

Tutorial 1 - Trabalhando um pouco com Hive

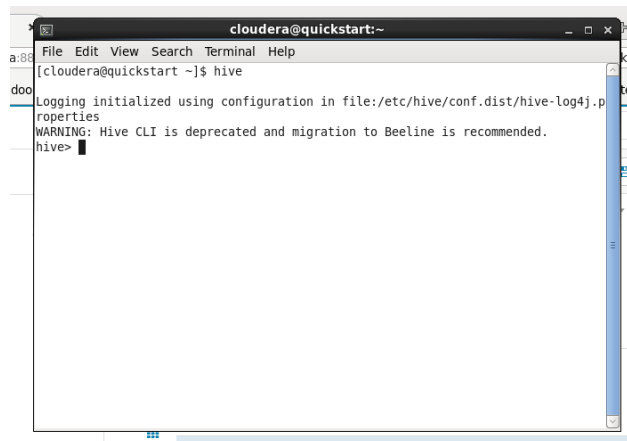
O HQL possui os comandos DDL e DML, isto é, **Hive Data Definition Language** (create/drop/alter/use) e **Hive Data Manipulation Language** (load, insert, update, delete, merge).

Neste tutorial vamos trabalhar no terminal, mas você pode fazer os mesmos comandos usando a interface do HUE disponível na máquina Cloudera.

Ativando o Hive Shell

Abra um terminal e digite "hive" em seguida aperte enter:

```
$ hive
```



Parte 1 - Criando uma base

Para criarmos uma base usamos o comando `CREATE DATABASE <nome base>`. Em nosso exemplo criaremos a base chamada testehive.

```
hive> create database testehive;
```

```
hive> create database testehive;  
OK  
Time taken: 4.951 seconds  
hive>
```



Podemos usar o comando opcional "IF NOT EXISTS" para indicar ao Hive que ele deve criar base apenas se ela não existir.

```
hive> create database if not exists testehive;
```

Podemos ver as bases do Hive com o seguinte comando:

```
hive> show databases;
```

```
hive> show databases;  
OK  
default  
testehive  
Time taken: 0.038 seconds, Fetched: 2 row(s)
```

O HQL nos permite usar LIKE para especificar strings:

```
hive> show databases like 't*';
```

```
hive> show databases like 't*';  
OK  
testehive  
Time taken: 0.049 seconds, Fetched: 1 row(s)
```

Podemos obter mais detalhes sobre uma base com o describe:

```
hive> describe database testehive;
```

```
hive> describe database testehive;  
OK  
testehive          hdfs://quickstart.cloudera:8020/user/hive/warehouse/test  
ehive.db           cloudera      USER  
Time taken: 0.458 seconds, Fetched: 1 row(s)
```

Parte 2 - Criando uma tabela

O Hive possui dois tipos de tabelas: **internas** e **externas**. Nas tabelas internas os dados são armazenados dentro da estrutura do Hive (user/hive/warehouse). No caso das tabelas externas o Hive cria uma referência para onde os dados estão armazenados. De maneira prática, se você exclui uma tabela interna o Hive apagará os dados, se você exclui uma tabela externa o Hive apagará apenas a referência e assim os dados estarão preservados na fonte.

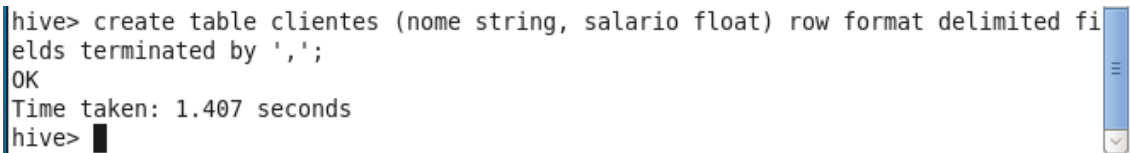
Para facilitar a criação da tabela vamos definir o banco de dados a ser usado. Em nosso caso, o testehive criado anteriormente. Digite o comando a seguir:

```
hive> use testehive;
```

```
hive> use testehive;  
OK  
Time taken: 0.053 seconds  
hive>
```

