EXPERIMENT 6

Implementation of HIVE Commands

THEORY

What is HIVE?

Hive is a data warehouse system that is used to analyze structured data. It is built on top of Hadoop. It was developed by Facebook.

Hive provides the functionality of reading, writing, and managing large datasets residing in distributed storage. It runs SQL-like queries called HQL (Hive query language) which get internally converted to MapReduce jobs.

Using Hive, we can skip the requirement of the traditional approach of writing complex MapReduce programs. Hive supports Data Definition Language (DDL), Data Manipulation Language (DML), and User Defined Functions (UDF).

What is HQL?

Hive defines a simple SQL-like query language for querying and managing large datasets called Hive-QL (HQL). It’s easy to use if you’re familiar with SQL Language. Hive allows programmers who are familiar with the language to write the custom MapReduce framework to perform more sophisticated analysis.

Uses of Hive:

1. The Apache Hive distributed storage.

2. Hive provides tools to enable easy data extract/transform/load (ETL)

3. It provides the structure of a variety of data formats.

4. By using Hive, we can access files stored in Hadoop Distributed File System (HDFS is used for querying and managing large datasets residing in) or in other data storage systems such as Apache HBase.

What are the components of HIVE?

Metastore :

Hive stores the schema of the Hive tables in a Hive Metastore. Metastore is used to hold all the information about the tables and partitions that are in the warehouse. By default, the metastore is run in the same process as the Hive service and the default Metastore is DerBy Database.

SerDe :

Serializer, Deserializer gives instructions to hive on how to process a record.

HIVE Organization

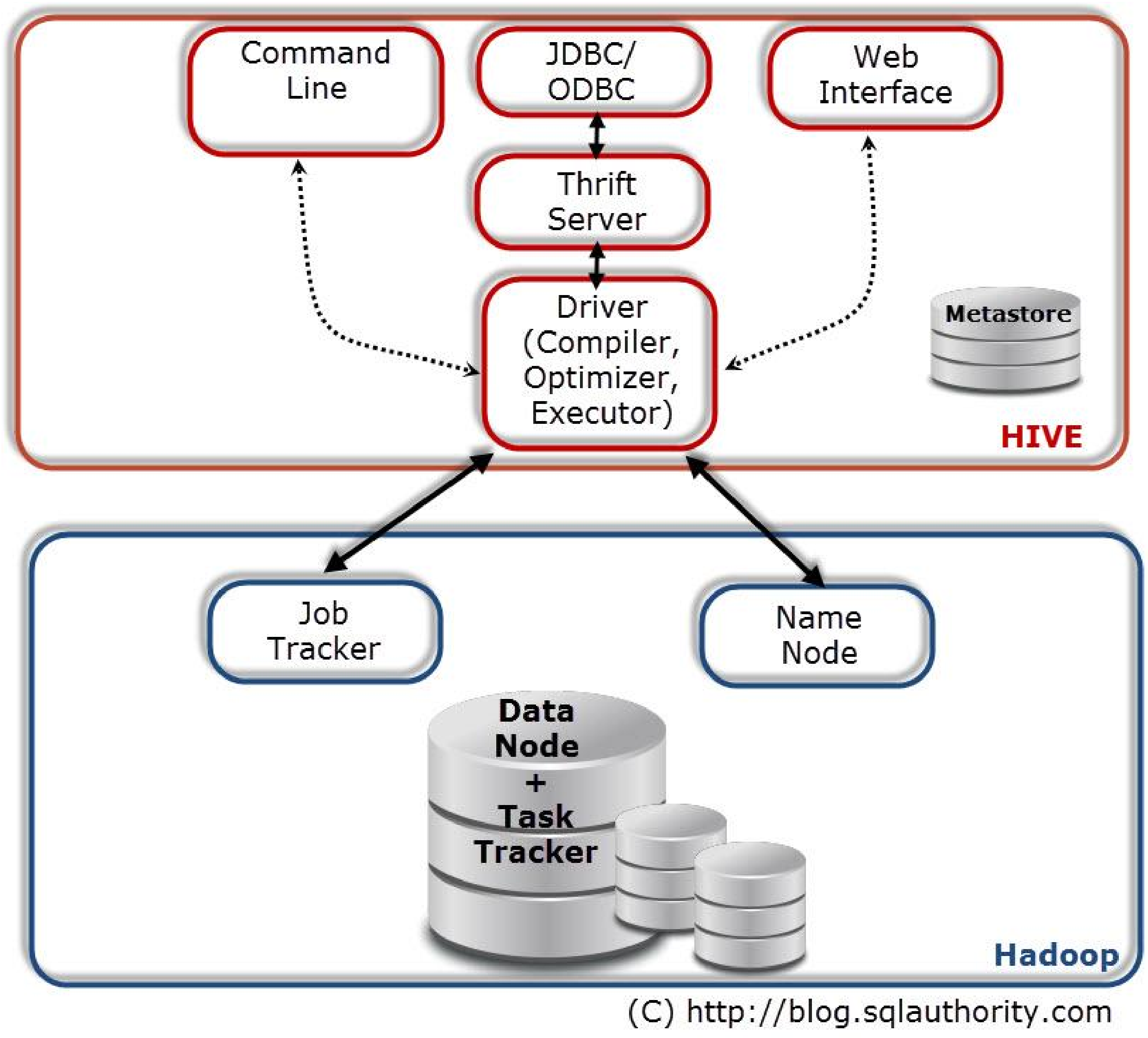
The data are organized in three different formats in HIVE.

