Lab 6: Apache Hive

L'objectif de ce TP est de :

- installation d'apache Hive
- Première utilisation d'apache Hive
- Réaliser des requêtes analytiques

Apache Hive est un software **datawarehouse** conçu pour lire, écrire et gérer de grands ensembles de données extraits du système de fichiers distribué d'Apache Hadoop (HDFS). En effet, il ne s'agit pas d'une base de données complète. Il ne stocke que les métadonnées et les données sont stockées uniquement dans HDFS. Ainsi, chaque requête écrite par l'utilisateur est convertie en code MapReduce qui interagit ensuite avec HDFS. Hive peut être utilisé comme système OLAP (Online Analytical Processing).

Hive est fourni avec HiveServer2, une interface serveur dotée de sa propre interface de ligne de commande (CLI) appelée **Beeline (client JDBC)**, qui permet de se connecter à Hive,

I. Installation Apache Hive

- Pull l'image depuis dockerHub : https://hub.docker.com/r/apache/hive/tags. La dernière image est 4.0.0-alpha-2

```
docker pull apache/hive:4.0.0-alpha-2
```

```
C:\Users\hp>docker pull apache/hive:4.0.0-alpha-2
4.0.0-alpha-2: Pulling from apache/hive
4f4fb700ef54: Pull complete
56a130901b37: Pull complete
eddb431f2fca: Pull complete
9d948bbdc842: Pull complete
239b6597911f: Pull complete
ee7a22f107a6: Pull complete
cd30f3d1e783: Pull complete
1a2de4cc9431: Pull complete
1efc276f4ff9: Pull complete
a2f2f93da482: Pull complete
60930a9c0895: Pull complete
b237b9d3d33a: Pull complete
d2421c7a4bbf: Pull complete
Digest: sha256:69e482fdcebb9e07610943b610baea996c941bb36814cf233769b8a4db41f9c1
Status: Downloaded newer image for apache/hive:4.0.0-alpha-2
docker.io/apache/hive:4.0.0-alpha-2
```

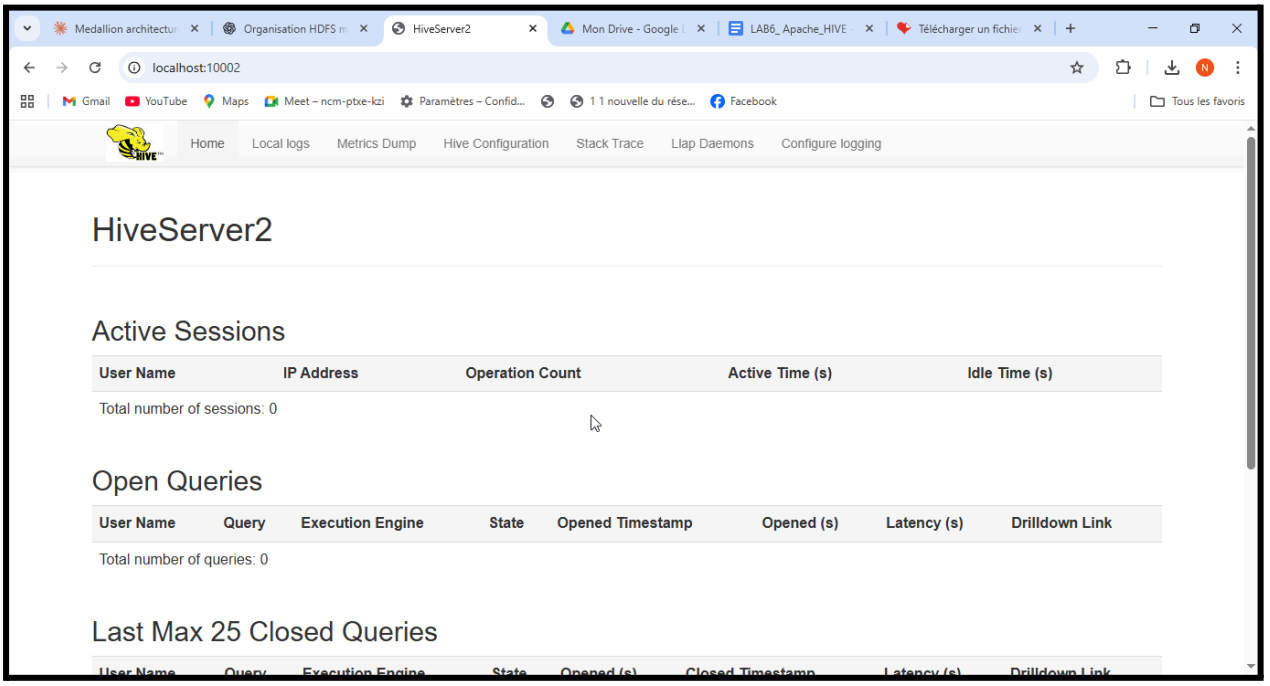
Pour une configuration rapide, démarrer HiveServer2 avec le Metastore « embedded »
(SGBD Derby comme BD metastore)

