Step Description:

Whenever we talk about a table, there are two things involved. One is the meta data of the table or the schema of the table and the second thing is the data itself.

Hive stores the meta data of a table in the metastore located in the Linux File System which is created when we install Hive and stores the data in the warehouse location which we create in the HDFS.

In this step we will do some additional configurations required to setup the Hive metastore and the warehouse and log in to the Hive Shell.

Commands:

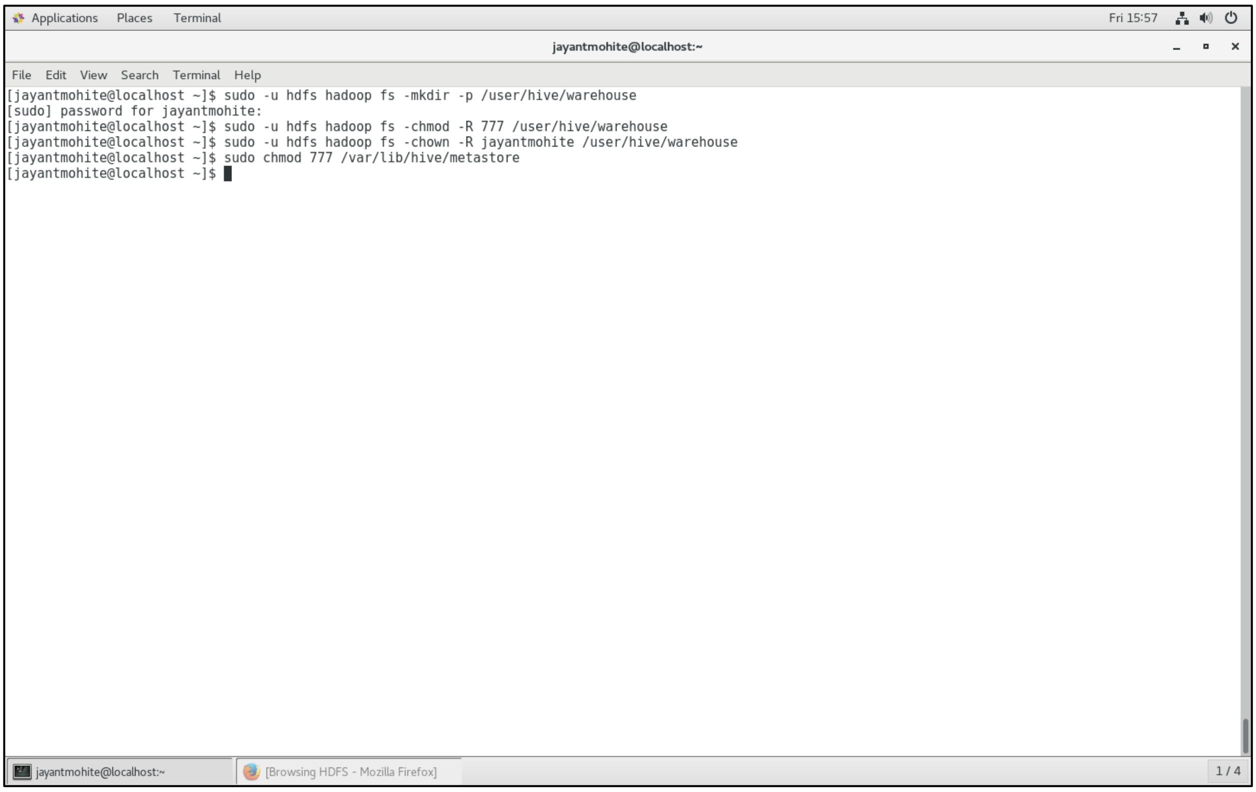
[jayantmohite@localhost] $ sudo -u hdfs hadoop fs -mkdir -p /user/hive/warehouse

[jayantmohite@localhost] $ sudo -u hdfs hadoop fs -chmod -R 777 /user/hive/warehouse

[jayantmohite@localhost] $ sudo -u hdfs hadoop fs -chown -R jayantmohite /user/hive/warehouse

