relationnelle-difference-non-relationnelle.html

Stockage de la donnée : les bases de données

Bases de données key-value



Source : inconnu

amazon DynamoDB

redis

\+
Facilement scalable

Temps de réponse en écriture / lecture très bas

\-
Mises à jour compliquées

Requêtes rudimentaires

https://www.oracle.com/fr/database/base-donnees-relationnelle-difference-non-relationnelle.html

Propriétés des bases de données : ACID vs BASE

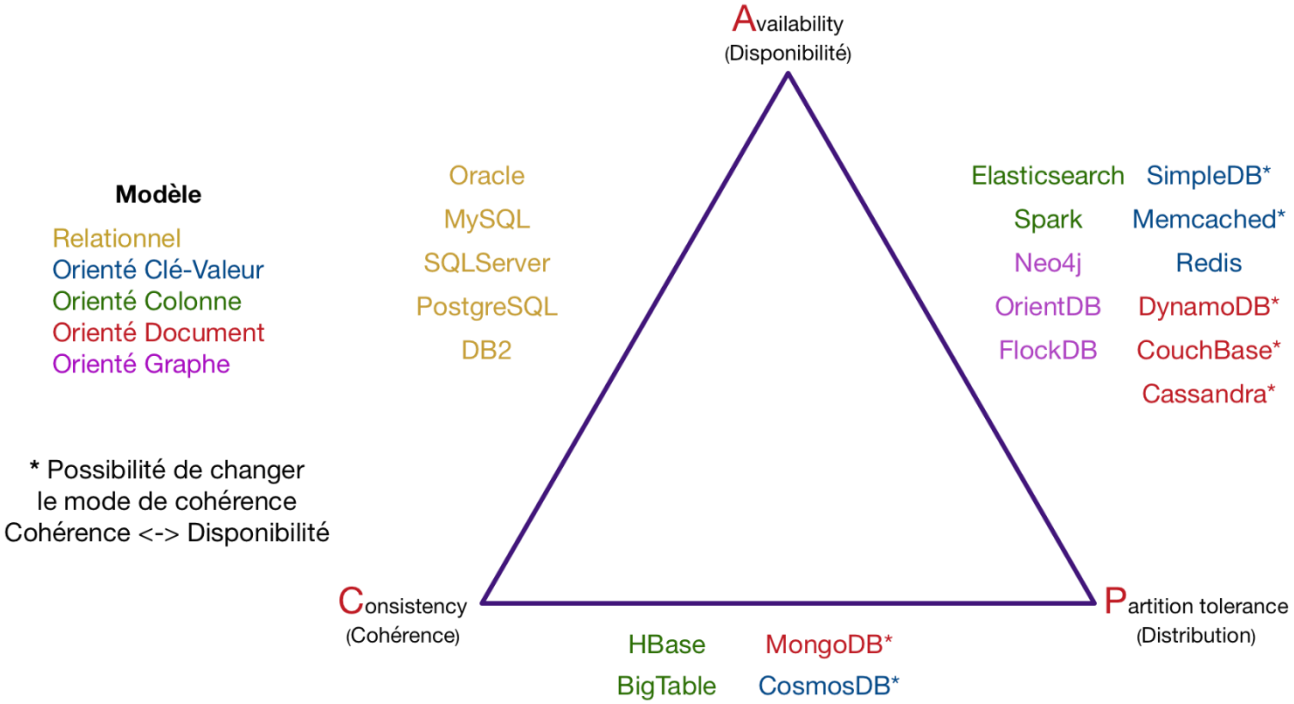
Relationnelle

- ***A**tomicité*
- ***C**ohérence*
- ***I**solation*
- ***D**urabilité*

Non Relationnelle

- **B***asically* ***A****vailable*
- **S***oft-state*
- **E***ventually consistent*

Théorème de CAP



https://user.oc-static.com/upload/2017/05/26/14958217637026_triangleCAP.png

Limites des bases de données relationnelles

- Incapable de gérer de très grands volumes de données (de l'ordre du péta-octet) (1)
- Impossible de gérer des débits extrêmes (plus que quelques milliers de requêtes par seconde) (2)
- Inadapté au stockage non structuré (3)
- Les propriétés ACID entraînent de sérieux surcoûts en latence, accès disques, temps CPU (4)

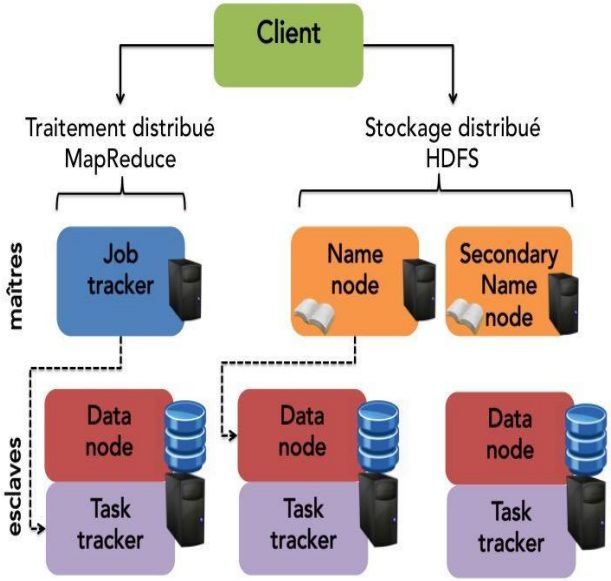
Limites des bases de données relationnelles

Big data
(non relationnelles
NoSql)

Comment pallier
à ces limites ?

