Apache Hive

[Introduction 2](#_Toc100847700)

[1. C’est quoi Apache Hive 2](#_Toc100847701)

[2. Apache Hive: un peu d’histoire 2](#_Toc100847702)

[3. Terminologie 2](#_Toc100847703)

[a) Metastore: 2](#_Toc100847704)

[b) Table : 2](#_Toc100847705)

[c) Partitions: 3](#_Toc100847706)

[d) Bucket/Cluster: 3](#_Toc100847707)

[4. Types de données 3](#_Toc100847708)

[a) Primitifs : 3](#_Toc100847709)

[b) Complexes : 3](#_Toc100847710)

[5. HiveQL, SQL pour Hive 3](#_Toc100847711)

[• DDL (Langage de Définition de Données) : 3](#_Toc100847712)

[• DML (Langage de Manipulation de Données) : 3](#_Toc100847713)

[• Fonction utilisateurs (UDF : User Defined Function) : 3](#_Toc100847714)

[6. Table interne vs table externe : 4](#_Toc100847715)

[a) Les tables internes : 4](#_Toc100847716)

[b) Tables externes : 4](#_Toc100847717)

[7. Hive: partitionnement des tables : 4](#_Toc100847718)

[a) Création de tables : 4](#_Toc100847719)

[b) Ajouter une partition : 5](#_Toc100847720)

[c) Renommer une partition : 5](#_Toc100847721)

[d) Supprimer une partition : 5](#_Toc100847722)

[Avantage du partitionnement : 5](#_Toc100847723)

[Inconvenient du partitionnement: 5](#_Toc100847724)

[8. Hive: Bucketing c’est quoi ? 5](#_Toc100847725)

[a) La répartition des données sur les bucket: 5](#_Toc100847726)

[b) Création de tables buckettées : 6](#_Toc100847727)

[c) Arborescence HDFS d’une table buckettée : 6](#_Toc100847728)

[9. Apache Hive: Architecture 7](#_Toc100847729)

[10. Formats de fichier supportés par Hive ? 8](#_Toc100847730)

# Introduction

# C’est quoi Apache Hive

* C’est un outil de datawarehousing dans HDFS
* Couche intermédiaire entre Mapreduce et SQL
* Cela consiste à écrire des requêtes SQL et Hive génère le job MapReduce ou Tez
* Conçu pour les utilisateurs SQL qui ne connaissent pas de langage de programmation
* En mode batch

# Apache Hive: un peu d’histoire

* Développé par Facebook pour des usages internes
* C’est après qu’il est devenu un projet Apache Open source

# Terminologie

### Metastore:

* Chargé du stockage des métadonnées pour chaque table
* Il enregistre par exemple les schémas, les localisations et les métadonnées de partitionnement
* Pour que le driver suive la distribution des bases de données au sein du cluster.
* En pratique, les données sont stockées à la manière d'un SGBDR relationnel

### Table :

* Les tables sont complètement abstraites.
* En réalité c’est sous forme de fichier que tout est stocké.

### Partitions:

* Ce sont des répertoires issus d’une division de la table initiale contenant les données
* la division se fait en fonction d’une colonne de la table
* Et le titre des partitions correspond à la valeur de la colonne clé choisie
* La clé de partitionnement doit être à faible cardinalité (exemple : année, ville, sexe, …)
* Ce sont les colonnes qui se trouvent souvent dans la clause WHERE des requêtes SQL

### Bucket/Cluster:

* Ce sont les des sous-répertoires d’une partition (ou issue d’une table)
* Les données des partitions ou de la table sont réparties sur x buckets sur une colonne clé
* La clé d’échantillonnage doit être une colonne de grosse cardinalité presque unique tel que les identifiant

# Types de données

Il existe plusieurs types de données qui peuvent être stockées dans les tables et Hive supporte ces différents types de fichier

### Primitifs :

* Ce sont les types simples comme le string, float, int, booléen, ...

### Complexes :

* Ce sont les types qui prennent en charge les listes, les collections ou les structures

# HiveQL, SQL pour Hive

HiveQL supporte les langages et les fonctions suivants (Attention PAS de delete et uptade) :

### DDL (Langage de Définition de Données) :

Create, alter, drop

### DML (Langage de Manipulation de Données) :

Load, insert, select

### Fonction utilisateurs (UDF : User Defined Function) :

En Hive, on peut créer des fonctions (d’agrégation)

# Table interne vs table externe :

C’est un choix qui affecte la façon dont les données sont chargées, contrôlées et gérées.

### Les tables internes :

Pour les tables internes, Hive déplace les données dans le répertoire de son entrepôt.

* Si la table est supprimée, les métadonnées de la table et les données seront supprimées.
* Les données sont temporaires
* Hive gère complètement le cycle de vie de la table et des données.

### Tables externes :

Pour les tables externes, Hive ne déplace pas les données dans son répertoire d’entrepôt.

* Mappage concret entre données existant et les métadonnées HDFS
* Si la table externe est supprimée, les métadonnées de la table sont supprimées mais PAS les données.
* Les données sont également utilisées en dehors de Hive. Par exemple, les fichiers de données sont lus et traités par un programme existant qui ne verrouille pas les fichiers.
* Les données restent à leur emplacement même après un DROP TABLE.