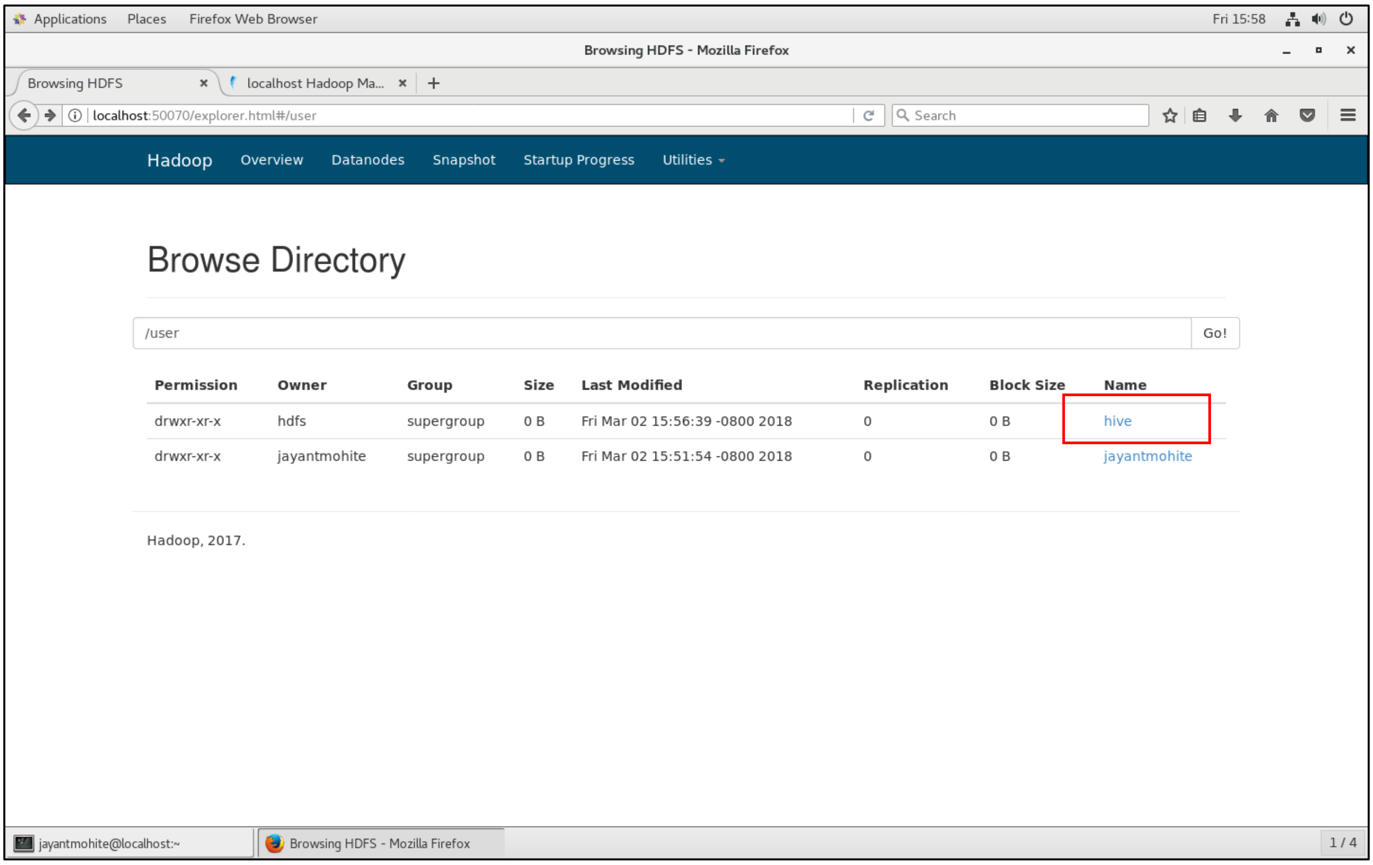
[jayantmohite@localhost] $ sudo chmod 777 /var/lib/hive/metastore