- Lancer ensuite les démons yarn et hdfs:

```
:\Users\hp>docker run -v ~/Documents/hadoop_project/:/shared_volume -d -p 10000:10000 -p 10002:10002 -p 9083:9083 --env
 SERVICE_NAME=hiveserver2 --name hiveserver2-standalone apache/hive:4.0.0-alpha-2
22e98caea016d3b79d2c6174dd02c788e32c578fc8117b037ac0d6e87799d12f

:\Users\hp>docker ps
ONTAINER ID   IMAGE                     COMMAND                  CREATED             STATUS              PORTS

            NAMES
22e98caea016   apache/hive:4.0.0-alpha-2   "sh -c /entrypoint.sh"   About a minute ago   Up About a minute   0.0.0.0:908
3->9083/tcp, [::]:9083->9083/tcp, 0.0.0.0:10000->10000/tcp, [::]:10000->10000/tcp, 0.0.0.0:10002->10002/tcp, [::]:10002-
>10002/tcp   hiveserver2-standalone

:\Users\hp>
```

- Pour accéder à HiveServer2 web via le navigateur via

  http://localhost:10002



II. Première utilisation beeline

Pour se connecter à serveur hive **hiveserver2**, on peut utiliser à l'invité de commande **Beeline** en fournissant une adresse IP et le port sur la chaîne d'URL de connexion JDBC. (Par défaut, le serveur écoute sur le port=10000)

- Accéder au shell du conteneur hiveserver2-standalone

```
C:\Users\DELL>docker exec -it hiveserver2-standalone bash
hive@bfb991da4f72:/opt/hive$ hadoop fs -ls
```

- HDFS est souvent déjà démarré dans les configurations Docker par défaut. Vérifier que HDFS est opérationnel :

```
hadoop fs -ls
```

```
:\Users\hp>docker exec -it hiveserver2-standalone bash
ive@22e98caea016:/opt/hive$ hadoop fs -ls
ound 16 items
rw-r--r--   1 root root      22349 2022-07-10 09:19 LICENSE
rw-r--r--   1 root root        230 2022-07-10 09:19 NOTICE
rw-r--r--   1 root root      29213 2022-07-10 09:19 RELEASE_NOTES.txt
rwxr-xr-x   - root root       4096 2023-05-02 18:52 bin
rwxr-xr-x   - root root       4096 2023-05-02 18:52 binary-package-licenses
rwxr-xr-x   - hive root       4096 2025-11-02 02:06 conf
rwxr-xr-x   - root root       4096 2023-05-02 18:52 contrib
rwxr-xr-x   - root root       4096 2023-05-02 18:52 data
rw-r--r--   1 hive hive      19959 2025-11-02 02:06 derby.log
rwxr-xr-x   - root root       4096 2023-05-02 18:52 examples
rwxr-xr-x   - root root       4096 2023-05-02 18:52 hcatalog
rwxr-xr-x   - root root       4096 2023-05-02 18:52 jdbc
rwxr-xr-x   - root root      20480 2023-05-02 18:52 lib
rwxr-xr-x   - hive hive       4096 2025-11-02 02:06 metastore_db
rwx-wx-wx   - hive hive       4096 2025-11-02 02:06 scratch_dir
rwxr-xr-x   - root root       4096 2023-05-02 18:52 scripts
ive@22e98caea016:/opt/hive$
```

- visualiser le contenu du fichier de configuration ***hive-site.xml*** dans ***/opt/hive/conf/***

```
hive@22e98caea016:/opt/hive$ cat /opt/hive/conf/hive-site.xml
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<configuration>
```

```
-->
<configuration>
    <property>
        <name>hive.server2.enable.doAs</name>
        <value>false</value>
    </property>
    <property>
        <name>hive.tez.exec.inplace.progress</name>
        <value>false</value>
    </property>
    <property>
        <name>hive.exec.scratchdir</name>
        <value>/opt/hive/scratch_dir</value>
    </property>
    <property>
        <name>hive.user.install.directory</name>
        <value>/opt/hive/install_dir</value>
    </property>
    <property>
        <name>tez.runtime.optimize.local.fetch</name>
        <value>true</value>
    </property>
    <property>
        <name>hive.exec.submit.local.task.via.child</name>
        <value>false</value>
    </property>
    <property>
        <name>mapreduce.framework.name</name>
        <value>local</value>
    </property>
```