Tables: They are very similar to RDBMS tables and contain rows and tables. Hive is just layered over the Hadoop File System (HDFS), hence tables are directly mapped to directories of the filesystems. It also supports tables stored in other native file systems.

Partitions: Hive tables can have more than one partition. They are mapped to subdirectories and file systems as well.

Buckets: In Hive data may be divided into buckets. Buckets are stored as files in a partition in the underlying file system.

Hive also has metastore which stores all the metadata. It is a relational database containing various information related to Hive Schema (column types, owners, key-value data, statistics, etc.). We can use MySQL database over here.



What are the limitations of HIVE?

• Hive is not designed for Online transaction processing (OLTP), it is only used for the Online Analytical Processing.

• Hive supports overwriting or apprehending data, but not updates and deletes.

• In Hive, sub queries are not supported.

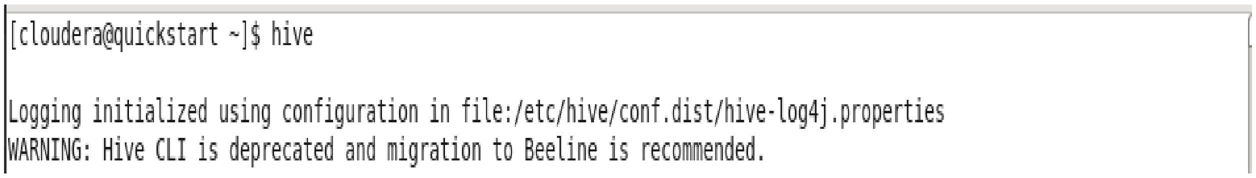
Why Hive is used inspite of Pig?

The following are the reasons why Hive is used in spite of Pig's availability:

- Hive-QL is a declarative language line SQL, PigLatin is a data flow language.
- Pig: a data-flow language and environment for exploring very large datasets.
- Hive: a distributed data warehouse.

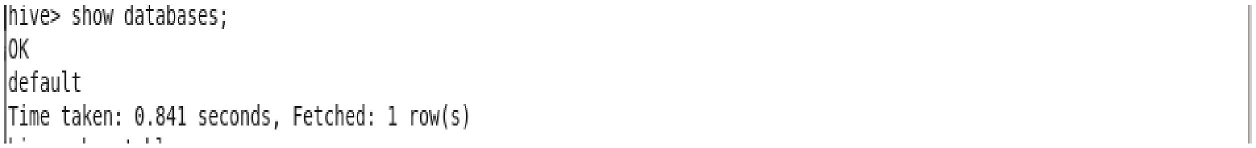
COMMANDS

1. To enter hive terminal
Command: hive



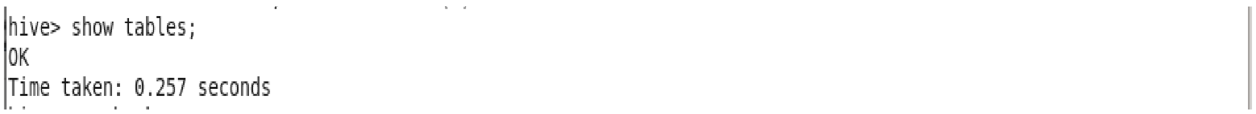
```
[cloudera@quickstart ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j.properties
WARNING: Hive CLI is deprecated and migration to Beeline is recommended.
```