Spécificité big data

- **(2) Architecture Maitre/Esclave : nœud « maitre » distribue des tâches aux nœuds « escalve »**
- **Calcul et stockage distribué et répliqué (1)**
- **Gestion temps réel et batch**
- **Stockage structuré et non structuré (3)**
- **BASE (4) vs ACID**



Exemple : architecture hadoop

Source : https://user.oc-static.com/upload/2017/03/21/149009497754_Diapositive4Hadoop.jpeg

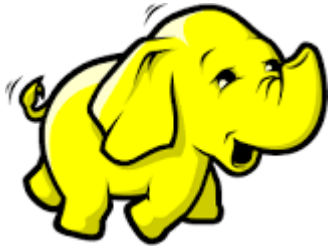
- https://openclassrooms.com/fr/courses/4297166-realisez-des-calculs-distribues-sur-des-donnees-massives/4308656-familiarisez-vous-avec-hadoop

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**



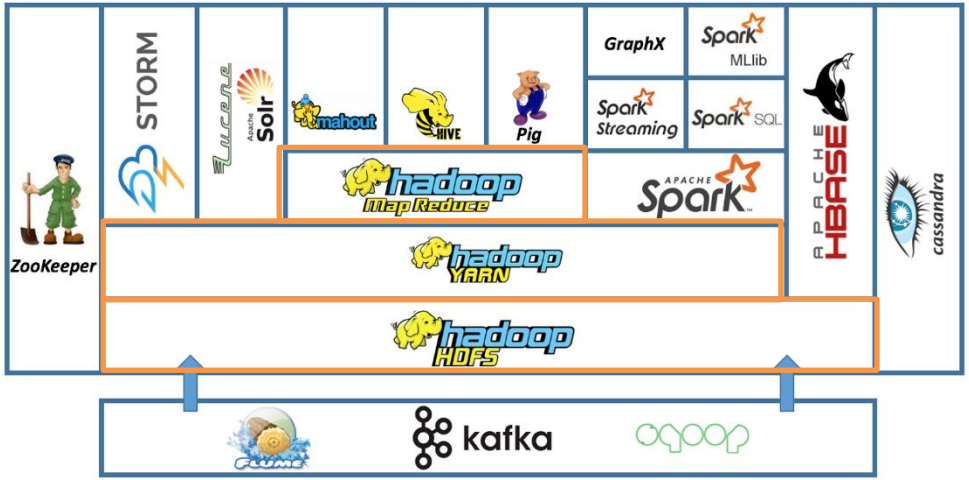
Module de base Hadoop : HDFS

Hadoop Distributed File System

Système de fichier

Stockage distribué

1To 1To 1To 1To 1To 1To 1To 1To



Le paradigme MapReduce

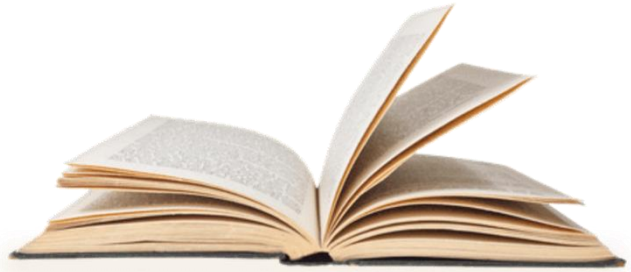
- Modèle de programmation pour permettre d'appliquer un calcul (Map) et d'agréger les résultats (Reduce).

- 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 MapReduce

- Comment compter le nombre d'occurrence d'un mot dans un livre ?
- Quelle stratégie pour une personne ?
- Quelle stratégie pour plusieurs personnes ?
- Comment compter le nombre d'occurrence de tous les mots dans un livre ?



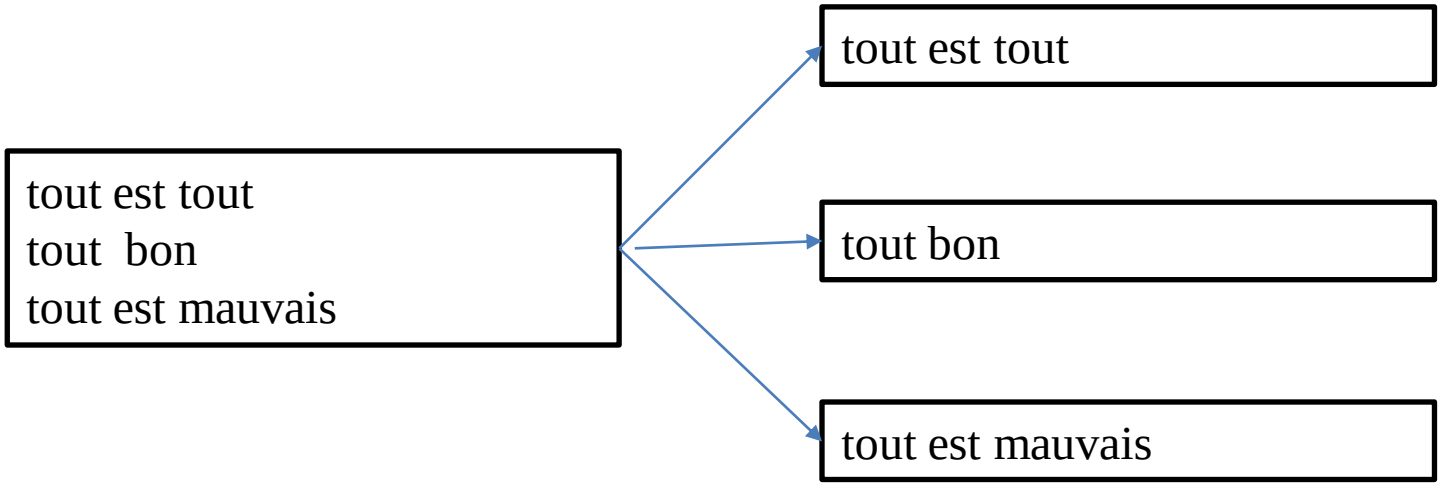
Le paradigme Map-Reduce : Exemple words count

