



2018/2019

# RAPPELS SÉANCE PRÉCÉDENTE

# INFORMATIQUE DÉCISIONNELLE, BIG DATA, HADOOP

- Énormément de données sont accessibles aux entreprises et de la valeur peut être en tirée (SI existant, parcours clients et connaissance clients, log machines)
- Les approches traditionnelles rencontrent des limites :
  - SGBDR: attention aux données > plusieurs To, attentions aux données non structurées
  - Analytique l'approche verticale plafonne

- Besoin d'avoir des approches distribuées – Stockage / Analyse

# BIG DATA, HADOOP

- Hadoop est la première réponse accessible pour traiter de la Big Data
  - **HDFS** : Stockage massif distribué à faible coût
  - **MapReduce** : FrameWork d'analyse des données
  - **YARN** : Management des job et des ressources d'un cluster

- Et aussi :
  - Open Source
  - Ecosystème complet: manipulation et analyse des données, administration, interfaces, …

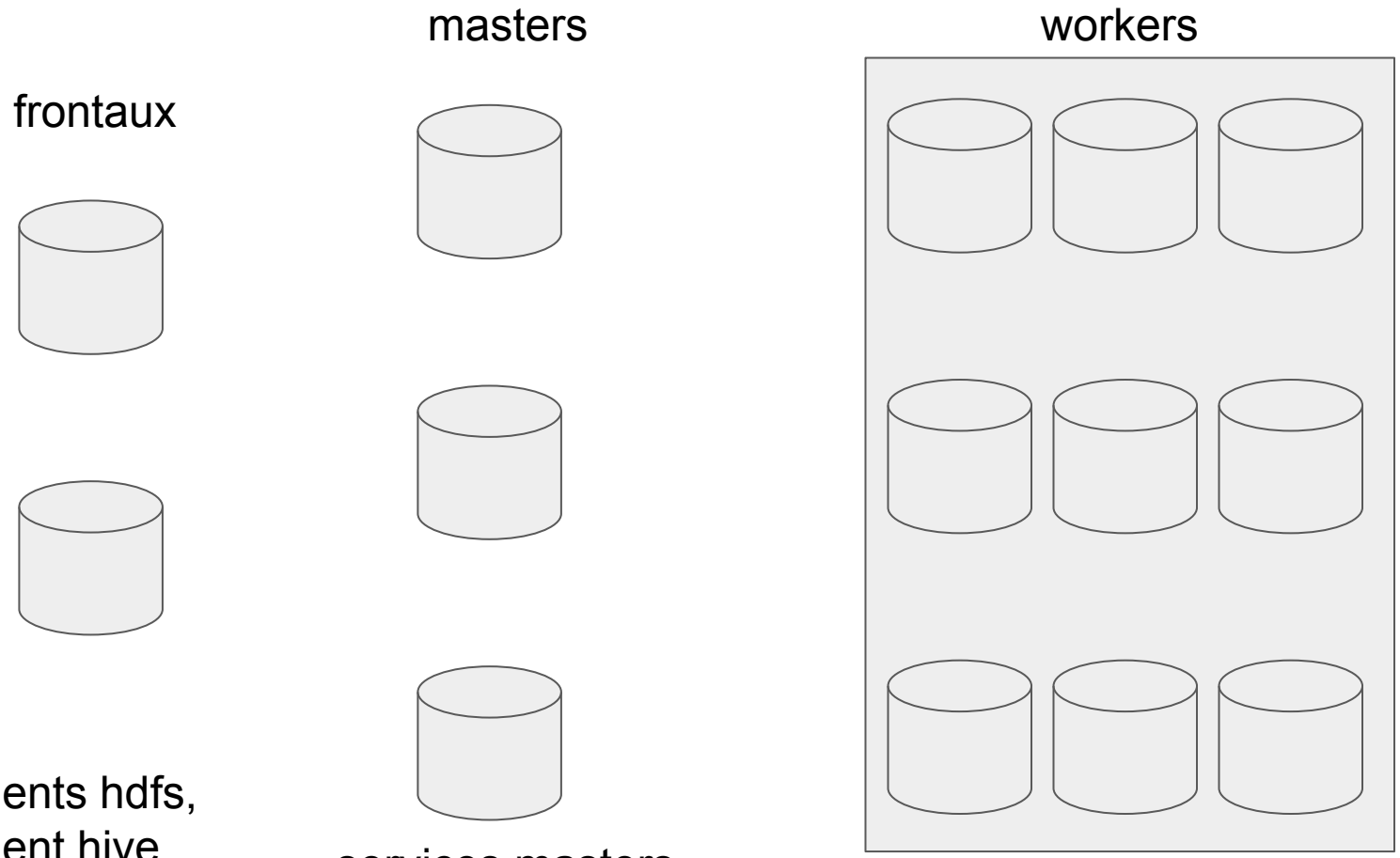
# BIG DATA, HADOOP
## *BIG DATA RAPPEL*

- Pourquoi utiliser Hadoop – mature pour :
  - Traitements **lourds en batch** : faire baisser en facteurs des traitements de plusieurs heures.
  - Stockage massif froid de tous types de données

- Donc bon pour :
  - Désengorgement d'entrepôt de données (DWH)
    - > Traitements ETL lourds (Filtre, consolidation, règles de gestions)
    - > Historisation
  - Création de datalake (stratégie de centralisation de toutes les sources pour un ou plusieurs besoin)

