

Atelier : Utiliser HIVE

HIVE peut être utilisé pour

L'objectif :

- Stocker et réaliser des calculs sur les statistiques de baseball de 1871 à 2011.
 - Nous allons trouver le joueur avec le plus de « run » pour chaque année.

Emplacement du fichier : /formation/ateliers/hive/

Réalisation : Vous aller créer des tables Hive pour stocker les données statistiques puis requêter ces tables pour obtenir les informations voulues

Chapitre correspondant : HIVE

1- Visualiser les données statistiques des batteurs :

- a- Ouvrir le terminal en ligne de commande de votre VM, et aller lire le fichier de Batting.csv situé dans le répertoire /formation/ateliers/hive/

```
# cd /formation/ateliers/hive/  
# less Batting.csv  
# less Master.csv
```

2- Copier ce fichier dans HDFS (soit via IHM soit via PIG)

```
grunt> copyFromLocal /formation/ateliers/hive/Batting.csv /demo/data
```

1- Créer une table temporaire pour stocker les données:

- a- Se connecter à Hive

```
# hive
```

- a- Créer une table externe pour visualiser le fichier des batteurs

```
create external table temp_batting ( playerID STRING, yearID INT, stint STRING,  
teamID STRING, lgID STRING, G STRING, G_batting STRING, AB STRING, R INT, H STRING,  
second_base STRING, third_base STRING, HR STRING, RBI STRING, SB STRING, CS STRING,  
BB STRING, SO STRING, IBB STRING, HBP STRING, SH STRING, SF STRING, GIDP STRING,  
G_old STRING)  
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','  
LOCATION '/user/hive/warehouse/demo';
```

- b- Que s'est-il passé sous hdfs /user/hive/warehouse/

```
Le dossier /user/hive/warehouse/demo est créé
```

- c- Requêtez la table et vous verrez qu'elle est vide.

- a- Déposer le fichier Batting.csv dans le répertoire hdfs /user/hive/warehouse/demo

```
grunt> cp Batting.csv /user/hive/warehouse/demo
```

Maintenant la table temp_batting permet de requêter le contenu du fichier présent dans /user/hive/warehouse/demo.

2- Créer une table pour stocker les données:

Créer une table nommée « batting » contenant l'identifiant des batteurs, l'année et le nombre de runs réalisés

```
create table batting (player_id STRING, year INT, runs INT);
```

La table batting est une table interne hive. Le repertoire /user/hive/warehouse/batting a été créé vide.

3- Identifier pour chaque année le batteur qui a réalisé le plus de Run et insérer ces informations dans la table batting:

```
insert overwrite table batting
SELECT tb.playerid,
       tb.yearid,
       tb.r
FROM temp_batting tb
JOIN
  (SELECT yearid y,
          max(r) AS runs
   FROM temp_batting
   GROUP BY yearid) my
  ON (tb.yearid=my.y
      AND tb.r=my.runs) ;
```

Vérifier le dossier /user/hive/warehouse/batting

Hive possède 2 colonnes virtuelles créées automatiquement pour chaque table
table: INPUT__FILE__NAME and BLOCK__OFFSET__INSIDE__FILE.

4- Récupérer le nom du fichier de stockage, l'id du joueur dont les runs sont supérieur à 150.

```
select INPUT__FILE__NAME, player_id FROM batting WHERE
runs > 150;
```

1- Supprimer la table temp_batting:

La table n'apparaît plus.

2- Vérifier le dossier /user/hive/warehouse/demo.

Le répertoire et le fichier Batting.csv existent toujours car c'est une table externe

3- Que se passe si nous recréons la table ?

Elle sera automatiquement alimentée avec les données de Batting.csv

4- Créer une table « local » temp_master

```
create table temp_master (col_value STRING);
```

5- Alimenter la table temp_master avec le fichier /user/cloudera/demo/Master.csv

```
LOAD DATA INPATH '/user/cloudera/demo/Master.csv' OVERWRITE INTO TABLE  
temp_master;
```

6- Vérifier si le fichier se trouve toujours dans le repertoire hdfs user/cloudera/demo?

7- Que se passe-t-il si nous supprimons la table temp master.csv ?

```
drop table temp_master;
```

le fichier /user/hive/warehouse/temp_master/Master.csv est lui aussi supprimé.

Partitions

8- Créer une table names qui contient une colonne id (entier) et une colonne name (text) et une colonne state (texte). Cette table sera partitionnée par la colonne state.

```
create table names (id int, name string)  
partitioned by (state string)  
row format delimited fields terminated by '\t';
```

Note : on remarque que la colonne de partition ne doit pas être répétée dans le script.