Le texte

Tout est tout
Tout bon.
Tout est mauvais

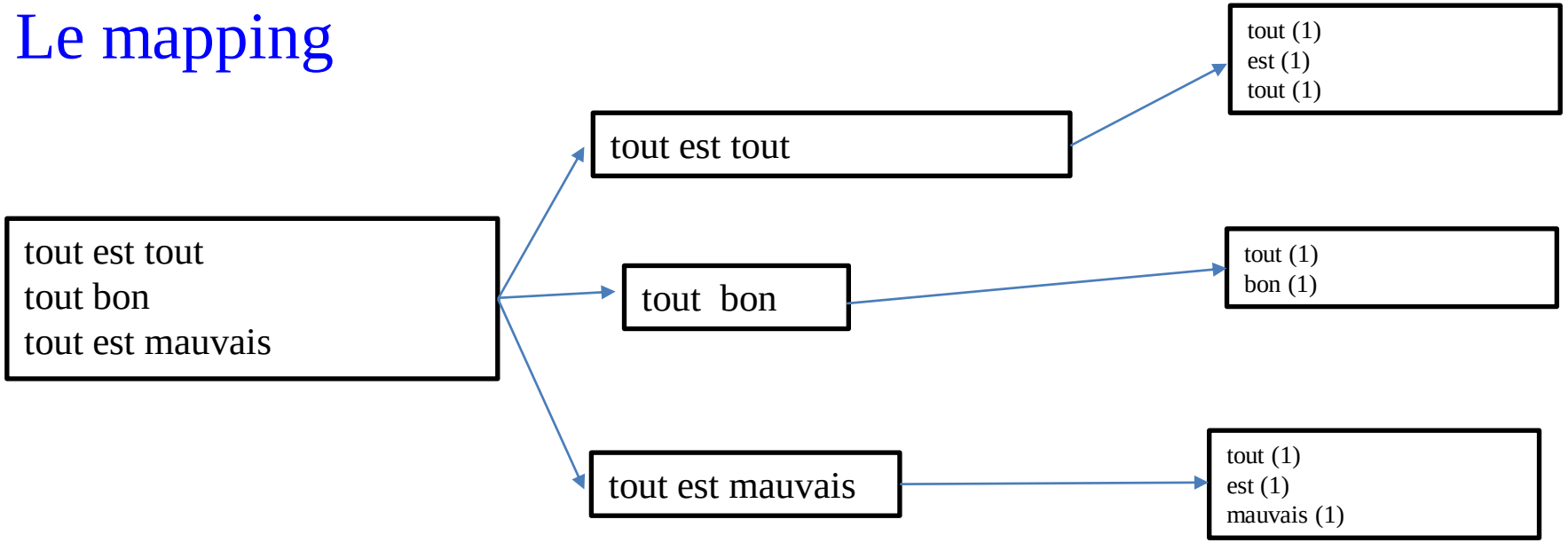
Le paradigme Map-Reduce : Exemple words count