2. To check the databases
Command: show databases;



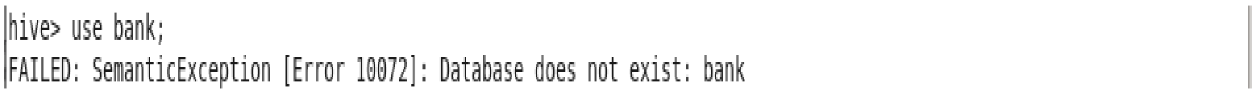
```
hive> show databases;
OK
default
Time taken: 0.841 seconds, Fetched: 1 row(s)
```

3. To check the tables
Command: show tables;



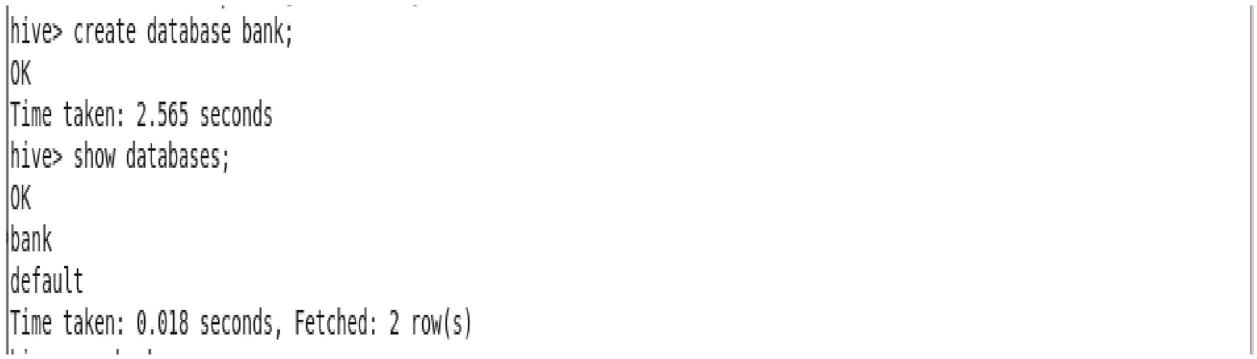
```
hive> show tables;
OK
Time taken: 0.257 seconds
```

4. To use a particular database
Command: use dbname;



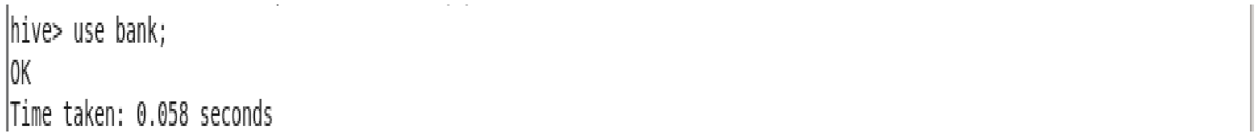
```
hive> use bank;
FAILED: SemanticException [Error 10072]: Database does not exist: bank
```

5. To create database
Command: create database retail;

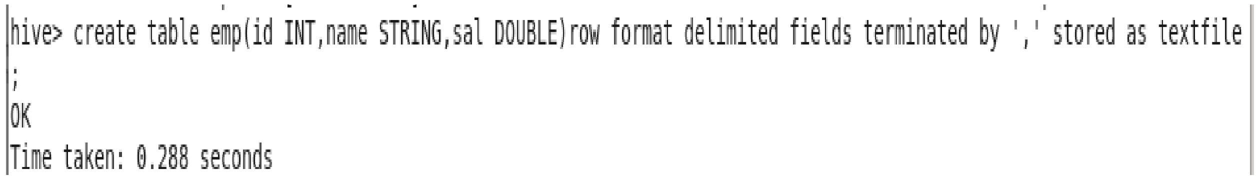


```
hive> create database bank;
OK
Time taken: 2.565 seconds
hive> show databases;
OK
bank
default
Time taken: 0.018 seconds, Fetched: 2 row(s)
```