9- Charger les fichiers hivedata_<<state>>.txt dans la table names

```
load data local inpath  
'/home/cloudera/formation/ateliers/hive/hivedata_ca.txt'  
into table names partition (state = 'CA');  
  
load data local inpath  
'/home/cloudera/formation/ateliers/hive/hivedata_co.txt'  
into table names;  
FAILED: SemanticException [Error 10062]: Need to specify partition columns  
because the destination table is partitioned  
  
load data local inpath  
'/home/cloudera/formation/ateliers/hive/hivedata_co.txt'  
into table names partition (state = 'CO');
```

```
load data local inpath
'/home/cloudera/formation/ateliers/hive/hivedata_sd.txt'
into table names partition (state = 'SD');
```

Attention : si on exécute depuis l'éditeur hive via Hue et qu'on a une erreur « Permission denied » => c'est qu'il n'y a pas les bons droits sur les fichiers en local.

10- Vérifier que toutes les données sont bien dans la table

```
hive> select * from names;
OK
1 Ulf CA
2 Manish CA
3 Brian CA
4 George CO
5 Mark CO
6 Rich SD
```

11- Vérifier les partitions avec la commande show partitions <table>

```
hive> show partitions names;
OK
state=CA
state=CO
state=SD
```

12- Comment se traduit les partitions sous hdfs :

```
hive> dfs -ls -R /apps/hive/warehouse/names/;
```

	DFS Output
1	drwxrwxrwx - cloudera supergroup 0 2016-08-26 05:21 /user/hive/warehouse/names/state=CA
2	-rwxrwxrwx 1 cloudera supergroup 22 2016-08-26 05:21 /user/hive/warehouse/names/state=CA/hivedata_ca.txt
3	drwxrwxrwx - cloudera supergroup 0 2016-08-26 05:24 /user/hive/warehouse/names/state=CO
4	-rwxrwxrwx 1 cloudera supergroup 15 2016-08-26 05:24 /user/hive/warehouse/names/state=CO/hivedata_co.txt
5	drwxrwxrwx - cloudera supergroup 0 2016-08-26 05:26 /user/hive/warehouse/names/state=SD
6	-rwxrwxrwx 1 cloudera supergroup 6 2016-08-26 05:26 /user/hive/warehouse/names/state=SD/hivedata_sd.txt

Remarque : Chaque partition possède son répertoire pour stocker ses données

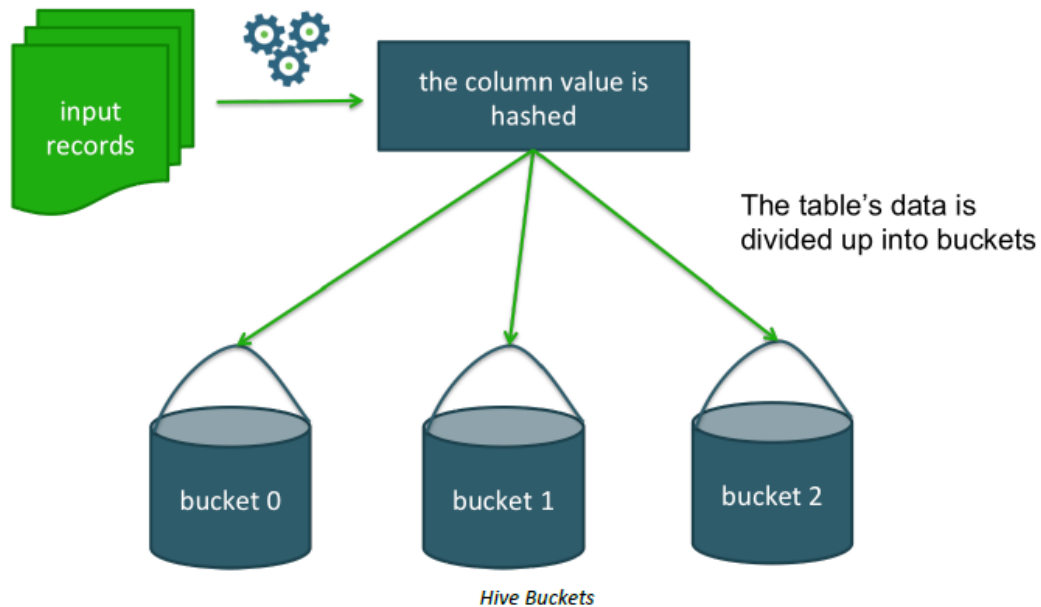
13- Quand dans une requête on spécifie la partition, Hive est beaucoup plus performant et va lire seulement le répertoire correspondant

```
hive> select * from names where state = 'CA';
OK
1 Ulf CA
2 Manish CA
3 Brian CA
```

14- Remarque : il n'y a pas de job MapReduce qui s'est exécuté. Pourquoi ?

le résultat de la requête est exactement le contenu du fichier de sous partition, donc il n'y a pas besoin de MapReduce,

Bucket



15- Exécuter la requête suivante :

```
create table names_bucket (id int, name string, state string)
  clustered by (id) into 2 buckets;
```

Chargement via load data => KO

```
load data local inpath
'/home/cloudera/formation/ateliers/hive/hivedata_ca.txt'
into table names_bucket ;
```

Il n'y aura que des NULL qui seront insérés car nous n'avons pas défini la table avec row format delimited. Si on fait un clustered, cela n'a pas de sens de charger avec un load data car hive créer des fichiers en fonction du nombre de bucket donc il faut charger avec des inserts. (Étape intermédiaire de load data dans une table temp puis création de bucket.