Le splitting



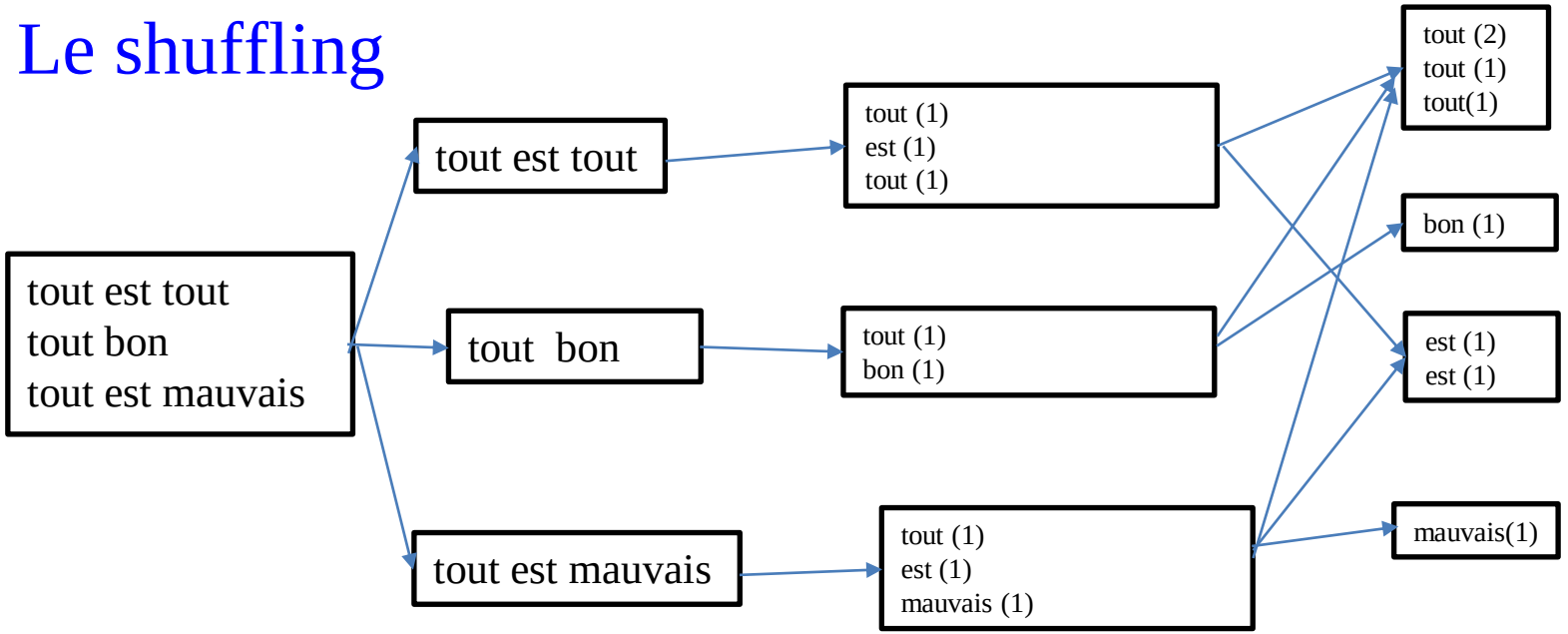
Le paradigme Map-Reduce : Exemple words count

Le mapping



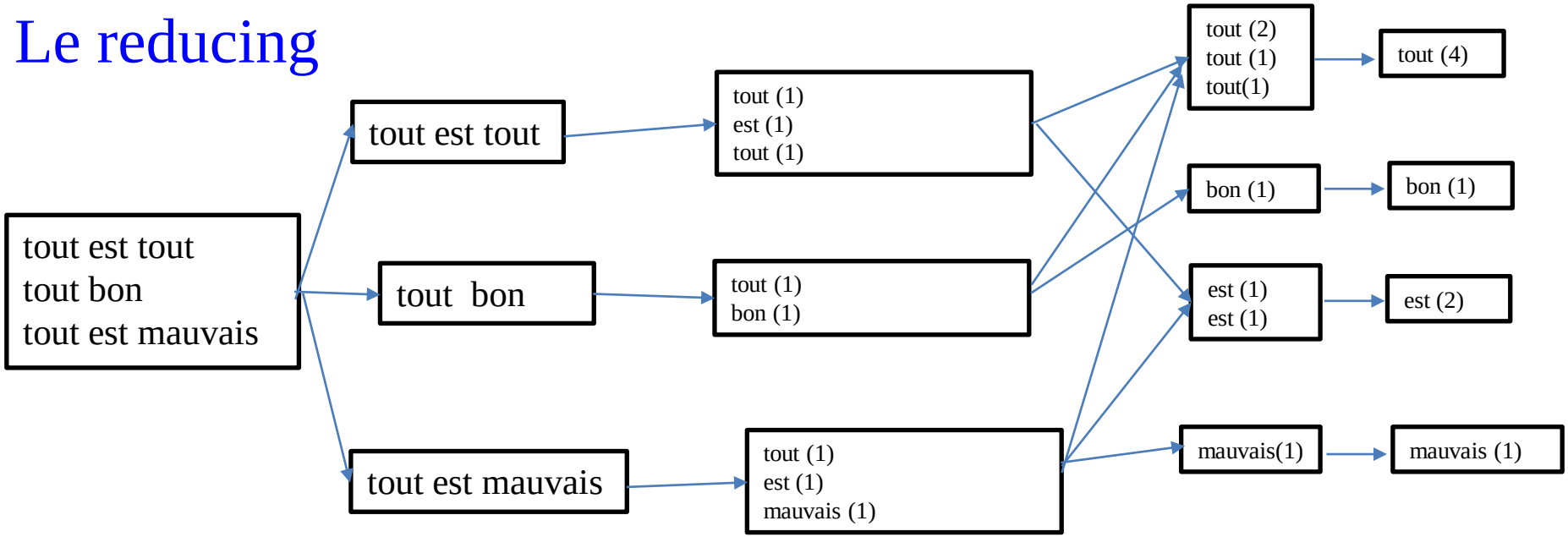
Le paradigme Map-Reduce : Exemple words count