# BIG DATA, HADOOP

## *TYPOLOGIES DES NŒUDS DANS UN CLUSTER*




# LES DISTRIBUTIONS HADOOP

# QU'EST CE QU'UNE DISTRIBUTION HADOOP ?

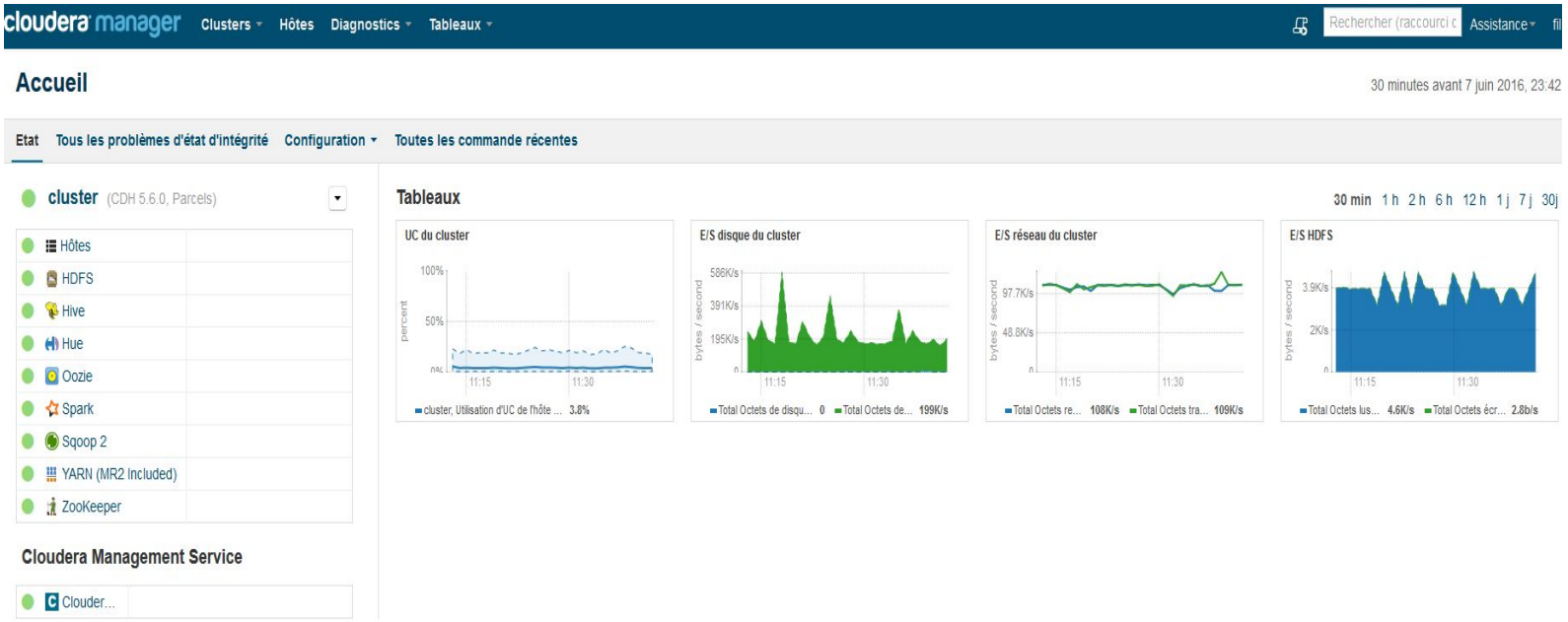
- Une distribution Hadoop est un ensemble de technologies de Hadoop, packagées par un éditeur
- Un package Hadoop contient des outils développés par l'entreprise qui simplifient l'installation et l'utilisation de Hadoop.

# Pourquoi une distribution ?

- Simplifie l'installation et la gestion des composants (managers)
  - Déploiement de Hadoop via une interface (web ou autre)
  - Monitoring des services et relance des services
- Package un ensemble de composants de Hadoop avec des outils facilitant leur utilisation (exemple : Ambari, Cloudera Hue)
- Assure la compatibilité entre les composants
- Support commercial
- VM Sandbox pour tester

# Déployer avec Ambari ou cloudera manager

- Permet d'ajouter des noeuds à un cluster à partir de l'adress ip
- Permet de répartir les services sur les noeuds
- Affiche la santé des noeuds



# Les principales distributions … avant 2019 !




- Fournissent une version packagée de Hadoop
- Développent des outils autour de Hadoop
- Proposent du conseil en entreprise
- Proposent un support technique
- Proposent des formations

# Cloudera


- Premier à proposer des VMs tests
- Employeur de Doug Cutting, le “père” de Hadoop

- Interface d’administration : cloudera manager
- Interface d’utilisation : HUE
- Alternative à MapReduce : Impala

# Hortonworks


- Technologie 100% open source
- Distribution la plus proche du Hadoop Open Source (nombreux contributeurs Hadoop chez Hortonworks)
- Organise les Hadoop Summit

- Interface d'administration : Ambari
- Interface d'utilisation : Ambari view (très jeune)
- Alternative à MapReduce : Tez

# MapR

- Hadoop modifié selon le besoin des entreprises
- Utilise mapR-FS à la place de HDFS (un noeud est à la fois master et slave)
- (d'après les retours d'expérience), couche de sécurité plus facile à mettre en place et à utiliser (Kerberos)

- Interface d'administration : MapR control System

# Services entreprise

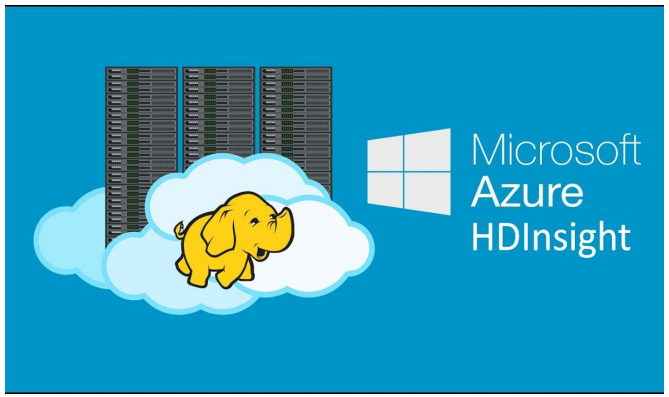



- Solution technique (package hadoop + outils adaptés à l'utilisation de Hadoop en entreprise )
- Support technique
- Conseil en entreprise
- Maintenance