# Hive: partitionnement des tables :

C’est le fait de répartir les données en fonction d’une clé de faible cardinalité

### Création de tables :

* **Syntaxe :**

Hive> Create [temporary] [external] table [if not exists] [db\_name.] table\_name

[(col\_name data\_type [COMMENT col\_comment], ...)]

[COMMENT table\_comment]

[ROW FORMAT row\_format]

[STORED AS file\_format]

* **Attention**

Ne pas oublier de préciser la nature de la table lors de sa création (par exemple : create EXTERNAL table)

### Ajouter une partition :

* **Syntaxe :**

Hive> ALTER TABLE table\_name

ADD PARTITION (column\_name culmn\_type);

LOCALISATION ‘hdfs\_directory\_path’

* **Remarque**

on créer une table partitionnée : create table … Partitioned by (column\_name culmn\_type);

### Renommer une partition :

hive> ALTER TABLE partition\_test PARTITION (year='2016',day='1') RENAME TO PARTITION (year='2016',day='01');

### Supprimer une partition :

Hive> ALTER TABLE logs DROP IF EXISTS PARTITION (year=2012,month=12,day=18);

## Avantage du partitionnement :

* Distribue la charge horizontalement
* Accélère les requêtes sur des volumes faibles de données

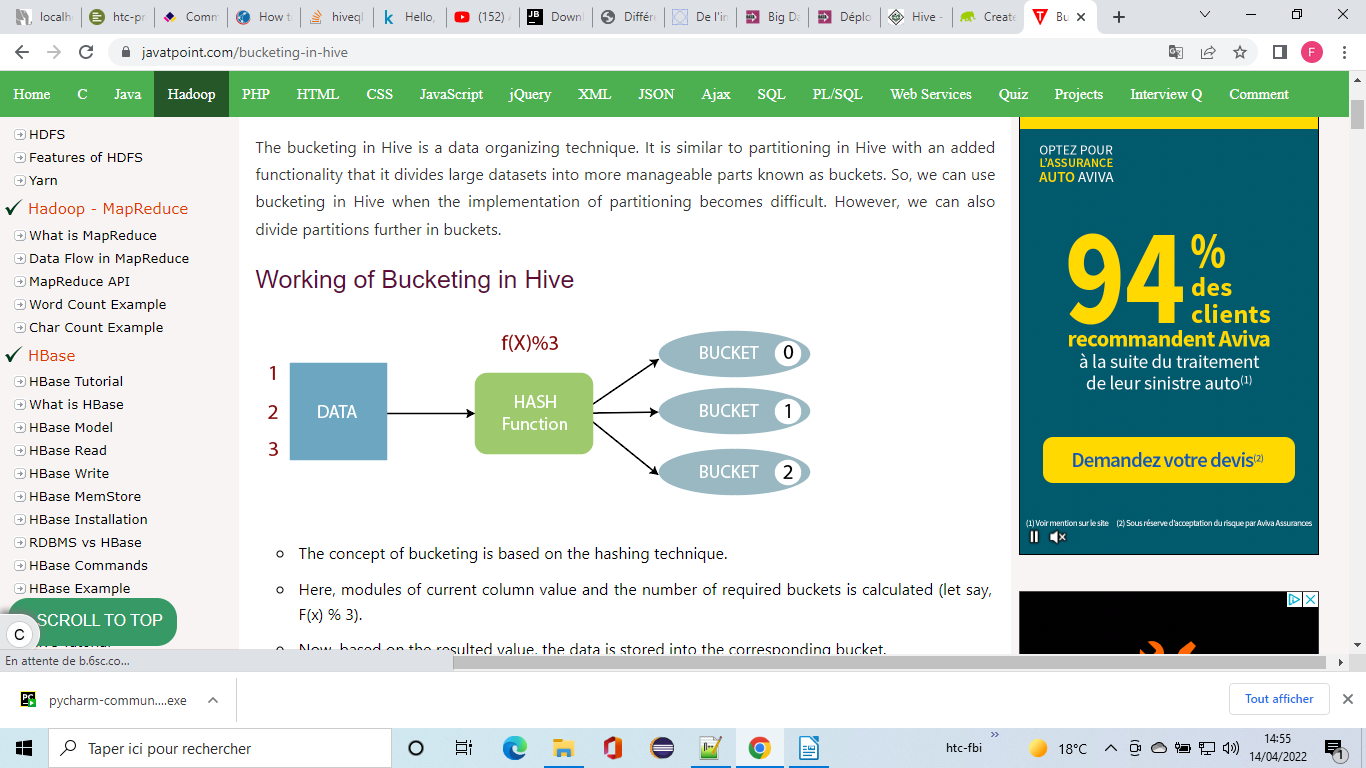
## Inconvenient du partitionnement:

* La clause GROUP BY peut être lent

# Hive: Bucketing c’est quoi ?

Si les données dans les partitions ou dans la table est encore trop volumineux, il est nécessaire de les divisé en plusieurs buckets ou sous-ensembles par échantillonnage.