6. To create table emp in retail database
Command: create table <tablename>;
Output:
hive> use retail;
hive> create table emp(id INT,name STRING,sal DOUBLE) row format delimited fields terminated by ',' stored as textfile;

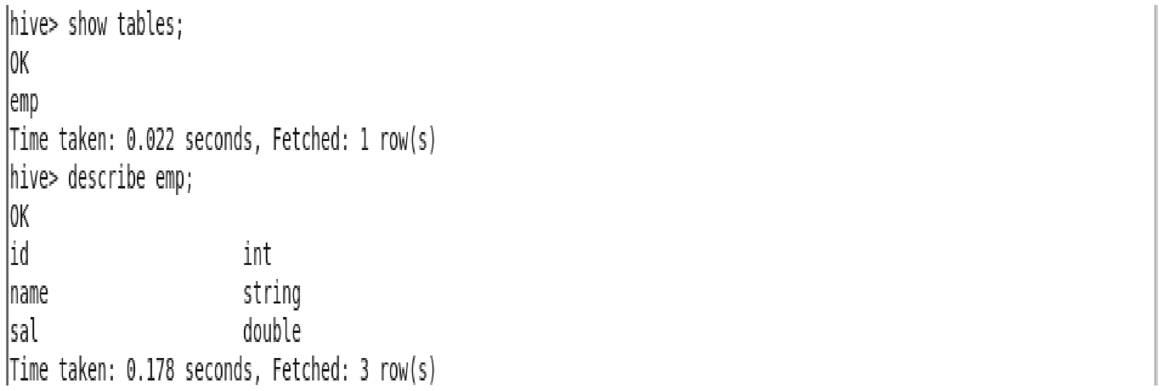


```
hive> use bank;
OK
Time taken: 0.058 seconds
```



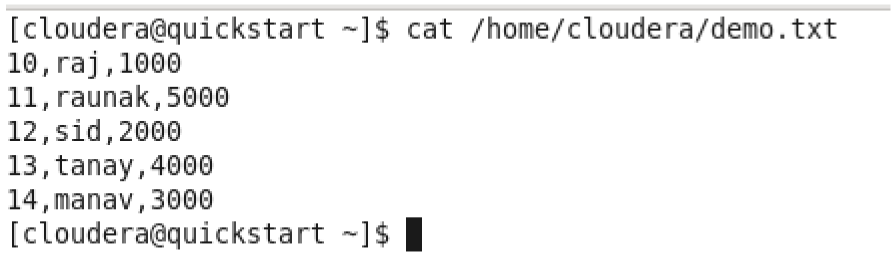
```
hive> create table emp(id INT,name STRING,sal DOUBLE)row format delimited fields terminated by ',' stored as textfile
;
OK
Time taken: 0.288 seconds
```

7. Schema information of table
Command: describe <tablename>;
Output:
hive> use retail;
hive> describe emp;

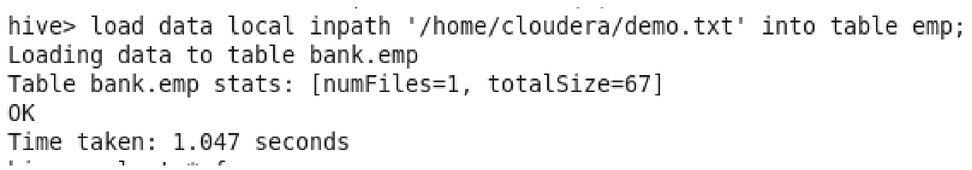


```
hive> show tables;
OK
emp
Time taken: 0.022 seconds, Fetched: 1 row(s)
hive> describe emp;
OK
id                  int
name                string
sal                 double
Time taken: 0.178 seconds, Fetched: 3 row(s)
```

8.To create file in training folder and save as
demo.txt
1,abc,2000
2,pqr,4500
To view contents of demo.txt file
[training@localhost ~]$ cat /home/training/demo.txt
1,abc,2000
2,pqr,4500
To load data from local path
hive> load data local inpath '/home/training/demo.txt' into table emp;