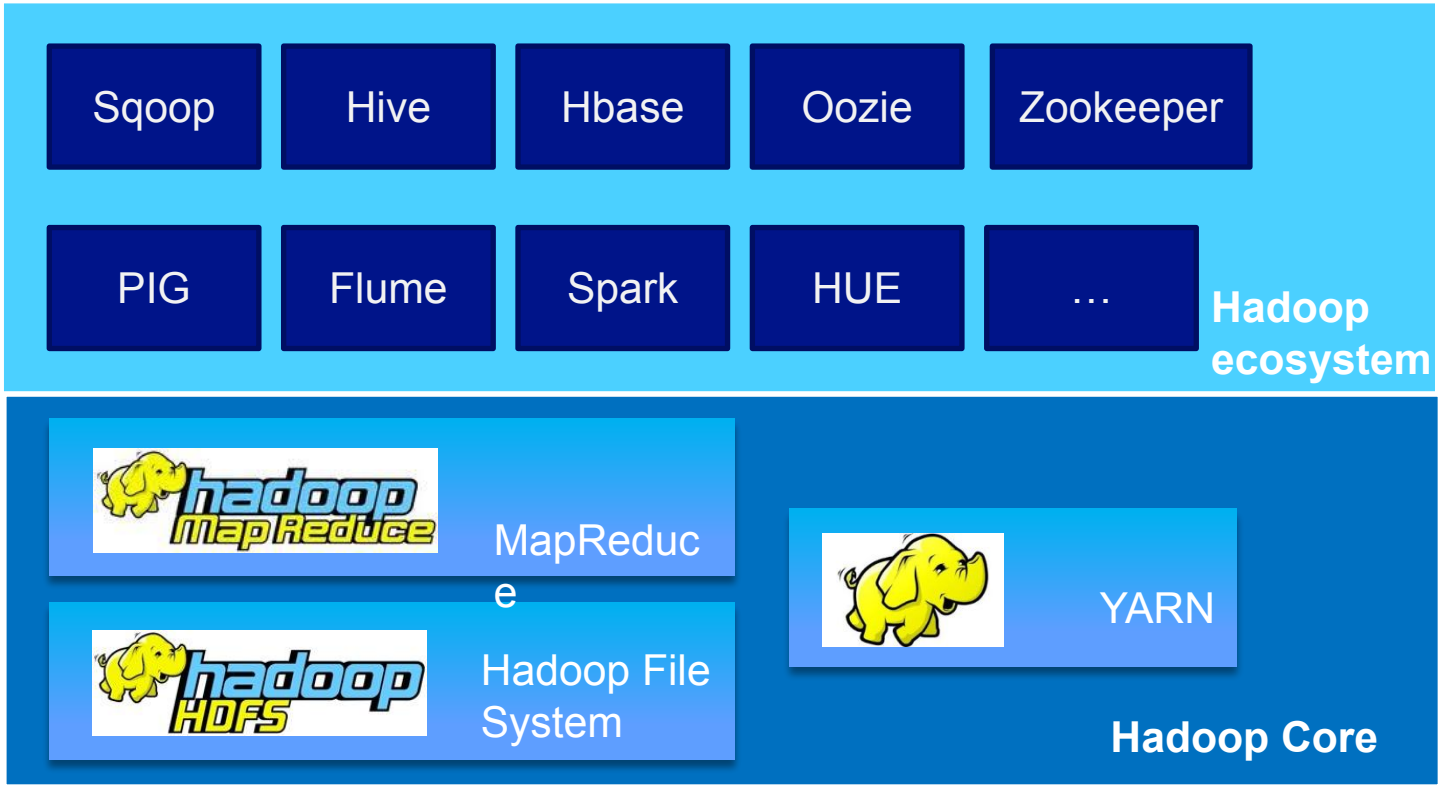
# Le Big Data en 2019

- Janvier 2019 : Fusion de Cloudera et HortonWorks
- Mai 2019: Annonce de fermeture de MapR
- Août 2019: Rachat de MapR par HPE (Hewlett Packard Enterprise)

Le cloud, le cloud, le cloud …

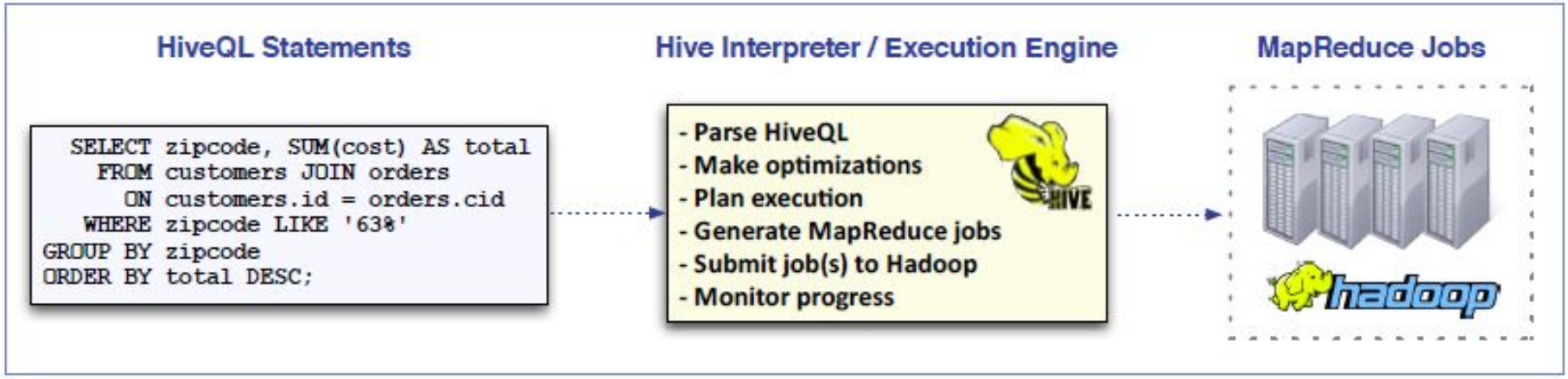

# ECOSYSTEME HADOOP



# HIVE

## Hive – Présentation

- Hive transforme des requêtes HiveQL (proche du SQL) en jobs d'un moteur de calcul (comme MapReduce, Tez ou Spark)
- Hive2 fonctionne en mode client serveur pour les requêtes. (utilisation de drivers JDBC et ODBC pour la compatibilité avec d'autres applications)
- Il s'occupe de lancer des jobs sur le cluster (interface avec YARN)
- Transactionnel possible (avec configuration)