Le shuffling

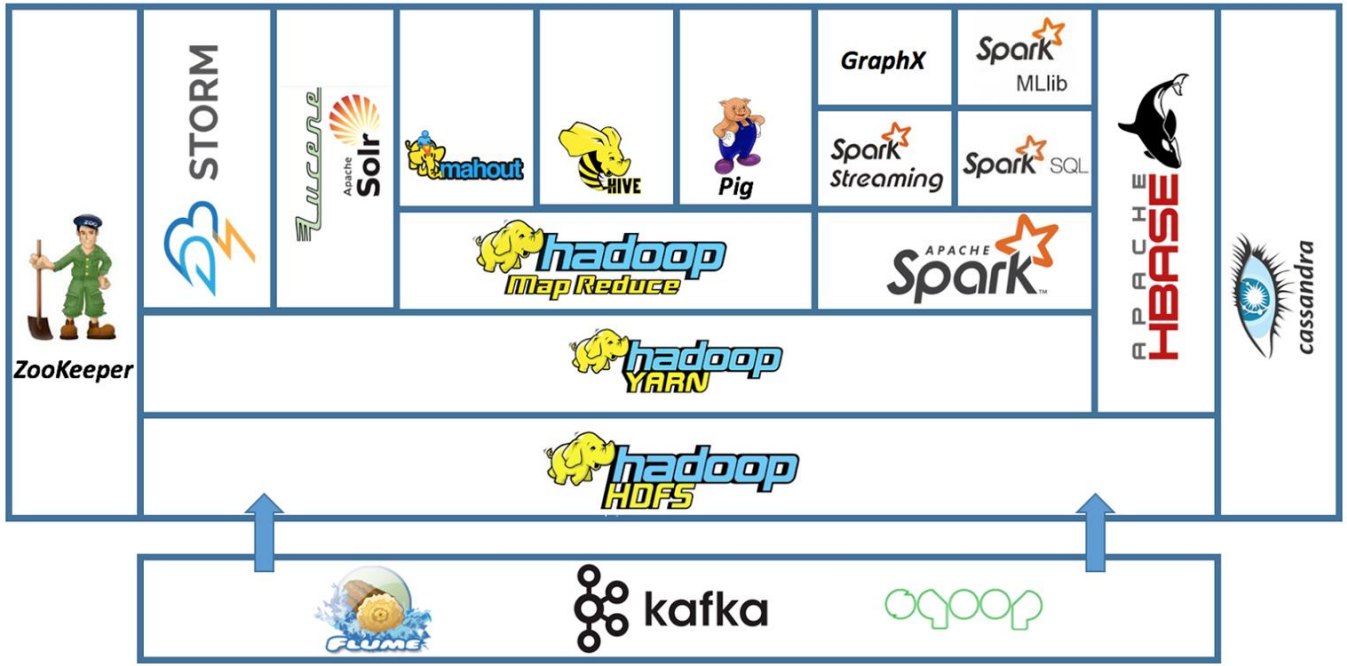


Le paradigme Map-Reduce : Exemple words count

Le reducing



Framework Hadoop



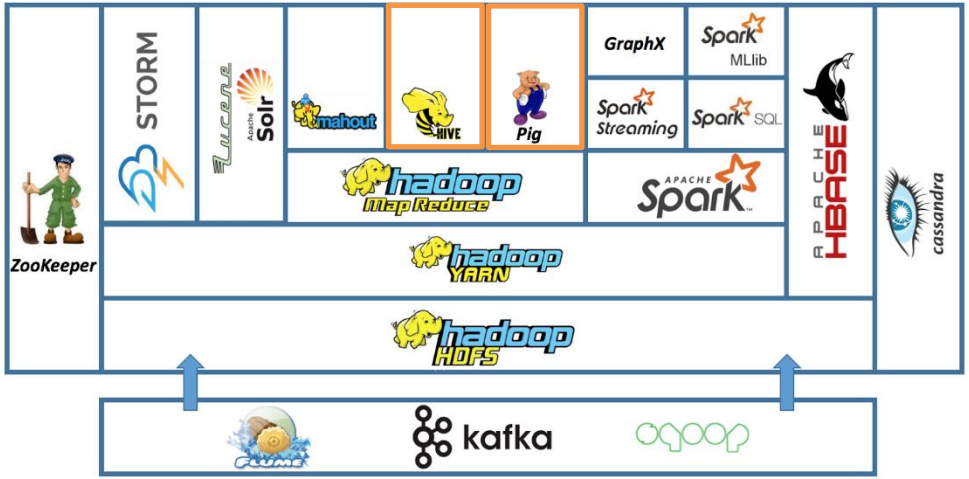
source : http://blog.newtechways.com/2017/10/apache-hadoop-ecosystem.html

Module Hadoop : Pig et Hive

- ❑ Pig est une plateforme haut niveau pour la création de programme MapReduce utilisé avec Hadoop.
- ❑ Le langage de cette plateforme est appelé le Pig Latin