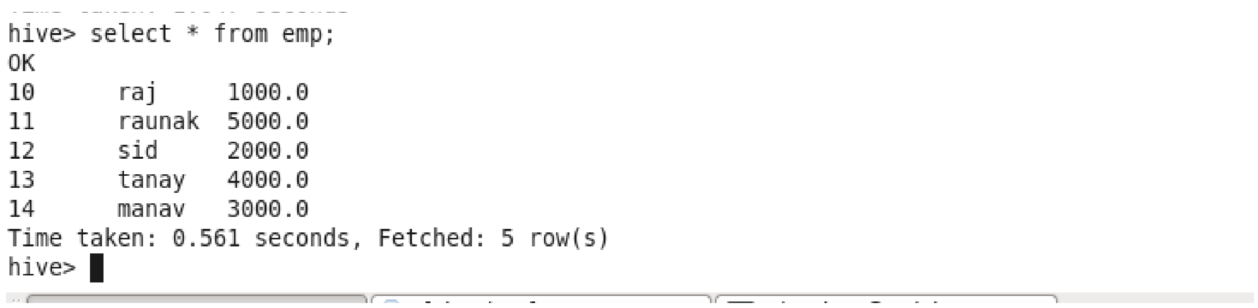


```
[cloudera@quickstart ~]$ cat /home/cloudera/demo.txt
10,raj,1000
11,raunak,5000
12,sid,2000
13,tanay,4000
14,manav,3000
[cloudera@quickstart ~]$ 
```



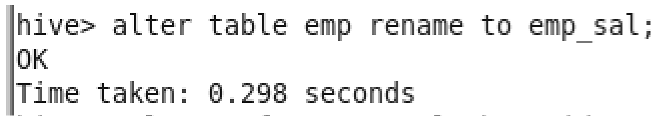
```
hive> load data local inpath '/home/cloudera/demo.txt' into table emp;
Loading data to table bank.emp
Table bank.emp stats: [numFiles=1, totalSize=67]
OK
Time taken: 1.047 seconds
```

9.To view contents of table
*Command: select * from emp;*
Output:



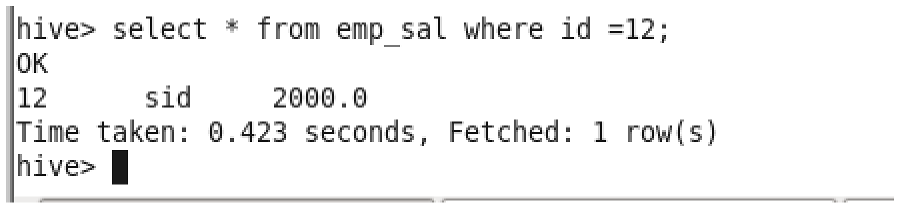
```
hive> select * from emp;
OK
10      raj     1000.0
11      raunak  5000.0
12      sid     2000.0
13      tanay   4000.0
14      manav   3000.0
Time taken: 0.561 seconds, Fetched: 5 row(s)
hive> 
```

10. To rename table name
Command: ALTER TABLE old_table_name RENAME TO new_table_name;
Output:
hive> use retail;
hive> alter table emp rename to emp_sal;