Neste exemplo, vamos criar uma tabela interna chamada “clientes” com os campos nome e salário. Para isso, usamos o CREATE TABLE e passamos a lista de campos da tabela.

```
hive> create table clientes (nome string, salario float) row format  
delimited fields terminated by ',';
```

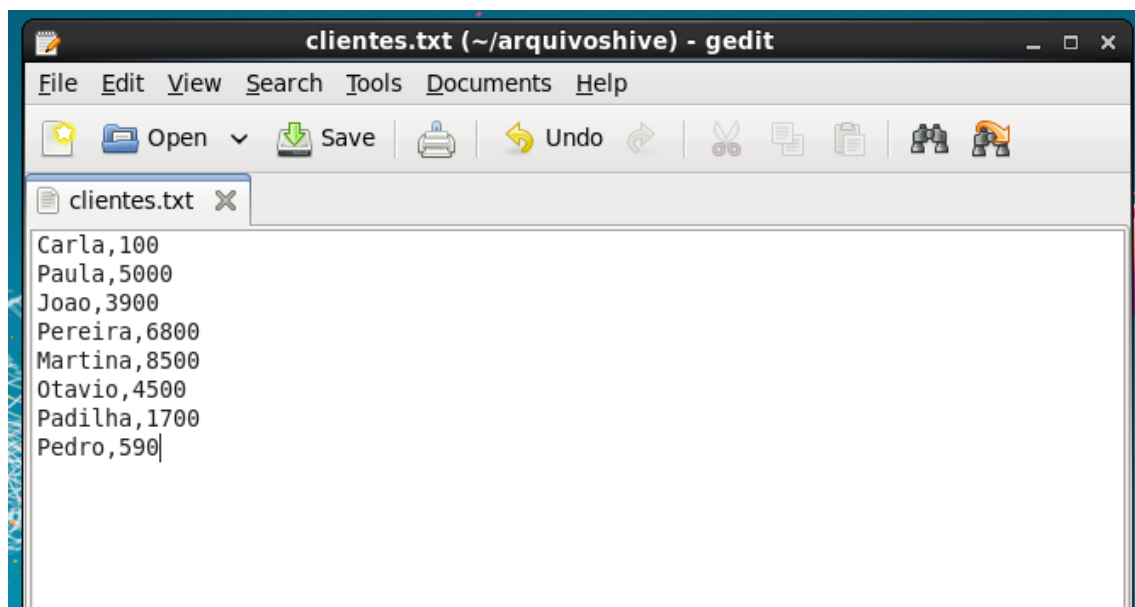


```
hive> create table clientes (nome string, salario float) row format delimited fi  
elds terminated by ',';  
OK  
Time taken: 1.407 seconds  
hive>
```

Parte 3 - Carregando arquivos para o Hive

Crie uma pasta chamada arquivoshive e salve um arquivo de texto na pasta com o nome clientes.txt com os seguintes valores:

Carla, 100
Paula, 5000
Joao, 3900
Pereira, 6800
Martina, 8500
Otavio, 4500
Padilha, 1700
Pedro, 590



clientes.txt (~/arquivoshive) - gedit

File Edit View Search Tools Documents Help

Open Save Undo

clientes.txt

Carla,100
Paula,5000
Joao,3900
Pereira,6800
Martina,8500
Otavio,4500
Padilha,1700
Pedro,590

Lembra que construímos a tabela com o separado ',' (delimited fields terminated by ','). Isso quer dizer que cada linha do arquivo será separada por virgula. Neste caso para cada linha teremos o nome e o salário, o Hive então colocará cada informação em sua respectiva coluna.

Carregando os dados

O LOAD permite carregar dados de arquivos simples para o Hive. 'LOCAL' significa que o arquivo de entrada está no sistema de arquivos local. Se 'LOCAL' for omitido, ele procurará o arquivo no HDFS.

```
hive> load data local inpath 'arquivoshive/clientes.txt' into table
clientes;
```

```
hive> load data local inpath 'arquivoshive/clientes.txt' into table clientes;
Loading data to table testehive.clientes
Table testehive.clientes stats: [numFiles=1, totalSize=92]
OK
Time taken: 0.804 seconds
```

Parte 4 - Carregando arquivos com partição

O Hive permite o particionamento por coluna em uma tabela. Isso quer dizer que o Hive criará uma estrutura de subdiretórios a partir do valor que atribuímos no campo particionado dentro do HDFS.