Par contre si nous faisons un insert into alors les données seront insérées :

16- Insérer les données :

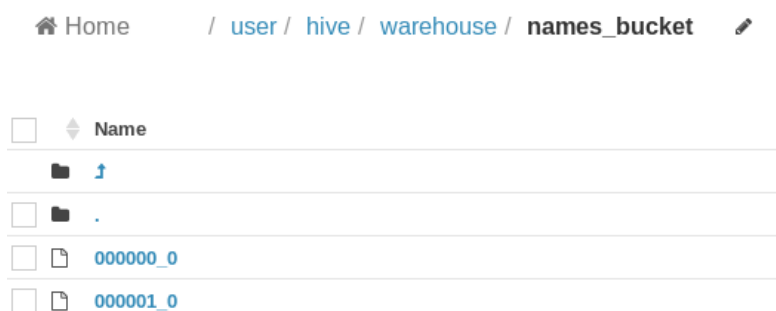
```
insert into names_bucket
select * from names where state ="CA";
```

17- Faire un select * from names_bucket et aller voir dans HDFS

18- Insérer les données de names en écrasant les données de la table names_bucket , puis aller voir dans HDFS
:

```
set hive.enforce.bucketing=true ;

insert overwrite table names_bucket
select * from names ;
```



Home	/	user	/	hive	/	warehouse	/	names_bucket	
<input type="checkbox"/>		Name							
<input type="checkbox"/>		↑							
<input type="checkbox"/>		.							
<input type="checkbox"/>		000000_0							
<input type="checkbox"/>		000001_0							

Skewed

Create a Skewed Table

a) **Supprimer le dossier** /user/cloudera/demo/data

```
hdfs dfs -rmr /user/cloudera/demo/data
```

b) **Copier le fichier salaries.txt présent dans** /home/cloudera/formation/ateliers/hive/ **dans HDFS** /user/cloudera/demo.

```
hdfs dfs -put salaries.txt /user/cloudera/demo/salaries.txt
```

c) **créer une table externe salarydata qui contient les données du fichier salaries.txt avec les colonnes gender (string), age (int), salary(double), zip(int)**

```
drop table if exists salarydata;

create external table salarydata (
    gender string,
    age int,
    salary double,
    zip int
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LOCATION '/user/cloudera/demo/';
```

- d) Créer une table skew_demo qui contient la meme structure que la table salarydata avec comme skewed values le code postal 94040 et 95102. Créer un fichier skew_demo.hive et exécuter le.

```
set hive.mapred.supports.subdirectories=true;
set hive.optimize.listbucketing=true;
set mapred.input.dir.recursive=true;

drop table if exists skew_demo;

create table skew_demo (
  gender string,
  age int,
  salary double,
  zip int
)
skewed by (zip) on (95102,94040) stored as directories;

insert overwrite table skew_demo
select gender,age,salary,zip from salarydata;
```

- e) Vérifier le contenu du dossier skew_demo : Que constatez-vous ?

🏠 Home / user / hive / warehouse / skew_demo ▼ History 🔍

<input type="checkbox"/>	⚡ Name	⬆️ Size	⚡ User	⚡ Group	⚡ Permissions	Date
<input type="checkbox"/>	📁 ↗		hive	supergroup	drwxrwxrwx	December 19, 20
<input type="checkbox"/>	📁 .		cloudera	supergroup	drwxrwxrwx	December 19, 20
<input type="checkbox"/>	📁 HIVE_DEFAULT_LIST_BUCKETING_DIR_NAME		cloudera	supergroup	drwxrwxrwx	December 19, 20
<input type="checkbox"/>	📁 zip=94040		cloudera	supergroup	drwxrwxrwx	December 19, 20
<input type="checkbox"/>	📁 zip=95102		cloudera	supergroup	drwxrwxrwx	December 19, 20

Ordre de tri (important)

La syntaxe pour trier : On peut utiliser order ou sort

```
select * from table_name [order | sort] by column_name;
```

Mais le comportement est différent s'il y a plusieurs reducers.

sort va trier par « reducer » alors que « order by » va trier sur tous les reducer mais pour cela il faut mettre la propriété suivante :

```
hive.optimize.sampling.orderby=true
```