Apache Pig

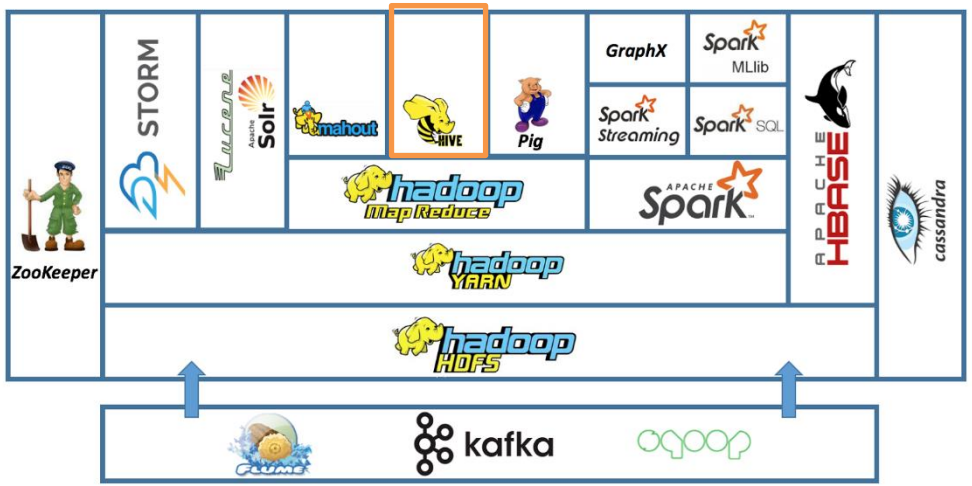
- ✔ Utilise l'évaluation paresseuse,
- ✔ Utilise des extract, transform, load (ETL),
- ✔ Stocke des données (tout moment d'un pipeline)
- ✔ Déclare le plan d'exécution,
- ✔ Exécute le workflow subdivisé selon un graphe, au lieu d'une exécution purement séquentielle.



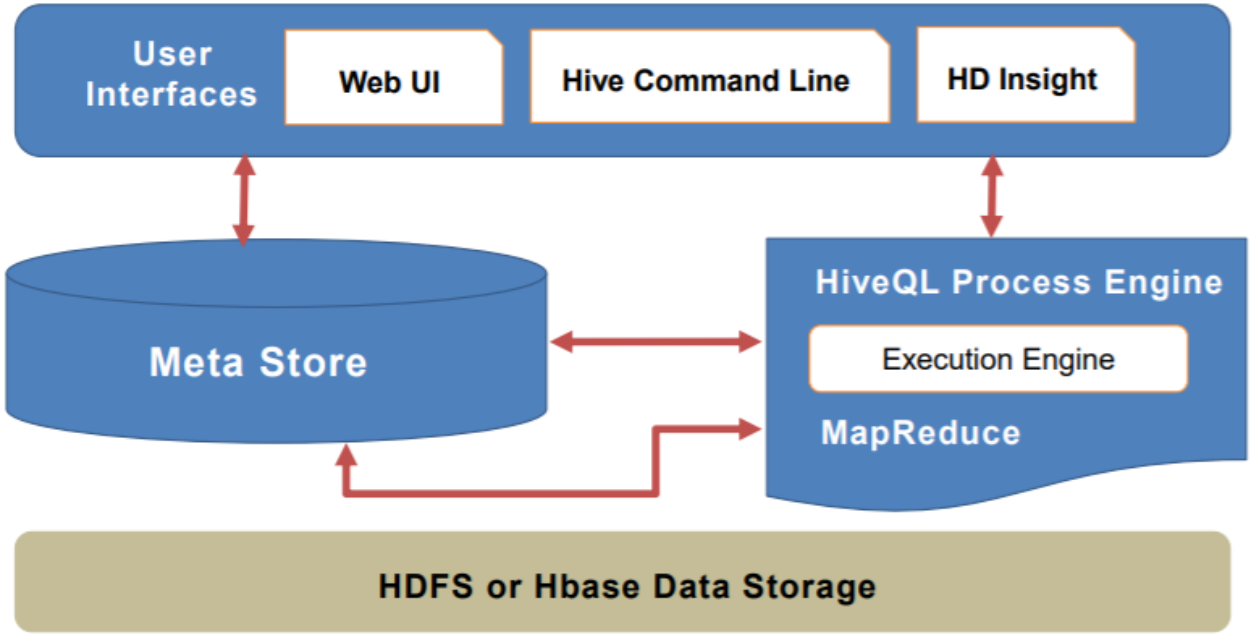
Hive

- 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



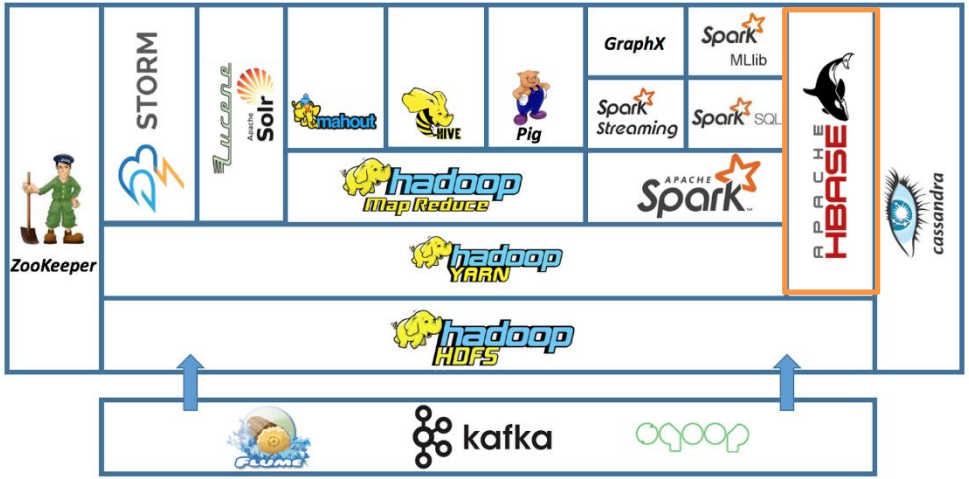
Hive : hive vs base relationnelle

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

Module Hadoop : HBase

- Système de gestion de base de données non-relationnelles distribué, écrit en Java, disposant d'un stockage structuré pour les grandes tables