*Ressource Cloudera*

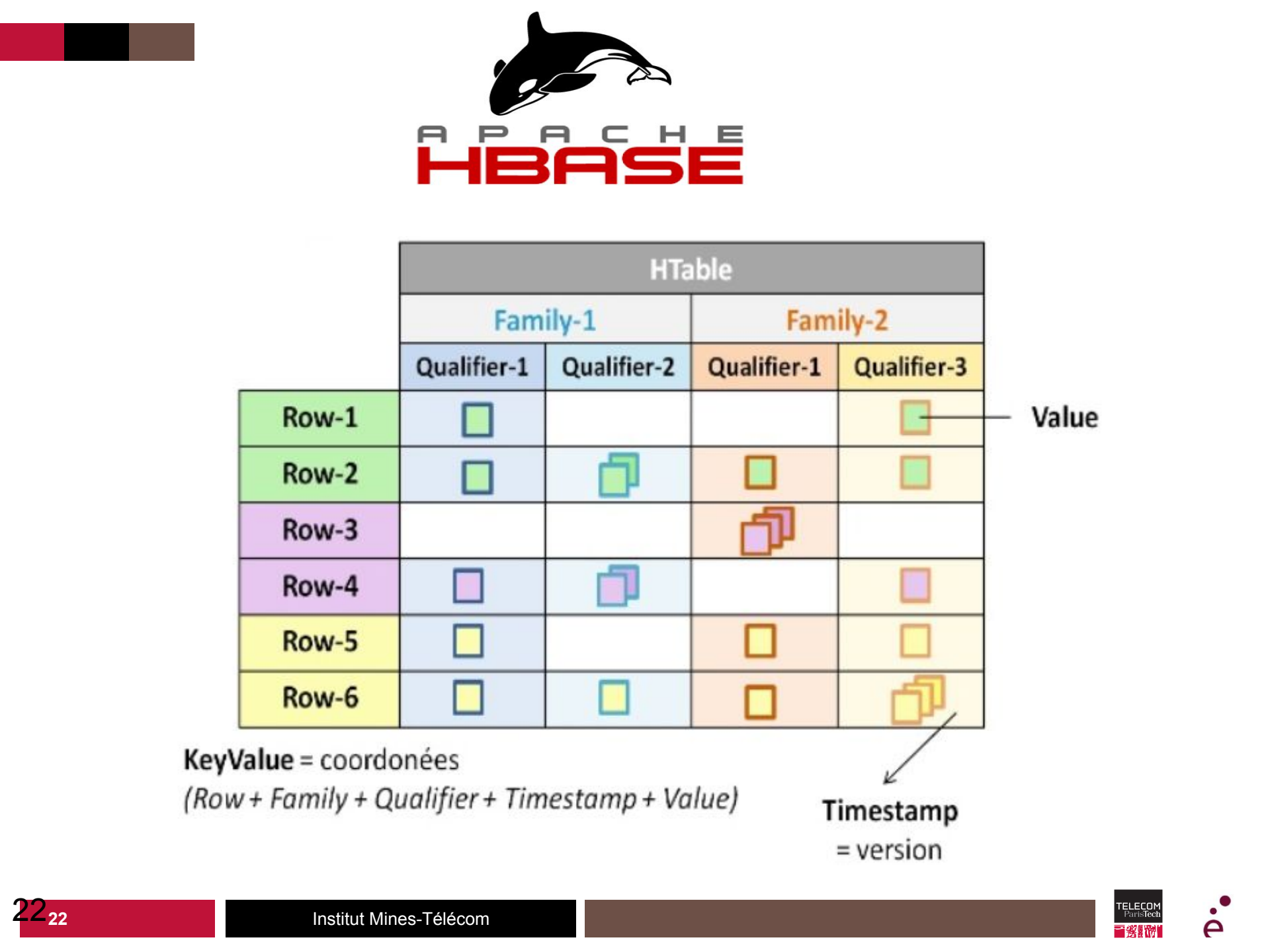
## Hive – Présentation
## Ses intérêts d'utilisation

- Il rend accessible l'utilisation de calculs distribués à un plus grand public
  - Pas de gestion de développement type logiciel à gérer
  - S'appuie sur une connaissance du SQL plus diffusée en entreprise
- Possibilité d'interagir avec d'autres outils via drivers (JDBC, ODBC)
- Très utilisé dans les entreprises

# ETL D'HADOOP : PIG

- ETL de Hadoop : permet de créer des chaînes de traitement dans Hadoop
- Langage de script pour hadoop, et Pig Latin (type requête SQL)
- Crée des jobs MapReduce (ou Tez, Impala...)
- De moins en moins utilisé, au profit de Spark ou Hive


- Hbase est une base de données orientée colonne.
- A partir d'un clé, stocke des valeurs dans des colonnes, elles mêmes dans des column families.
- Aucun modèle au préalable. On stocke ce qu'on veut, on ajoute des colonnes.
- Insérer une valeur dans une colonne déjà remplie pour cette rowkey remplacera sa valeur. Un historique peut être stocké et requêté.
- Très rapide pour récupérer les données associées à la rowkey.

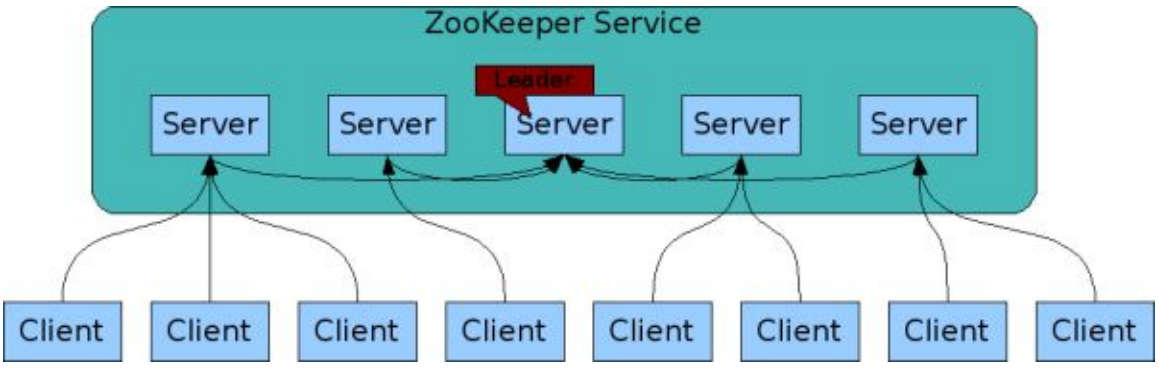

APACHE
HBASE
HTable
Family-1
Family-2
Qualifier-1
Qualifier-2
Qualifier-1
Qualifier-3
Row-1
Row-2
Row-3
Row-4
Row-5
Row-6
Value
KeyValue = coordonées
(Row + Family + Qualifier + Timestamp + Value)
Timestamp
= version

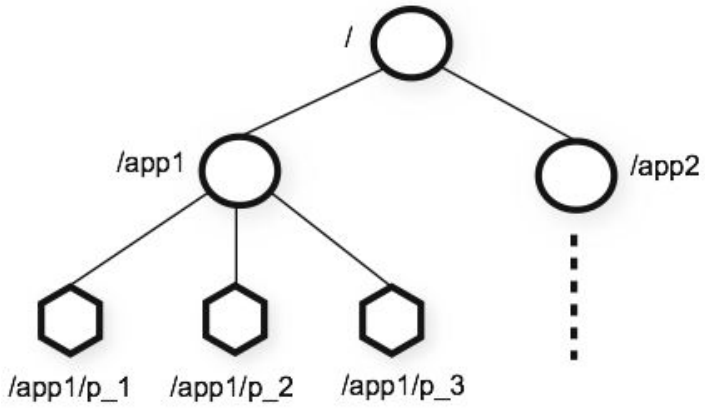

- Permet de gérer des jobs Hadoop récurrents ou ponctuels
- Intègre de nombreuses applications de l'écosystème Hadoop
  - Ex: Spark, Hive, Pig, …
- Permet de coordonner des jobs via des Coordinators (Lancements réguliers, selon une condition)
- Un workflow est un ensemble d'actions et de conditions organisées sous la forme d'un graphe orienté acyclique (DAG)
- Les actions (action nodes) peuvent être des jobs MapReduce ou Spark, des requêtes Hive, des scripts Pig, Java ou Shell, etc…)
- Les conditions (decision nodes) peuvent porter sur le bon déroulement des actions précédents, ou sur des métriques exportées.
- Les Workflow sont définis en XML.