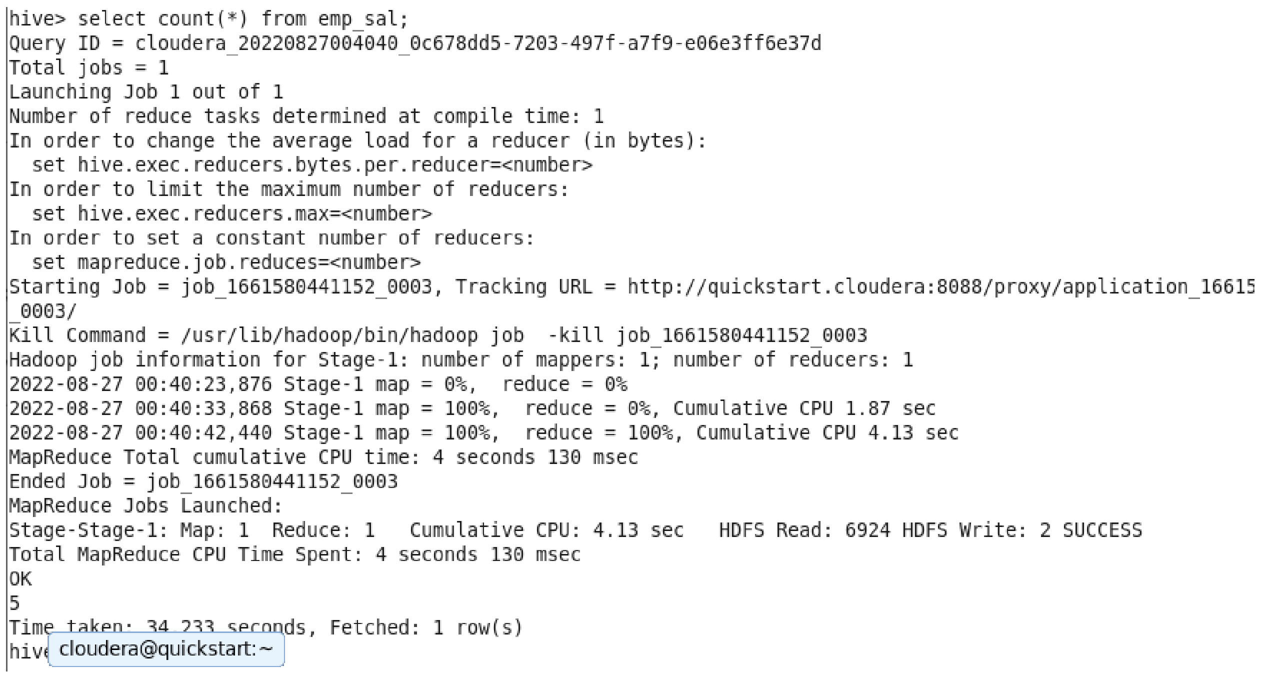
```
hive> alter table emp rename to emp_sal;
OK
Time taken: 0.298 seconds
```

11. Selecting data
*hive> select * from emp_sal where id=1;*



```
hive> select * from emp_sal where id =12;
OK
12      sid     2000.0
Time taken: 0.423 seconds, Fetched: 1 row(s)
hive>
```