- ✔ Orienté Colonne
- ✔ Architecture Maitre/Esclave
- ✔ Utilise HDFS pour faciliter la distribution



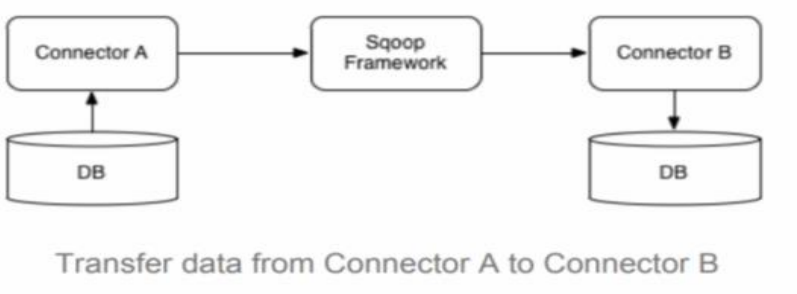
Collecte de la donnée : Sqoop

• **Sqoop** permet l'import/export des données depuis/vers des entrepôts de données externes, en particulier les bases relationnelles : **Ingestion de données**

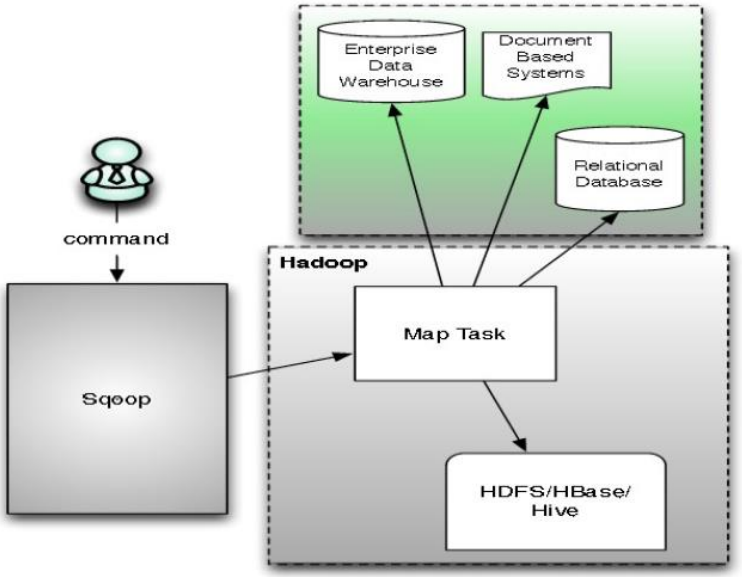
- Commandes prédéfinies
- Utilise les jobs Map de MapReduce
- Prend en charge le chargement incrémental (incremental loads)
- Import sous forme de csv ou dans une base de données
- Nombreux connecteurs disponibles
- Mise à jour des données sans recharger toute la table
- Ecris en Java

Sqoop : Fonctionnement

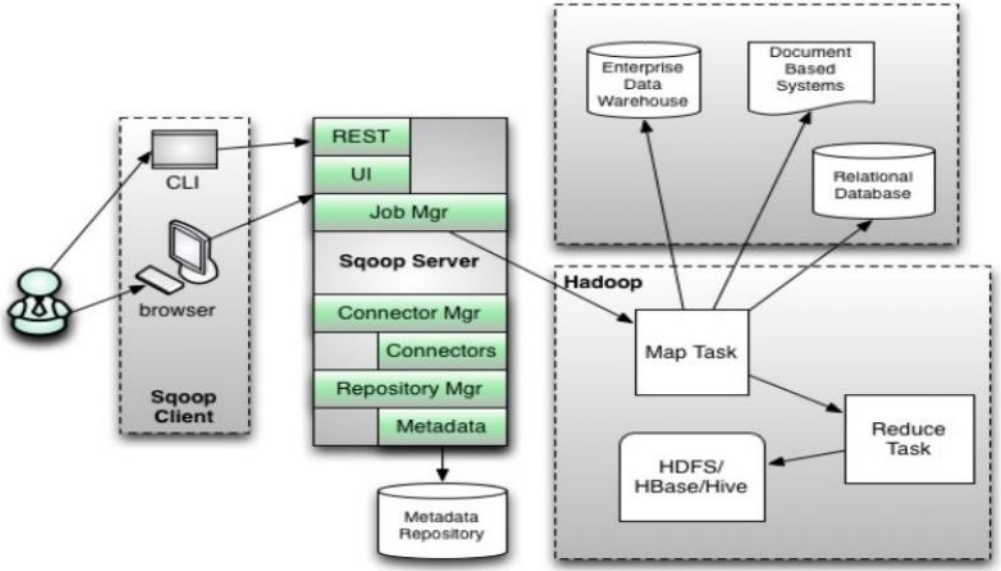
- Données divisées en partitions
- Mappers transfer data
- Les types sont déterminées via le meta-data de la base de données
- Export en plusiers format (CSV, Avro, parquet)
- Peut importer dans Hive, Hbase……