# Un coordinateur de service distribué pour des applications distribuées


APACHE ZooKeeper™

- Leader Election (Namenode, HiveServer, Kafka, Oozie, …)
- Gestion de configuration (Tous (?) les services distribués de Hadoop)
- Simple, scalable, rapide, efficace



TELECOM ParisTech

INSTITUT Mines-Télécom

# FOCUS SUR HIVE

# HIVE

## Hive – Présentation

## Son fonctionnement

- Hive requête des tables comme un RDBMS
  - Une table est un dossier dans HDFS
  - Par défaut ces tables / dossiers se trouvent dans /user/hive/warehouse/table_name

- Hive gère la structure des tables (les schéma des fichiers) grâce à son metastore
  - Ces metadata sont générées lors de la création des tables
  - Le metastore est généralement une base tiers MySQL ou PostgreSQL que hive requête à la volée

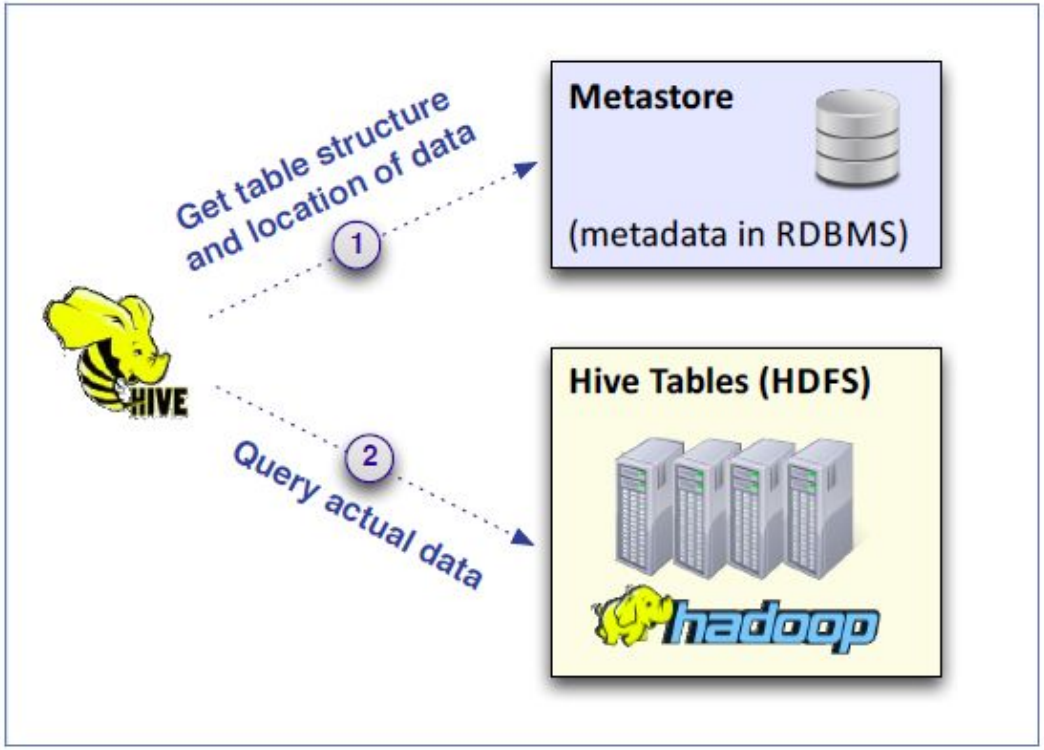
- Hive utilise un Compiler (nom du composant Hive) pour parser les requêtes ,construire un “logical plan” qui consiste en un arbre d’opérations (filter, join, …). Ce plan est ensuite optimisé et transformé en une série d’opération Map-Reduce.

- Hive utilise un Execution Engine (Tez, MapReduce, Spark) pour exécuter les opérations.

# Hive – Présentation

## Son fonctionnement

- Hive consulte son metastore avant l'exécution de ses requêtes (récupération / vérification du noms des colonnes, des types, …)