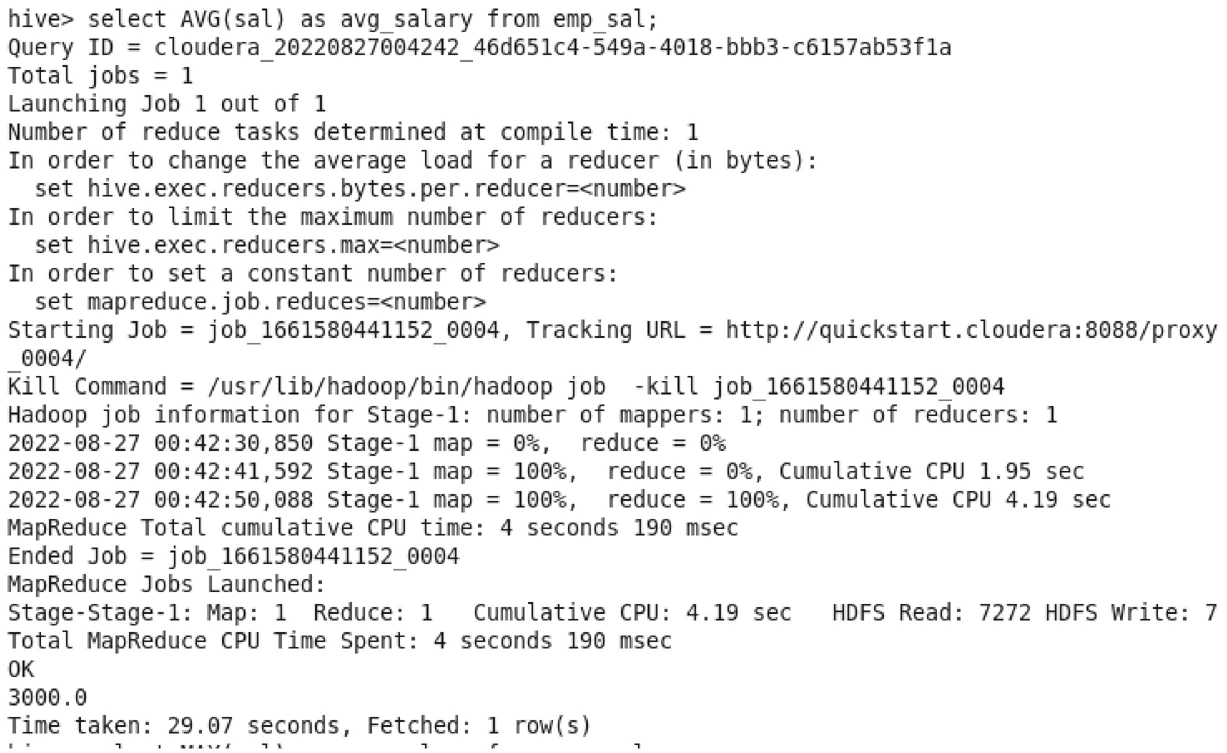
12. To count number of records in table
hive> select count(*) from emp_sal;



```
hive> select count(*) from emp_sal;
Query ID = cloudera_20220827004040_0c678dd5-7203-497f-a7f9-e06e3ff6e37d
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1661580441152_0003, Tracking URL = http://quickstart.cloudera:8088/proxy/application_16615
_0003/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1661580441152_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-08-27 00:40:23,876 Stage-1 map = 0%,  reduce = 0%
2022-08-27 00:40:33,868 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.87 sec
2022-08-27 00:40:42,440 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.13 sec
MapReduce Total cumulative CPU time: 4 seconds 130 msec
Ended Job = job_1661580441152_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 4.13 sec   HDFS Read: 6924 HDFS Write: 2 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 130 msec
OK
5
Time taken: 34.233 seconds, Fetched: 1 row(s)
hive cloudera@quickstart:~
```

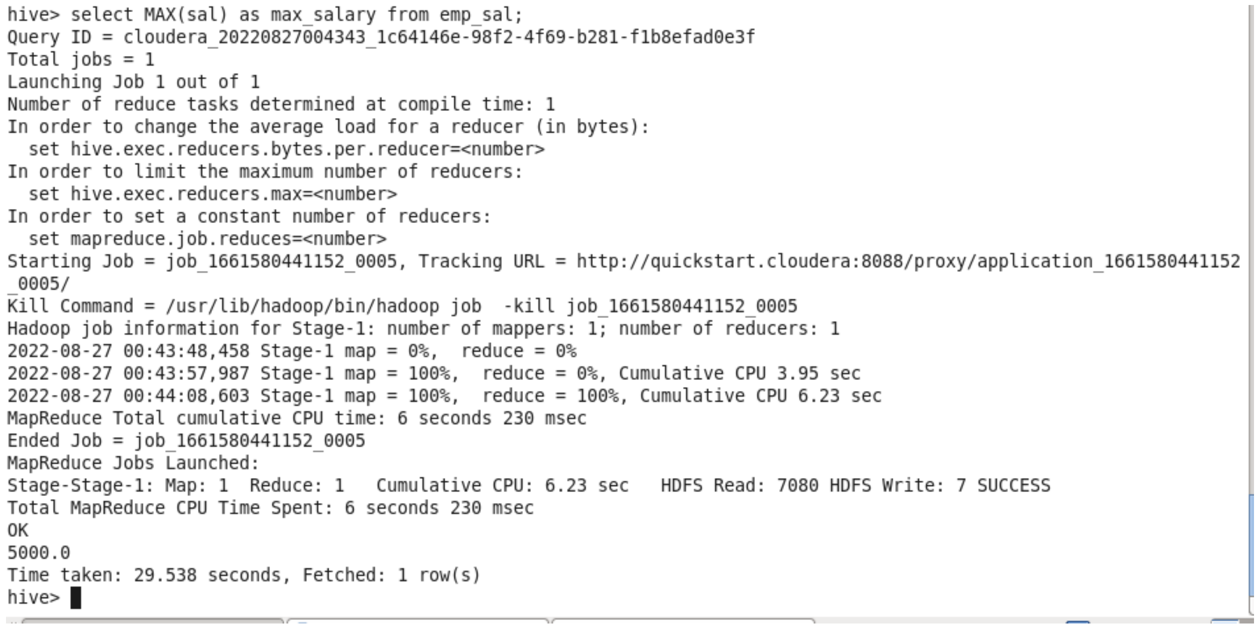
13. Try using aggregate commands using HQL(Try creating tables with group by fields and
execute the aggregate commands)