- Accéder au shell de hive beeline en utilisant la commande suivante : login « **scott** » avec mot de passe « ***tiger*** »)

```
beeline -u jdbc:hive2://localhost:10000 scott tiger
```

- afficher les bases de données disponibles:

```
0: jdbc:hive2://localhost:10000> show databases ;
INFO  : Compiling command(queryId=hive_20251102015937_0ebab8ab-8b14-4ba4-ab5a-249c60ca8e87): show
atabases
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:[FieldSchema(name:database_name, type:string, com
ent:from deserializer)], properties:null)
INFO  : Completed compiling command(queryId=hive_20251102015937_0ebab8ab-8b14-4ba4-ab5a-249c60ca8e
7); Time taken: 3.304 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hive_20251102015937_0ebab8ab-8b14-4ba4-ab5a-249c60ca8e87): show
atabases
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20251102015937_0ebab8ab-8b14-4ba4-ab5a-249c60ca8e
7); Time taken: 0.126 seconds
+----------------+
| database_name  |
+----------------+
| default        |
+----------------+
1 row selected (4.274 seconds)
```

III. Analyse de données de réservation d'hôtels

Dans ce TP, nous allons travailler sur un ensemble de données concernant les réservations d'hôtels. L'objectif principal est de manipuler, analyser et extraire des informations pertinentes sur les clients, les hôtels et leurs réservations. Pour ce faire, nous utiliserons Apache Hive pour stocker et interroger les données. Les données sont disponibles dans trois fichiers :

1. Créer la base de données

- Créer la base de données hotel_booking ;

```
CREATE DATABASE hotel_booking;
USE hotel_booking;
```

- Lister le contenu du répertoire /opt/hive/data/warehouse. Qu'est ce que vous remarquez ?

```
C:\Users\DELL>docker exec -it hiveserver2-standalone /bin/bash
hive@bfb991da4f72:/opt/hive$ ls -l /opt/hive/data/warehouse
total 4
drwxr-xr-x 2 hive hive 4096 Nov  2 02:03 hotel_booking.db
```

2. Créer les tables

- Créer les tables pour stocker les informations des clients et des hôtels.

- Pour employer les partitions et les buckets, il faudra commencer par activer les propriétés suivantes :

```
0: jdbc:hive2://localhost:10000> set hive.exec.dynamic.partition
=true;
No rows affected (0.162 seconds)
```

```
0: jdbc:hive2://localhost:10000> set hive.exec.dynamic.partition
.mode=nonstrict;
No rows affected (0.013 seconds)
```

```
CREATE TABLE clients ( client_id INT, nom STRING, email STRING,
telephone STRING )
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

```
No rows affected (0.009 seconds)
0: jdbc:hive2://localhost:10000> CREATE TABLE clients ( client_i
d INT, nom STRING, email STRING,
. . . . . .> telephone STRING )
. . . . . .> ROW FORMAT DELIMITED FIELDS TER
MINATED BY ','
. . . . . .> STORED AS TEXTFILE;
INFO  : Compiling command(queryId=hive_20251102022541_cc917332-3
19c-4b12-83f2-12d07d7b3199): CREATE TABLE clients ( client_id IN
T, nom STRING, email STRING,
telephone STRING )
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:null, propertie
s:null)
INFO  : Completed compiling command(queryId=hive_20251102022541_
cc917332-319c-4b12-83f2-12d07d7b3199); Time taken: 0.367 seconds
INFO  : Concurrency mode is disabled, not creating a lock manage
r
INFO  : Executing command(queryId=hive_20251102022541_cc917332-3
19c-4b12-83f2-12d07d7b3199): CREATE TABLE clients ( client_id IN
T, nom STRING, email STRING,
telephone STRING )
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20251102022541_
cc917332-319c-4b12-83f2-12d07d7b3199); Time taken: 0.752 seconds
No rows affected (1.167 seconds)
```

Créer la table ***reservations*** avec une partition par ***date_debut*** pour optimiser les requêtes en fonction des dates.