*Ressource Cloudera*

# Hive – Présentation
# Son fonctionnement

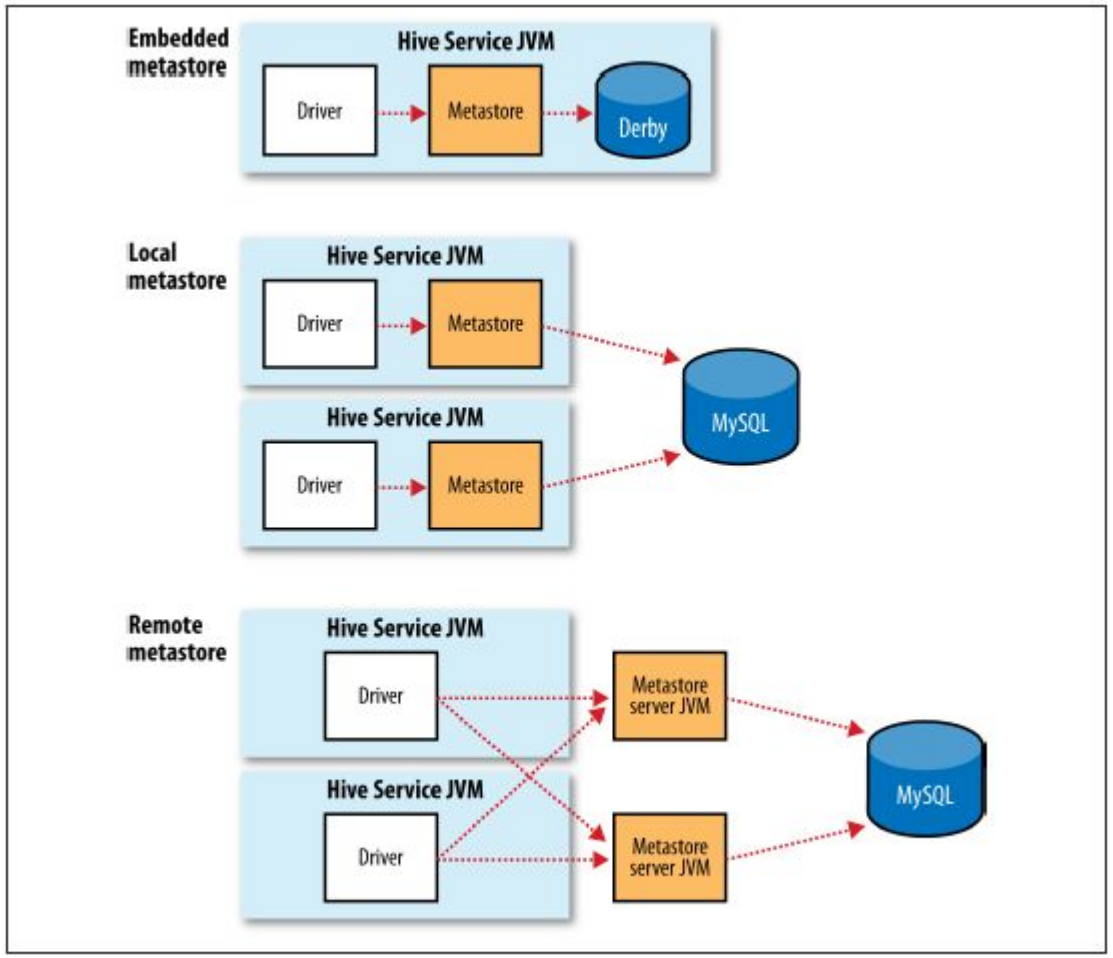



*Figure 17-2. Metastore configurations*

Embedded (Derby Database):

- Une seule session Hive peut accéder au Metastore, pas de concurrence possible.

Local Metastore:

- Une BDD externe, MySQL, PostgreSQL, … Accès concurrent possible

Remote Metastore:

- BDD externe et Metastore Service externe. Isolation des services / accès BDD, meilleure sécurité

# Hive – Présentation
## Son fonctionnement

Schema on Read:

Les données sont copiées avant que le schéma soit vérifié. C'est lors de l'exécution d'une requête que le schéma est vérifié.

- Initial load rapide (copie de fichier uniquement)
- Pas besoin de connaître le besoin au load (ingestion des données), donc plus flexible
- Plusieurs schémas peuvent être définis pour les mêmes données (tables externes)

# Hive – Présentation
## Son fonctionnement

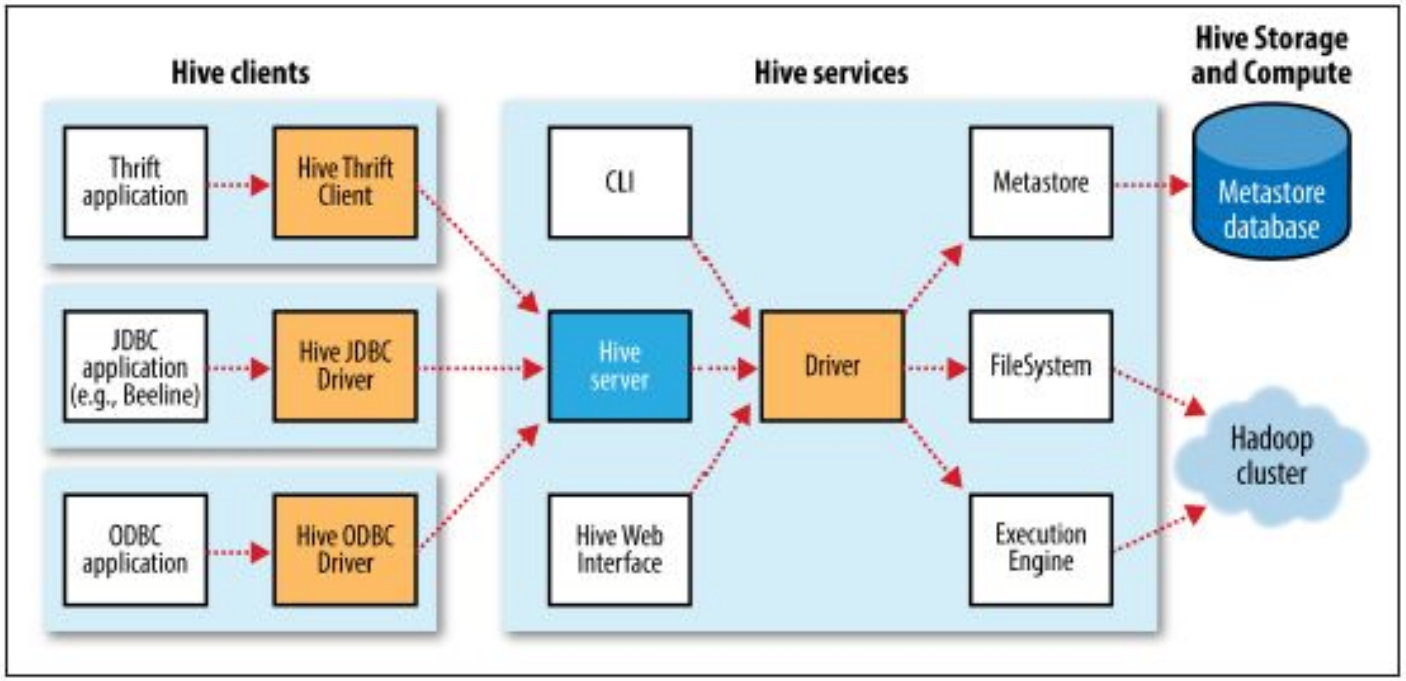
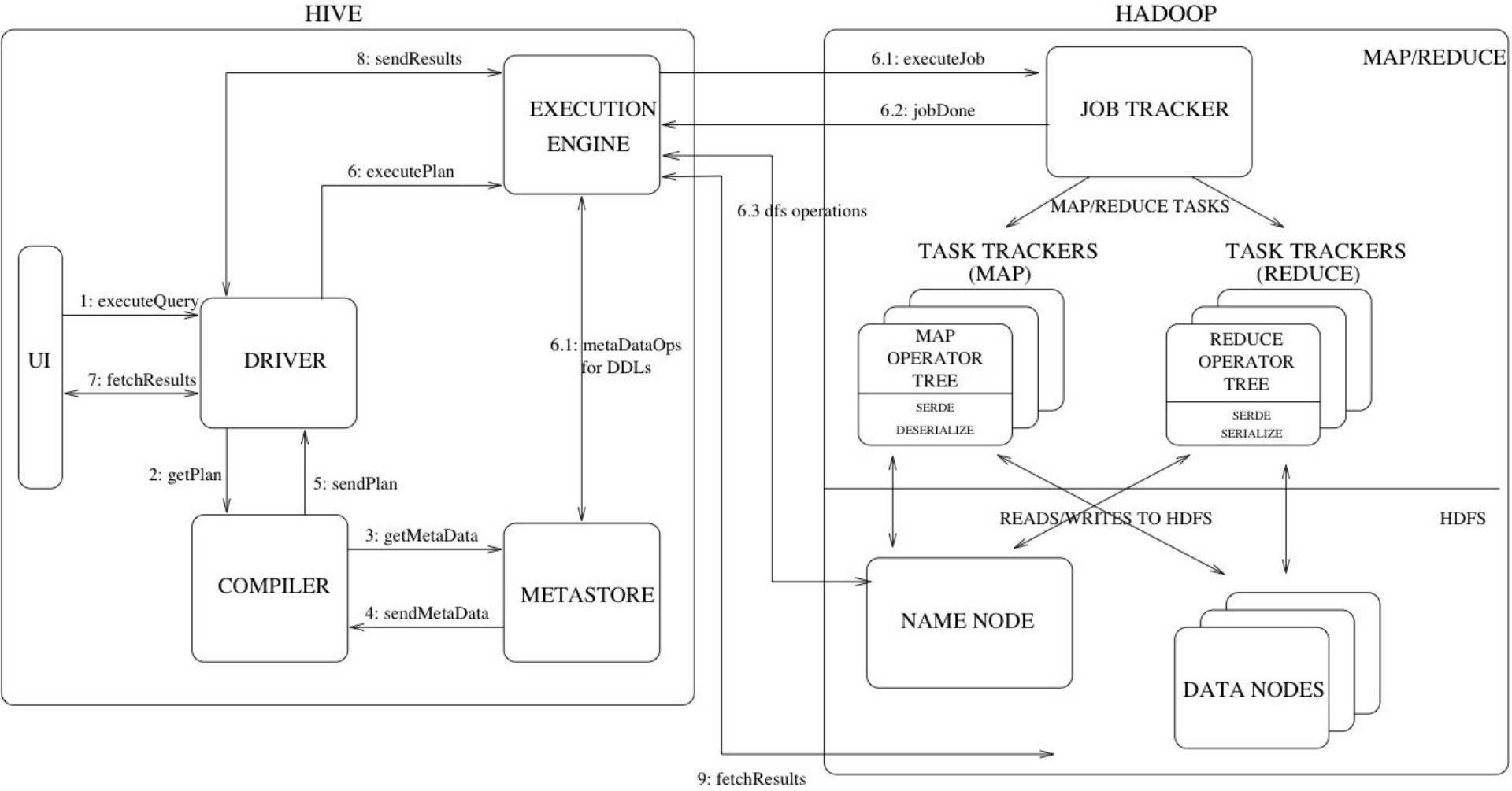