hive > select AVG(sal) as avg_salary from emp_sal;



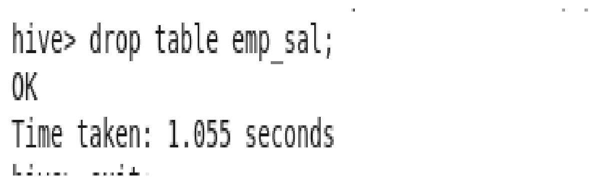
```
hive> select AVG(sal) as avg_salary from emp_sal;
Query ID = cloudera_20220827004242_46d651c4-549a-4018-bbb3-c6157ab53f1a
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1661580441152_0004, Tracking URL = http://quickstart.cloudera:8088/proxy
_0004/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1661580441152_0004
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-08-27 00:42:30,850 Stage-1 map = 0%,  reduce = 0%
2022-08-27 00:42:41,592 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 1.95 sec
2022-08-27 00:42:50,088 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.19 sec
MapReduce Total cumulative CPU time: 4 seconds 190 msec
Ended Job = job_1661580441152_0004
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 4.19 sec   HDFS Read: 7272 HDFS Write: 7
Total MapReduce CPU Time Spent: 4 seconds 190 msec
OK
3000.0
Time taken: 29.07 seconds, Fetched: 1 row(s)
```

hive > select MAX(sal) as max_salary from emp_sal;



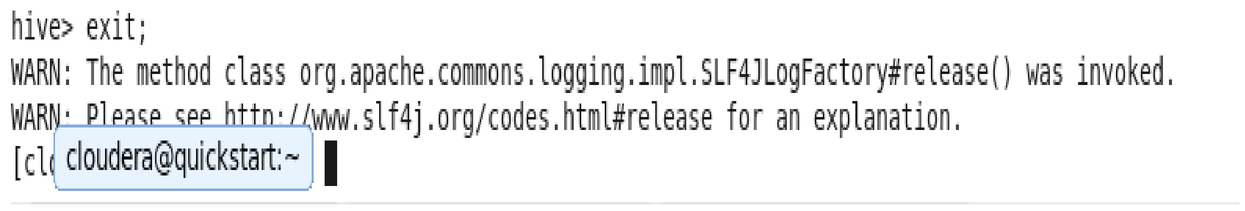
```
hive> select MAX(sal) as max_salary from emp_sal;
Query ID = cloudera_20220827004343_1c64146e-98f2-4f69-b281-f1b8efad0e3f
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1661580441152_0005, Tracking URL = http://quickstart.cloudera:8088/proxy/application_1661580441152
_0005/
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_1661580441152_0005
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2022-08-27 00:43:48,458 Stage-1 map = 0%,  reduce = 0%
2022-08-27 00:43:57,987 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.95 sec
2022-08-27 00:44:08,603 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 6.23 sec
MapReduce Total cumulative CPU time: 6 seconds 230 msec
Ended Job = job_1661580441152_0005
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 6.23 sec   HDFS Read: 7080 HDFS Write: 7 SUCCESS
Total MapReduce CPU Time Spent: 6 seconds 230 msec
OK
5000.0
Time taken: 29.538 seconds, Fetched: 1 row(s)
hive> 
```

14. To drop table
hive> drop table emp_sal;



```
hive> drop table emp_sal;
OK
Time taken: 1.055 seconds
```

15. To exit from Hive terminal
hive> exit;



```
hive> exit;
WARN: The method class org.apache.commons.logging.impl.SLF4JLogFactory#release() was invoked.
WARN: Please see http://www.slf4j.org/codes.html#release for an explanation.
[cl
```