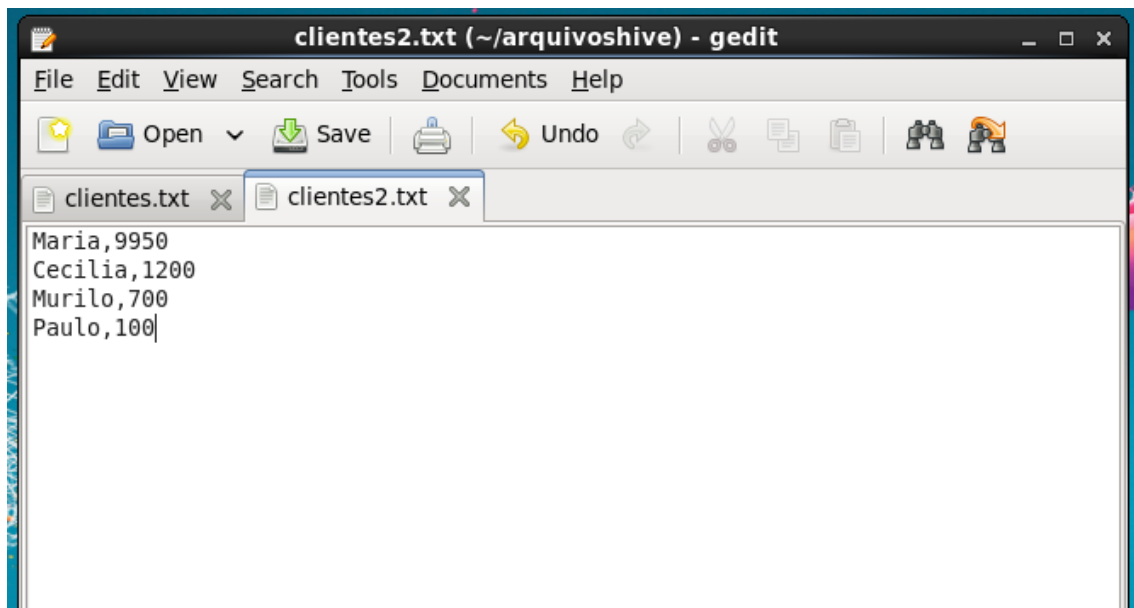
Vamos criar outra tabela chamada clientes2, com os campos nome e salário, e com o campo particionado data. Isto é, estamos **particionando de acordo com uma data** de inserção dos dados.

```
hive> create table clientes2 (nome string, salario float) partitioned
by (data string) row format delimited fields terminated by ',';
```

```
hive> create table clientes2 (nome string, salario float) partitioned by (data s
tring) row format delimited fields terminated by ',';
OK
Time taken: 0.148 seconds
```

Agora crie o arquivo clientes2.txt e salve na pasta arquivoshive com os seguintes valores:

```
Maria, 9950
Cecilia, 1200
Murilo, 700
Paulo, 100
```



Vamos carregar na tabela clientes2 os dois arquivos de texto, mas cada um será inserido **em uma partição diferente**.

```
hive> load data local inpath 'arquivoshive/clientes.txt' into table
clientes2 partition (data='2018-10-12');
```

```
hive> load data local inpath 'arquivoshive/clientes.txt' into table clientes2 pa
rtition (data='2018-10-12');
Loading data to table testehive.clientes2 partition (data=2018-10-12)
Partition testehive.clientes2{data=2018-10-12} stats: [numFiles=1, numRows=0, to
talSize=92, rawDataSize=0]
OK
Time taken: 1.621 seconds
```

```
hive> load data local inpath 'arquivoshive/clientes2.txt' into table
clientes2 partition (data='2018-10-21');
```

```
hive> load data local inpath 'arquivoshive/clientes2.txt' into table clientes2 p
artition (data='2018-10-21');
Loading data to table testehive.clientes2 partition (data=2018-10-21)
Partition testehive.clientes2{data=2018-10-21} stats: [numFiles=1, numRows=0, to
talSize=45, rawDataSize=0]
OK
Time taken: 0.91 seconds
```