```
0: jdbc:hive2://localhost:10000> CREATE TABLE reservations (
. . . . . .>     reservation_id INT,
. . . . . .>     client_id INT,
. . . . . .>     hotel_id INT,
. . . . . .>     nb_nuits INT,
. . . . . .>     prix_total DOUBLE
. . . . . .> )
. . . . . .> PARTITIONED BY (date_debut STRI
NG)
. . . . . .> ROW FORMAT DELIMITED
. . . . . .> FIELS TERMINATED BY ','
. . . . . .> STORED AS TEXTFILE;
INFO  : Compiling command(queryId=hive_20251102022720_91d4bd59-5
f93-47b5-b548-e8e79b8ce830): CREATE TABLE reservations (
reservation_id INT,
client_id INT,
hotel_id INT,
nb_nuits INT,
prix_total DOUBLE
)
PARTITIONED BY (date_debut STRING)
ROW FORMAT DELIMITED
FIELS TERMINATED BY ','
STORED AS TEXTFILE
```

3. Charger les données dans les tables

- Charger les données dans la table clients et hotels.
-

```
0: jdbc:hive2://localhost:10000> LOAD DATA LOCAL INPATH '/shared
_volume/DATSETS_HIVE/clients.txt' INTO TABLE clients;
INFO  : Compiling command(queryId=hive_20251102024327_30e240c8-d
421-4737-8388-eb08b079f0d5): LOAD DATA LOCAL INPATH '/shared_vol
ume/DATSETS_HIVE/clients.txt' INTO TABLE clients
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:null, propertie
s:null)
INFO  : Completed compiling command(queryId=hive_20251102024327_
30e240c8-d421-4737-8388-eb08b079f0d5); Time taken: 0.42 seconds
INFO  : Concurrency mode is disabled, not creating a lock manage
r
INFO  : Executing command(queryId=hive_20251102024327_30e240c8-d
421-4737-8388-eb08b079f0d5): LOAD DATA LOCAL INPATH '/shared_vol
ume/DATSETS_HIVE/clients.txt' INTO TABLE clients
INFO  : Starting task [Stage-0:MOVE] in serial mode
INFO  : Loading data to table default.clients from file:/shared_
volume/DATSETS_HIVE/clients.txt
INFO  : Starting task [Stage-1:STATS] in serial mode
INFO  : Executing stats task
INFO  : Table default.clients stats: [numFiles=1, numRows=0, tot
alSize=1579, rawDataSize=0, numFilesErasureCoded=0]
INFO  : Completed executing command(queryId=hive_20251102024327_
30e240c8-d421-4737-8388-eb08b079f0d5); Time taken: 0.922 seconds
No rows affected (1.374 seconds)
```

```
0: jdbc:hive2://localhost:10000> CREATE TABLE hotels (
. . . . . .>     hotel_id INT,
. . . . . .>     nom_hotel STRING,
. . . . . .>     ville STRING,
. . . . . .>     nombre_etoiles INT
. . . . . .> )
. . . . . .> ROW FORMAT DELIMITED
. . . . . .> FIELDS TERMINATED BY ','
. . . . . .> STORED AS TEXTFILE;
INFO  : Compiling command(queryId=hive_20251102024624_dc919bea-b
6ba-4f40-a30c-0655ce55c471): CREATE TABLE hotels (
hotel_id INT,
nom_hotel STRING,
ville STRING,
nombre_etoiles INT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
```

```
): jdbc:hive2://localhost:10000> LOAD DATA LOCAL INPATH '/shared
_volume/DATSETS_HIVE/hotels.txt' INTO TABLE hotels;
INFO  : Compiling command(queryId=hive_20251102024847_f22f1b69-6
7d0-42da-9816-e9ef0d2f5d8c): LOAD DATA LOCAL INPATH '/shared_vol
ume/DATSETS_HIVE/hotels.txt' INTO TABLE hotels
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:null, propertie
s:null)
INFO  : Completed compiling command(queryId=hive_20251102024847_
f22f1b69-67d0-42da-9816-e9ef0d2f5d8c); Time taken: 0.076 seconds
INFO  : Concurrency mode is disabled, not creating a lock manage
r
INFO  : Executing command(queryId=hive_20251102024847_f22f1b69-6
7d0-42da-9816-e9ef0d2f5d8c): LOAD DATA LOCAL INPATH '/shared_vol
ume/DATSETS_HIVE/hotels.txt' INTO TABLE hotels
INFO  : Starting task [Stage-0:MOVE] in serial mode
INFO  : Loading data to table default.hotels from file:/shared_v
olume/DATSETS_HIVE/hotels.txt
INFO  : Starting task [Stage-1:STATS] in serial mode
INFO  : Executing stats task
INFO  : Table default.hotels stats: [numFiles=1, numRows=0, tota
lSize=852, rawDataSize=0, numFilesErasureCoded=0]
INFO  : Completed executing command(queryId=hive_20251102024847_
f22f1b69-67d0-42da-9816-e9ef0d2f5d8c); Time taken: 0.606 seconds
No rows affected (0.717 seconds)
```

- Charger les données dans les tables reservations

