esprit
Ecole Supérieure Privée
d'Ingénierie et de Technologies

Chp5 – Langages de requête Hadoop- Hive

Mme Ines Slimene
ines.slimen@esprit.tn

Plan module

- Introduction
- Écosystème Hadoop
- HDFS
- MapReduce
- **Langages de requête Hadoop :Hive, Pig**
- SGBDNR
  - Différences entre une BDNR et une BD relationnelle
  - Typologies des BD non relationnelles
- Etude d'un SGBDNR : HBase

Plan

- Introduction
- Pig, Hive –Similarités
- Pig, Hive –Différences
- Hive
  - Cas d'utilisation
  - Concepts
  - Metastore
  - Commandes

Introduction

- Pour éviter les complexités du modèle de programmation Hadoop, quelques langages de développent d'application ont été intégrés a Hadoop.
  - Pig 
  - Hive 

Pig, Hive –Similarités

- Réduire la taille des programmes java
- Les applications sont traduites en programmes MapReduce en arrière plan.
- Fonctionnalité extensible
- Interopérabilité avec les autres langages
- Non conçu pour les lecture/ écriture aléatoire ou les requêtes de faible latence.

Pig, Hive –Différences

Caractéristiques	Pig	Hive
Développé par	Yahoo!	Facebook
Langage	Pig latin	HiveQL
Type de langage	Data flow	SQL
Structure de données supportée	Complex	Structuré
Schema	optionnel	Obligatoire

Hive

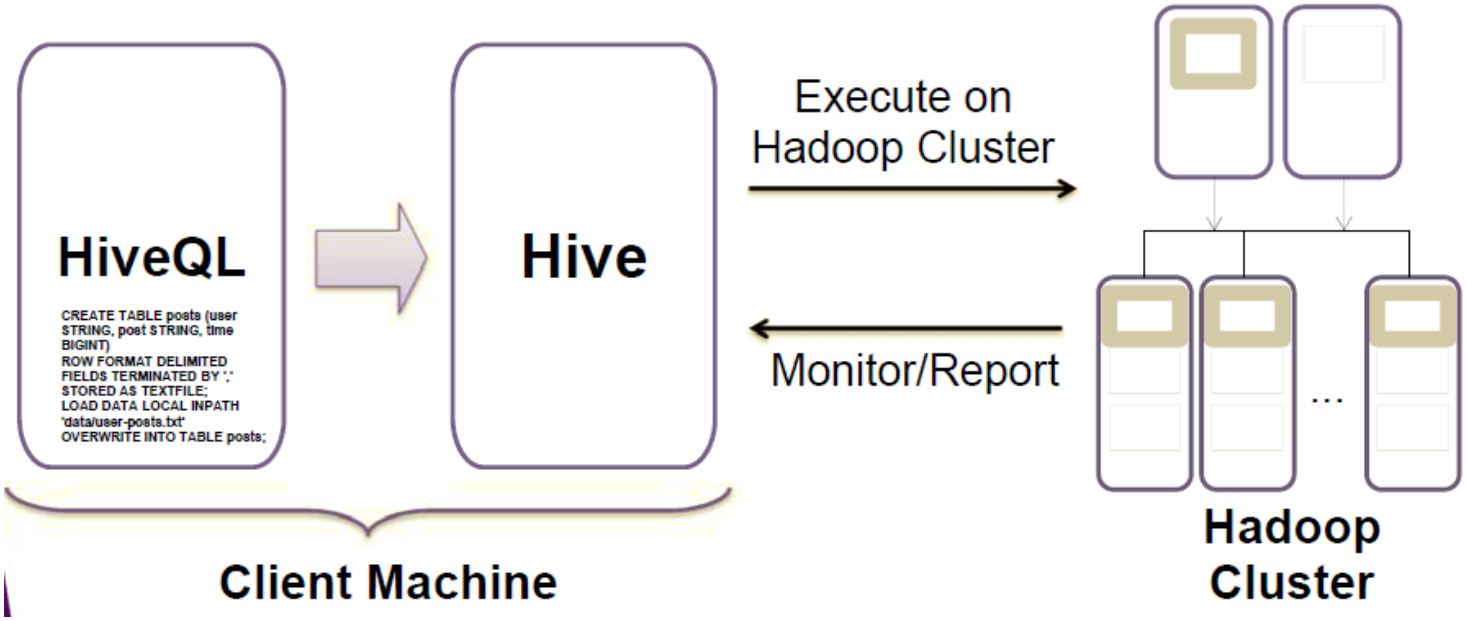
- Solution Data Warehouse intégrée dans Hadoop
- Fournit un langage de requête similaire au SQL nommé HiveQL
  - nécessite un apprentissage minimale pour les personnes ayant une expertise SQL
  - public cible : analystes de données
- Les travaux de développement Hive ont commencé sur Facebook en 2007

Offres de Hive

- Capacité de structuration des différents formats de données
- Interface simple pour l'analyse et la synthèse de grandes quantités de données
- L'accès aux fichiers sur les différents supports de stockage de données tels que HDFS et HBase

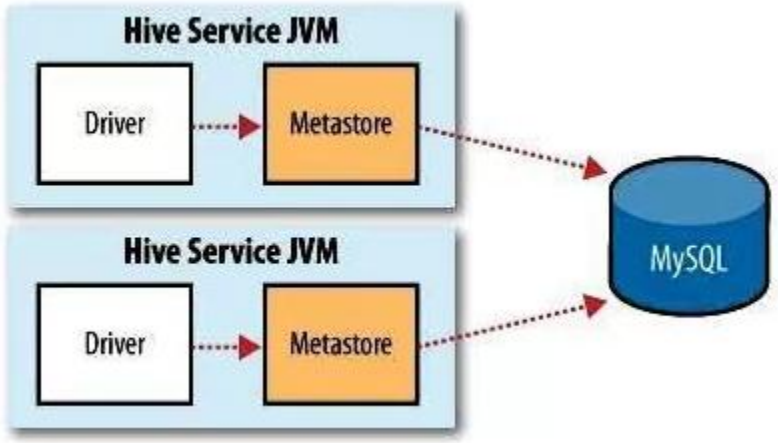
Hive

- Traduit les requêtes HiveQL en un ensemble de jobs MapReduce qui seront exécutés dans un cluster Hadoop.