Parte 5 - Manipulando os dados

Selecionando dados.

```
hive> select * from clientes;
```

```
hive> select * from clientes;  
OK  
Carla      100.0  
Paula      5000.0  
Joao       3900.0  
Pereira    6800.0  
Martina    8500.0  
Otavio     4500.0  
Padilha    1700.0  
Pedro      590.0  
Time taken: 3.673 seconds, Fetched: 8 row(s)
```

hive> select nome from clientes;

```
hive> select nome from clientes;  
OK  
Carla  
Paula  
Joao  
Pereira  
Martina  
Otavio  
Padilha  
Pedro  
Time taken: 0.327 seconds, Fetched: 8 row(s)
```

hive> select nome from clientes2;

```
hive> select * from clientes2;  
OK  
Carla      100.0    2018-10-12  
Paula      5000.0   2018-10-12  
Joao       3900.0   2018-10-12  
Pereira    6800.0   2018-10-12  
Martina    8500.0   2018-10-12  
Otavio     4500.0   2018-10-12  
Padilha    1700.0   2018-10-12  
Pedro      590.0    2018-10-12  
Maria      9950.0   2018-10-21  
Cecilia    1200.0   2018-10-21  
Murilo     700.0    2018-10-21  
Paulo      100.0    2018-10-21  
Time taken: 0.276 seconds, Fetched: 12 row(s)
```

Com partição

hive> select nome from clientes where data='2018-10-21';

```
hive> select * from clientes2 where data='2018-10-21';  
OK  
Maria      9950.0   2018-10-21  
Cecilia    1200.0   2018-10-21  
Murilo     700.0    2018-10-21  
Paulo      100.0    2018-10-21
```

Outras análises

```
hive> select nome from clientes where data='2018-10-21' and nome  
like '%a%';
```

```
hive> select * from clientes2 where data='2018-10-21' and nome like '%a%';  
OK  
Maria 9950.0 2018-10-21  
Cecilia 1200.0 2018-10-21  
Paulo 100.0 2018-10-21  
Time taken: 1.971 seconds, Fetched: 3 row(s)
```

Atividade

Nesta atividade, vamos trabalhar uma base que já temos no HDFS, a purchases.txt.

Se você tenha dúvida ou queria confirmar seu resultado, **as respostas estarão no final do tutorial.**

Tente fazer os seguintes passos:

1. Crie uma tabela externa apontando para o arquivo purchases.txt que está na pasta testehdfs. Lembre-se da estrutura do arquivo:

Data	Hora	Cidade	Item	Valor	Pagamento
2012-01-01	09:00	Fort Worth	Women's Clothing	153.57	Visa
2012-01-01	09:00	San Diego	Music	66.08	Cash
2012-01-01	09:00	Pittsburgh	Pet Supplies	493.51	Discover

2. Exiba o total de vendas por item.
3. Exiba o total de vendas por cidade.

Obs: caso você não tenha o arquivo na pasta testehdfs siga os passos:

1. Criar a pasta no HDFS (caso não tenha):
hdfs dfs -mkdir testehdfs
2. Copiar os dados da pasta testebigdata para a testehdfs:
hdfs dfs -put testebigdata/purchases.txt testehdfs

Caso não tenha o arquivo você pode baixar no link:

<https://raw.githubusercontent.com/juandecarrion/udacity-hadoop-course/master/testdata/purchases.txt>

Resposta:

USE testehive;

CREATE EXTERNAL TABLE IF NOT EXISTS compras (

data date,

hora string,

cidade string,

item string,

valor float,

pagamento string)

ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t'

LOCATION 'hdfs://quickstart.cloudera:8020/user/cloudera/testehdfs';

SELECT item, SUM(valor) FROM compras GROUP BY item;

SELECT cidade, SUM(valor) FROM compras GROUP BY cidade;