*Figure 17-1. Hive architecture*

# Hive Architecture



# HIVE

## Hive – Présentation

## Hive n'est pas un datawarehouse

- Hive peut simuler les fonctions d'un datawarehouse mais il ne remplace pas les RDBMS traditionnels, il les complète

| | Relational Database | Hive |
|---|---|---|
| **Query language** | SQL | HiveQL |
| **Update individual records** | Yes | No |
| **Delete individual records** | Yes | No |
| **Transactions** | Yes | No |
| **Index support** | Extensive | Limited |
| **Latency** | Very low | High |
| **Data size** | Terabytes | Petabytes |

*Ressource Cloudera*

## Hive – Présentation

## Hive shell

- On peut interagir avec hive via son cli le hive shell son fonctionnement et sa syntaxe est calqué sur MySQL

```
$ hive
hive> SELECT id_personnage, nom_personnage,
prenom_presonnage FROM t_personnage
WHERE cd_region = 'NORTH';

1000    Snow        John
1001    Stark       Sensa
1002    Stark       Robert

hive> quit;
```

## Hive – Utilisation

## Des databases et des tables

- Chaque table manipulée appartient à une database (celle par defaut et la base « default »
- On peut les visualiser avec SHOW DATABASES

```
hive> SHOW DATABASES;
default
George_R_R_Martin
```

- En créer une nouvelle avec CREATE DATABASE
  - Une nouvelle database correspond à un nouveau dossier dans HDFS
- On peut switcher de database avec l'instruction USE

```
hive> use George_R_R_Martin;
hive> select * from t_personnage
```

## Hive – Utilisation

## Des databases et des tables

- On peut afficher la liste des tables d'une database avec la commande SHOW TABLES

```
hive> use George_R_R_Martin;
hive> show tables;
t_personnage
t_royaumme
t_villes
```

## Hive – Utilisation

## Des databases et des tables

- On peut afficher la structure d'une table avec la commande DESCRIBE

```
hive> DESCRIBE George_R_R_Martin.t_personnage;

nom  string
prenom   string
dt_naissance timestamp
age  int
```

## Hive – Utilisation

## La syntaxe de Hive

- La syntaxe de Hive est un sous-ensemble de SQL-92 son fonctionnement est proche de MySQL
- Les requêtes sont séparées par des « ; »
- les commentaires se font avec « -- »

```
$ cat requete.hql

-- Winter is comming !
SELECT id_personnage, nom_personnage, prenom_presonnage
    FROM t_personnage
    WHERE cd_region = 'NORTH';
```

## Hive – Utilisation

## La syntaxe de Hive – SQL

- On sélectionne des champs d'une table grâce à SELECT

```
hive> SELECT id_personnage, nom_personnage, prenom_presonnage
    FROM t_personnage;
```

- On sélectionne tous les champs d'une table grâce à SELECT *

```
hive> SELECT * FROM t_personnage;
```

# HIVE

## Hive – Utilisation

## La syntaxe de Hive – SQL

- Hive support la clause LIMIT pour plafonner le nombre de lignes à remonter

```
hive> SELECT * FROM t_personnage LIMIT 10;
```

- On peut aussi trier les résultats avec ORDER BY pour garantir un TOP

```
hive> SELECT * FROM t_personnage ORDER BY age LIMIT 10;
```

## Hive – Utilisation

## La syntaxe de Hive – SQL

- La clause **WHERE** est là pour filtrer les données

```
hive> SELECT * FROM t_personnage WHERE age > 10;
```