```
s
--------------------------------------------------------------
------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNN
ING  PENDING  FAILED  KILLED
--------------------------------------------------------------
------------------------------
Map 1 .......... container     SUCCEEDED      1          1
  0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1
  0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1
  0        0       0       0
--------------------------------------------------------------
------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TI
ME: 5.87 s
--------------------------------------------------------------
```

4. Utiliser des partitions et des buckets

- Nous allons modifier la table ***hotels*** pour ajouter des partitions par ***ville*** et utiliser des buckets pour la table des réservations par ***client_id*** afin d'optimiser les jointures.

```
0: jdbc:hive2://localhost:10000> CREATE TABLE hotels_partitioned
 (
. . . . . .>     hotel_id INT,
. . . . . .>     nom_hotel STRING,
. . . . . .>     nombre_etoiles INT
. . . . . .> )
. . . . . .> PARTITIONED BY (ville STRING)
. . . . . .> ROW FORMAT DELIMITED
. . . . . .> FIELDS TERMINATED BY ','
. . . . . .> STORED AS TEXTFILE;
INFO  : Compiling command(queryId=hive_20251102025701_d1cd4a1f-7
f8f-405e-9a50-17944038e3be): CREATE TABLE hotels_partitioned (
hotel_id INT,
nom_hotel STRING,
nombre_etoiles INT
)
PARTITIONED BY (ville STRING)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
```

```
INFO  : Executing command(queryId=hive_20251102025808_a5a91758-2
d0a-4cbe-aa16-32079e1c4ee8): CREATE TABLE reservations_bucketed
(
reservation_id INT,
client_id INT,
hotel_id INT,
date_debut STRING,
date_fin STRING,
prix_total DECIMAL(10,2)
)
CLUSTERED BY (client_id) INTO 4 BUCKETS
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20251102025808_
a5a91758-2d0a-4cbe-aa16-32079e1c4ee8); Time taken: 0.065 seconds
```

```
--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1            container       RUNNING      1          0        1        0       0       0
Reducer 2        container        INITED      1          0        0        1       0       0
--------------------------------------------------------------------------------
       INITED      1          0
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2        container       RUNNING      1          0        1        0       0       0
--------------------------------------------------------------------------------
       INITED      1          0
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        1       0       0
--------------------------------------------------------------------------------
     SUCCEEDED      1          1
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 2.28 s
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  default.hotels_partitioned partiti
--------------------------------------------------------------------------------11-02_03-00-03_238_742629207061085
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
     SUCCEEDED      1          1
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      1          1        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 3.21 s
--------------------------------------------------------------------------------
```

- Charger les tables créés.
- Lister le contenu du répertoire /opt/hive/data/warehouse/hotel_booking.db/. Qu'est ce que vous remarquez ?

5. Utilisation de requêtes simples

- Lister tous les clients
- Lister tous les hôtels à Paris
- Lister toutes les réservations avec les informations sur les hôtels et les clients

6. Requêtes avec jointures

- Afficher le nombre de réservations par client
- Afficher les clients qui ont réservé plus que 2 nuitées
- Afficher les Hôtels réservés par chaque client
- Afficher les noms des hôtels dans lesquels il y a plus qu'une réservation.
- Afficher les noms des hôtels dans lesquels il y a pas de réservation.

7. Requêtes imbriquées

- Afficher les clients ayant réservé un hôtel avec plus de 4 étoiles
- Afficher le Total des revenus générés par chaque hôtel

8. Utilisation de fonctions d'agrégation avec partitions et buckets

- Revenus totaux par ville (partitionnée)
- Nombre total de réservations par client (bucketed)

9. Nettoyage et suppression des données

Supprimer les tables créés précédemment.

10. Script hql

Refaire le même traitement en organisant le code en trois scripts HiveQL distincts :

1. **Creation.hql** : Contient les commandes pour créer la base de données, les tables (clients, hôtels, réservations), ainsi que les partitions et les buckets si nécessaires.
2. **Loading.hql** : Contient les instructions pour charger les données depuis les fichiers texte (clients.txt, hotels.txt, reservations.txt) dans les tables Hive.
3. **Queries.hql** : Inclut toutes les requêtes SQL précédentes, telles que les jointures, les agrégations et les requêtes imbriquées.

Ces scripts doivent être exécutés dans l'ordre pour reproduire entièrement le traitement