Sqoop 1 : Architecture



Sqoop 2: Architecture



Sqoop : Exemple

```
sqoop import
--connect jdbc:mysql://db.foo.com:3306/bar \
--table EMPLOYEES \
--columns "employee_id,first_name,last_name,job_title"
--where "start_date > '2018-01-01'"
```

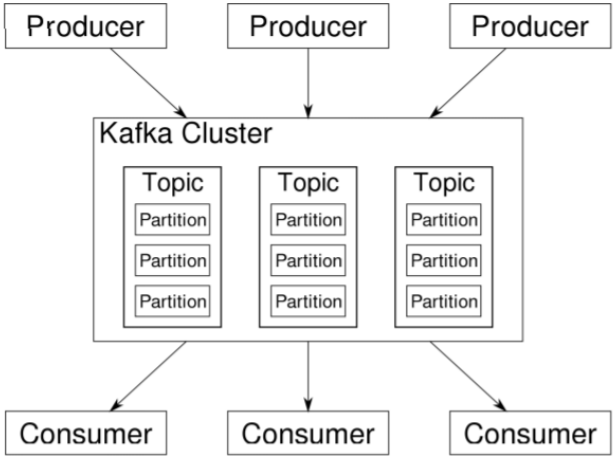
```
[cloudera@quickstart ~]$ sqoop import-all-tables \
    -m 1 \
    --connect jdbc:mysql://quickstart:3306/retail_db \
    --username=retail_dba \
    --password=cloudera \
    --compression-codec=snappy \
    --as-parquetfile \
    --warehouse-dir=/user/hive/warehouse \
    --hive-import
```

Collecte de la donnée : Kafka

• Permet de créer des pipelines de traitement de données en temps réel
• Modèle producteur-consommateur
• Traitements distribués
• Faible latence (batch, cache)

Cas d'usage:

- ✔ Message broker
- ✔ Website activity tracking
- ✔ Metrics
- ✔ Log Aggregation
- ✔ Stream Processing



Collecte de la donnée : Kafka

Topics: Catégorie des flux de messages gérés par Kafka

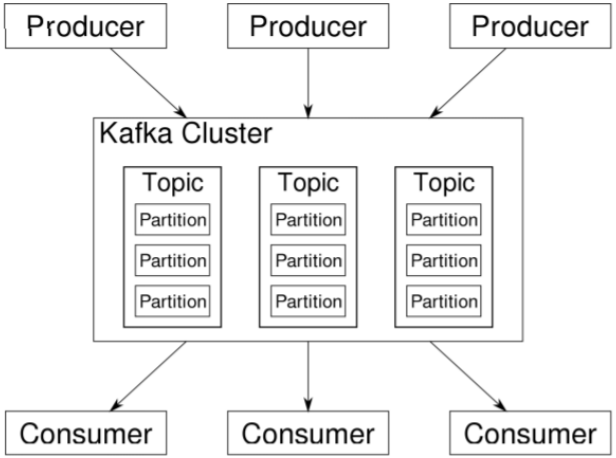
• Producers: Processus publiant les messages dans les topics

• Consumers: Processus souscrivant à des topics pour recevoir les données publiées

• Les topics sont constitués de partitions dont les messages sont ordonnés

• Rétention déterminée dans le temps

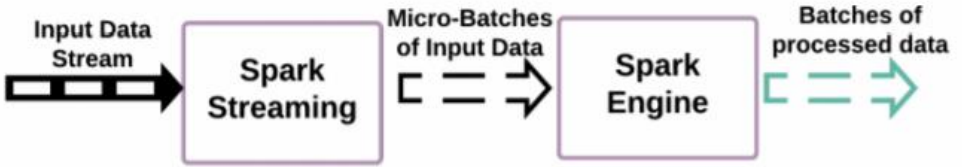
Tolérance à la panne: N réplicat = N-1 failures sans perte de messages



Collecte de la donnée et traitement: Apache streaming

- Les architectures Big Data possèdent souvent une couche batch et une couche temps réel
- Traitement en une seule passe, pas de stockage Intermédiaire

- Spark streaming
  - ✔ Fonctionne par mini batches
  - ✔ Compatibilité avec Spark
  - ✔ Tolérance aux pannes



Collecte de la donnée et traitement: Apache Flume

- Flume est un service distribué, fiable et disponible pour collecter, agréger et déplacer efficacement de grandes quantités de données de journal.
- Architecture simple et flexible basée sur des flux de données continus.

Collecte de la donnée et traitement: Apache Flume

- Robuste et tolérant aux pannes avec des mécanismes de fiabilité réglables et de nombreux mécanismes de basculement et de récupération.
- Modèle de données extensible simple qui permet une application analytique en ligne.
- Traitement en une seule passe, pas de stockage

ELK

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

ELK

- 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 …..)

ELK

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

Questions ?

Merci