- Hive supporte aussi la clause **IN** dans le **WHERE**

```
hive> SELECT * FROM t_personnage WHERE age IN (10, 19);
```

- Ainsi que les clauses **OR** et **AND**

```
hive> SELECT * FROM t_personnage WHERE age = 10 AND prenom =
‘Luc’  ;
```

## Hive – Utilisation

## La syntaxe de Hive – SQL

- Hive supporte l'utilisation d'alias – utile pour la lisibilité des requêtes complexes

```
hive> SELECT p.nom, v.taille_ville FROM t_personnage p
INNER JOIN t_ville v ON p.ville = v_ville
WHERE p.age > 10;
```

- Attention le **AS** ne marche pas

## Hive – Utilisation

## La syntaxe de Hive – SQL

- Hive supporte l'utilisation d'UNION / UNION ALL

```
SELECT p.nom, v.taille_ville FROM t_personnage p
WHERE p.age > 10;
UNION
SELECT p.nom, v.taille_ville FROM t_personnage p
WHERE p.age < 10;
```

## Hive – Utilisation

## La syntaxe de Hive – SQL

- Hive supporte l'utilisation de sous-requêtes

```
SELECT p.nom, p.prenom FROM
   ( SELECT nom, prenom FROM t_personnage WHERE age = 10
LIMIT 20) p ;
```

- Toutes les sous-requêtes doivent être nommées.

## Hive – Utilisation

## La syntaxe de Hive – Les data types

- Hive supporte les types classiques des bases de données mais leur nom s'inspirent plutôt du java que des RDBMS
- Entier :
  - TINYINT (- 128 à 127) , SMALLINT (-32 768 à 32 767), INT (-2,147,483,648 à 2,147,483,647), BIGINT (-2^63 à -2^63 -1)
- Décimaux :
  - FLOAT , DOUBLE (beaucoup plus précis)
- Les autres types :
  - STRING, BOOLEAN (TRUE / FALSE), TIMESTAMP, …

## Hive – Utilisation

## La syntaxe de Hive – Création d'une table

- Requête basique - création d'une table

```
CREATE TABLE tablename (colname DATATYPE, ...)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY char
STORED AS {TEXTFILE|SEQUENCEFILE|RCFILE}
```

- Cela crée un répertoire dans /user/hive/warehouse

# HIVE

## Hive – Utilisation

## La syntaxe de Hive – Création d'une table

- Exemple pour créer la table iris

```
CREATE TABLE iris(
 sepal_length float,
 sepal_width float,
 petal_length float,
 petal_width float,
 species string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY char
STORED AS TEXTFILE;
```

- Cela crée un répertoire dans /user/hive/warehouse

## Hive – Utilisation

## La syntaxe de Hive – Chargement d'une table

- Pour charger la table il suffit de déplacer le fichier de données dans le répertoire de la table – exemple pour iris

```
hadoop fs –cp pig/iris.csv /apps/hive/warehouse/iris
```

- On peut aussi utiliser la commande LOAD dans Hive, attention cela déplace les données

```
hive> LOAD DATA INPATH 'pig/iris.csv' INTO TABLE iris;
```

## Hive – Utilisation

## La syntaxe de Hive – Chargement d'une table – external

- La technique vue plus tôt nécessite de déplacer les données il est possible de créer une table en lui spécifiant un répertoire déjà existant sur HDFS

```
CREATE EXTERNAL TABLE iris(
 sepal_length float,
 sepal_width float,
 petal_length float,
 petal_width float,
 species string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY "\t"
LOCATION '/user/cloudera/pig/;
```

## Hive – Utilisation

## La syntaxe de Hive – Chargement d'une table – external

- Une autre technique pour créer et loader une table de données et le CREATE TABLE … AS SELECT. Cette technique permet de matérialiser le résultat d'une requête select.

```
CREATE TABLE iris_v2 AS
SELECT  sepal_length,
        sepal_width,
        petal_length * petal_width,
        species from iris);
```

## Hive – Utilisation

## La syntaxe de Hive – Drop d'une table

- On utilse DROP pour supprimer une table

```
DROP TABLE iris;
```

- Si la table n'est pas en external les données sont supprimées.

## Hive – Utilisation

## Les fonctions d'agrégation

- Hive gère comme en SQL les fonctions d'agrégations classiques :
  - COUNT(), SUM(), MAX(), MIN(), AVG()
- Elles s'utilisent généralement avec une clause GROUP BY pour délimiter la clef l'agrégation

```
-- Moyenne des ages / nom de famille
SELECT nom, AVG(age) FROM t_personnage
GROUP BY nom;
```

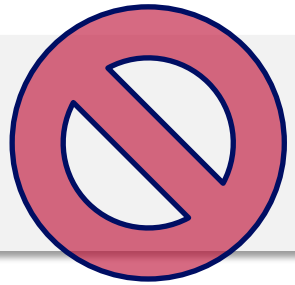
```
-- Nombre des personnages / nom de famille
SELECT nom, count(*) FROM t_personnage
GROUP BY nom;
```

## Hive – Utilisation

## Filtrer des résultats d'agrégations

- Il n'est pas permis d'utiliser un WHERE sur le résultat d'une fonction d'agrégation directement

```
-- Moyenne des ages / nom de famille
SELECT nom, AVG(age) FROM t_personnage
GROUP BY nom
WHERE AVG(age) > 5;
```



## Hive – Utilisation

## Filtrer des résultats d'agrégations

- Deux solutions : soit passer par une sous-requête :

```
-- Moyenne des ages / nom de famille filtré
SELECT a.nom, a.average FROM (
SELECT nom, AVG(age) average FROM t_personnage
GROUP BY nom
) a
WHERE average > 5;
```

## Hive – Utilisation

## Filtrer des résultats d'agrégations

- Deux solutions : sinon plus court utiliser HAVING:

```
-- Moyenne des ages / nom de famille filtré
SELECT nom, AVG(age) average FROM t_personnage
GROUP BY nom
HAVING AVG(age) > 5;
```

## Hive – Utilisation

## La syntaxe de Hive – Jointures

- Hive supporte la jointure de plusieurs tables avec la syntaxe suivante :

```
SELECT p.nom, v.taille_ville
FROM t_personnage p
INNER JOIN t_ville v ON p.ville = v.ville
```

- On peut aussi faire une LEFT ou RIGHT JOIN
- Attention la syntaxe plus ancienne de jointure comme celle-ci n'est pas supportée

```
SELECT p.nom, v.taille_ville
FROM t_personnage p, t_ville
WHERE p.ville = v.ville
```