Hive : Metastore

- mysql -u root –p
- mysql> show databases;
- mysql> show tables;
- mysql> select TBL_NAME from TBLS;



Hive : Concepts

- Inspiré des bases de données relationnelles
  - **Base de données**: ensemble de table
  - **Table**: ensemble de lignes qui ont le même schéma (même nombre de colonnes)
  - **Ligne**: un enregistrement, ensemble de colonnes
  - **Colonne**: contient la valeur et le type d'un champs

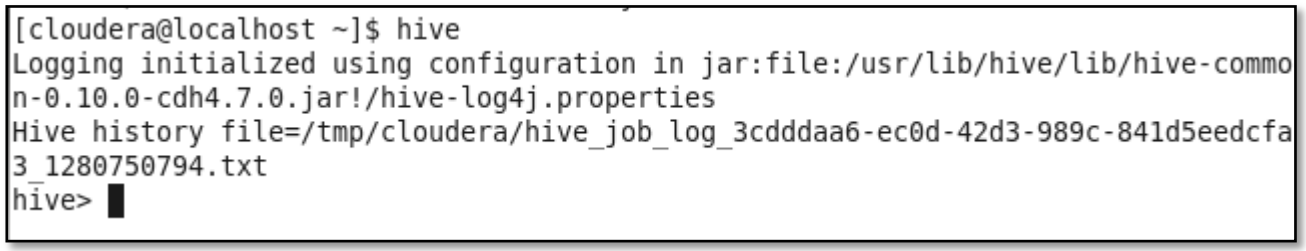
Hive : Concepts

Les types de données Hive :

- TINYINT, SMALLINT, INT, BIGINT
- BOOLEAN
- FLOAT
- DOUBLE
- STRING
- BINARY
- TIMESTAMP
- DECIMAL
- …

Création de base de données

- Démarrage Hive :
  - hive



- Création base de données :

hive> create database test;

hive> use test;

Création de table

- Création de table pour le stockage des données qui existent dans le fichier /ch5_data/user-posts.txt

```
hive> CREATE TABLE posts (user STRING, post STRING, time BIGINT)
        ROW FORMAT DELIMITED
        FIELDS TERMINATED BY ','
        STORED AS TEXTFILE;
```

- hive> show tables; (Afficher la liste des tables)
- hive> describe posts; (Description de la table posts)

Insertion données dans une table

- hive> LOAD DATA LOCAL INPATH '/home/cloudera/ch5_data/hive/user-posts.txt'
  OVERWRITE INTO TABLE posts;
- hadoop fs -ls /user/hive/warehouse

Affichage des données

- hive> select count (1) from posts;
- hive> select * from posts where user="user2";
- hive> select * from posts where time<=1343182133839 limit 2;

Suppression d'une table

- hive> DROP TABLE posts;
- hive> exit;
- hadoop fs -ls /user/hive/warehouse

External table

hive> CREATE **EXTERNAL** TABLE posts
(user STRING, post STRING, time BIGINT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION '/user/cloudera/test/';

- Hive charge les fichiers dans le répertoire **/user/cloudera/test/** et non pas dans le datawarhouse hive.

Partitions

- Pour augmenter sa performance, Hive a la possibilité de diviser les données (en partition).
  - Les valeurs de la colonne partitionnée divisent une table en segments
  - Semblable à des index de bases de données relationnelles
- Les partitions doivent être correctement emballées par les utilisateurs
  - Lors de l'insertion de données on doit spécifier une partition
- Au moment de la requête, Hive filtrera automatiquement les partitions.

Création de table partitionnée

```
hive> CREATE TABLE posts (user STRING, post STRING, time BIGINT)
PARTITIONED BY(country STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
hive> describe posts;
```

Chargement de données dans des tables partitionnées

- hive> LOAD DATA LOCAL INPATH '/home/cloudera/ch5_data/hive/user-posts-US.txt'

OVERWRITE INTO TABLE posts

PARTITION(country='US');

- hive> LOAD DATA LOCAL INPATH '/home/cloudera/ch5_data/hive/user-posts-AUSTRALIA.txt'

OVERWRITE INTO TABLE posts

PARTITION(country='AUSTRALIA');

Table partitionnée

- Les partitions sont physiquement stockés dans des répertoires distincts
- hive> show partitions posts;
- $ hadoop fs –ls -R /user/hive/warehouse/posts

/user/hive/warehouse/posts/**country=AUSTRALIA**

/user/hive/warehouse/posts/country=AUSTRALIA/user-posts-AUSTRALIA.txt

/user/hive/warehouse/posts/**country=US**

/user/hive/warehouse/posts/country=US/user-posts-US.txt

Jointure

- Soit les 2 tables suivantes : posts et likes
- hive> select * from posts limit 10;
- hive> select * from likes limit 10;
- hive> CREATE TABLE posts_likes (user STRING, post STRING, likes_count INT);

Jointure

- hive> INSERT OVERWRITE TABLE posts_likes

SELECT p.user, p.post, l.nblike

FROM posts p JOIN likes l ON (p.user = l.user);

- hive> select * from posts_likes limit 10;