### La répartition des données sur les bucket:

* Pour le bucketting il faut choisir la clé d’échantillonnage
* L’identification des buckets se fait par une fonction de hashage sur une colonne % (modulo) le nombre de buckets qui doit être spécifié
* Les types ayant le même hash atterrissent dans le même bucket

### 

### Création de tables buckettées :

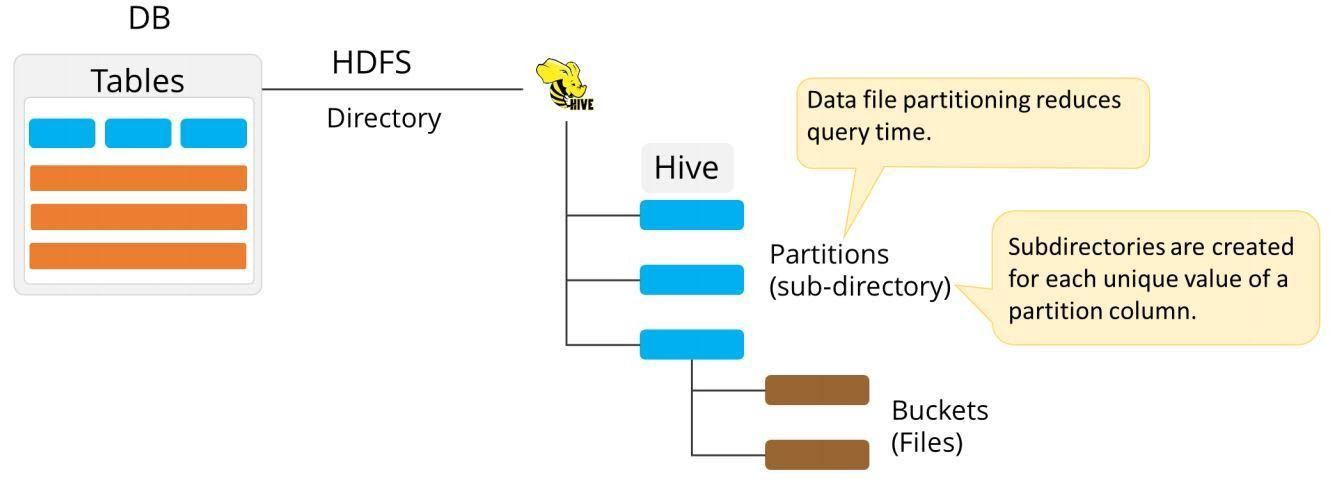
hive> create table emp\_bucket(Id int, Name string , Salary float)

CLUSTER BY (id) INTO 3 BUCKETS

row format delimited

fields terminated by ',' ;

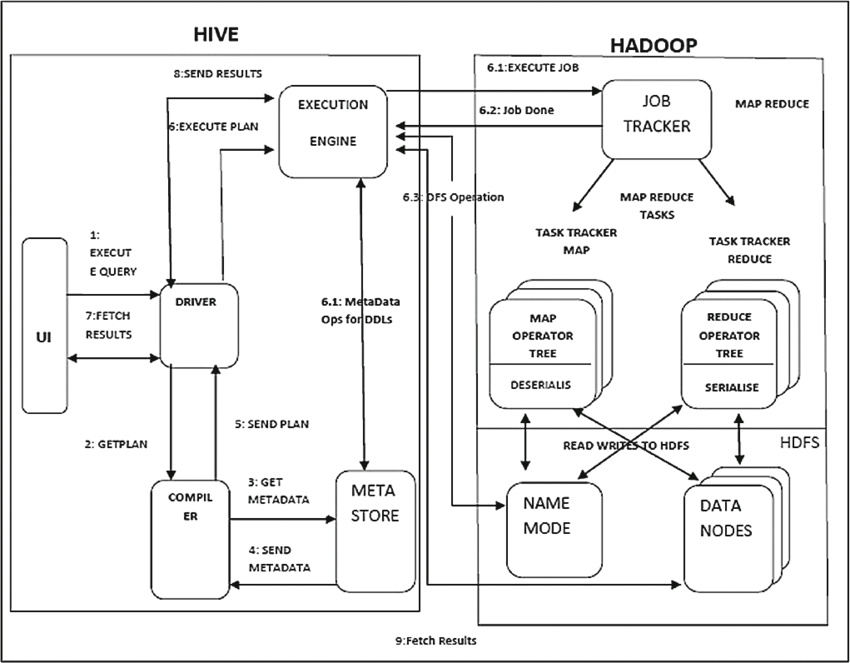
### Arborescence HDFS d’une table buckettée :



* Les buckets peuvent être mis en place à partir de la table

# Apache Hive: Architecture

* Le driver gère les sessions HQL
* Parser analyse la syntaxe
* Planner gère les plan d’exécution
* Optimiser optimise le plan d’exécution
* Metastore : point central de l’architecture Hive



# Formats de fichier supportés par Hive ?

* Le format par défaut supporté par Hive est l’ORC
* Text File, Sequence File, RC File, Avro File, Parquet et Protobuf