Step Visualization 1:



```
[jayantmohite@localhost ~]$ sudo -u hdfs hadoop fs -mkdir -p /user/hive/warehouse
[sudo] password for jayantmohite:
[jayantmohite@localhost ~]$ sudo -u hdfs hadoop fs -chmod -R 777 /user/hive/warehouse
[jayantmohite@localhost ~]$ sudo -u hdfs hadoop fs -chown -R jayantmohite /user/hive/warehouse
[jayantmohite@localhost ~]$ sudo chmod 777 /var/lib/hive/metastore
[jayantmohite@localhost ~]$ 
```

Step Visualization 2:



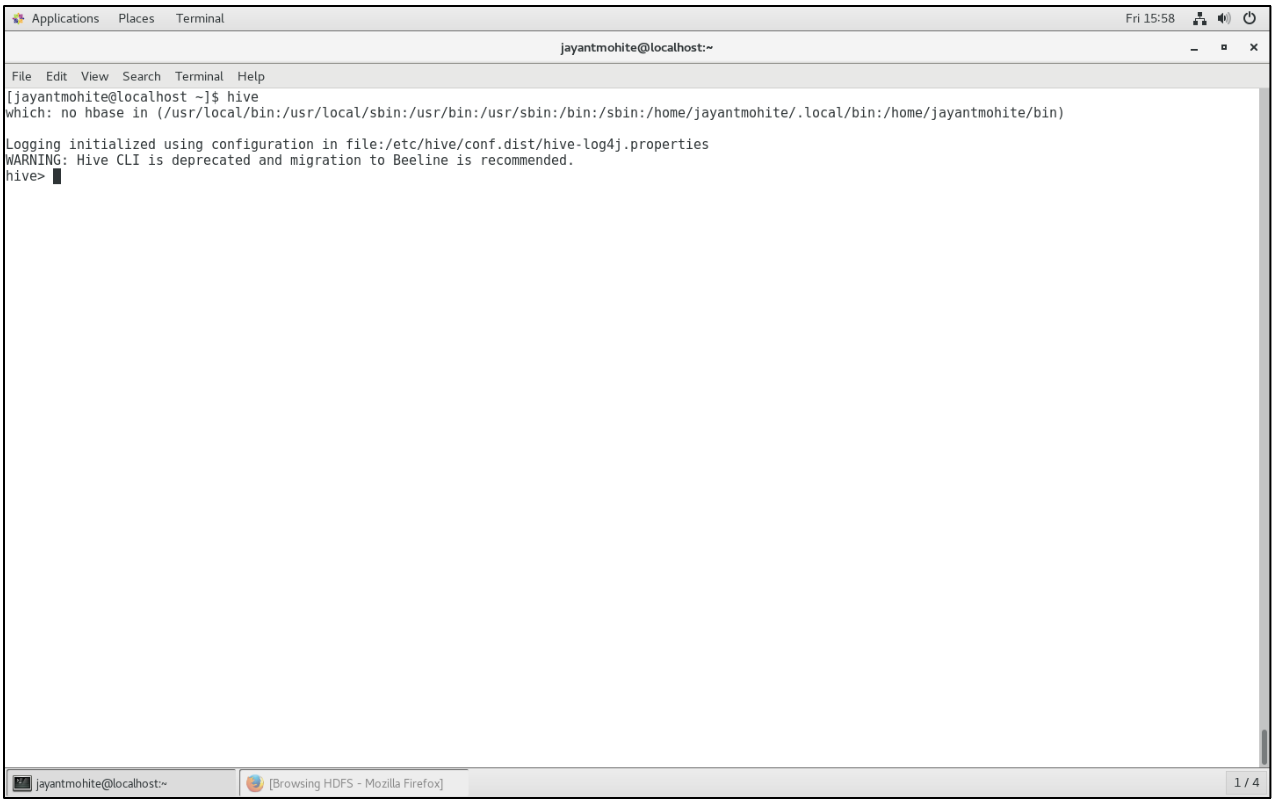
Browse Directory

/user

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	hdfs	supergroup	0 B	Fri Mar 02 15:56:39 -0800 2018	0	0 B	hive
drwxr-xr-x	jayantmohite	supergroup	0 B	Fri Mar 02 15:51:54 -0800 2018	0	0 B	jayantmohite

Hadoop, 2017.

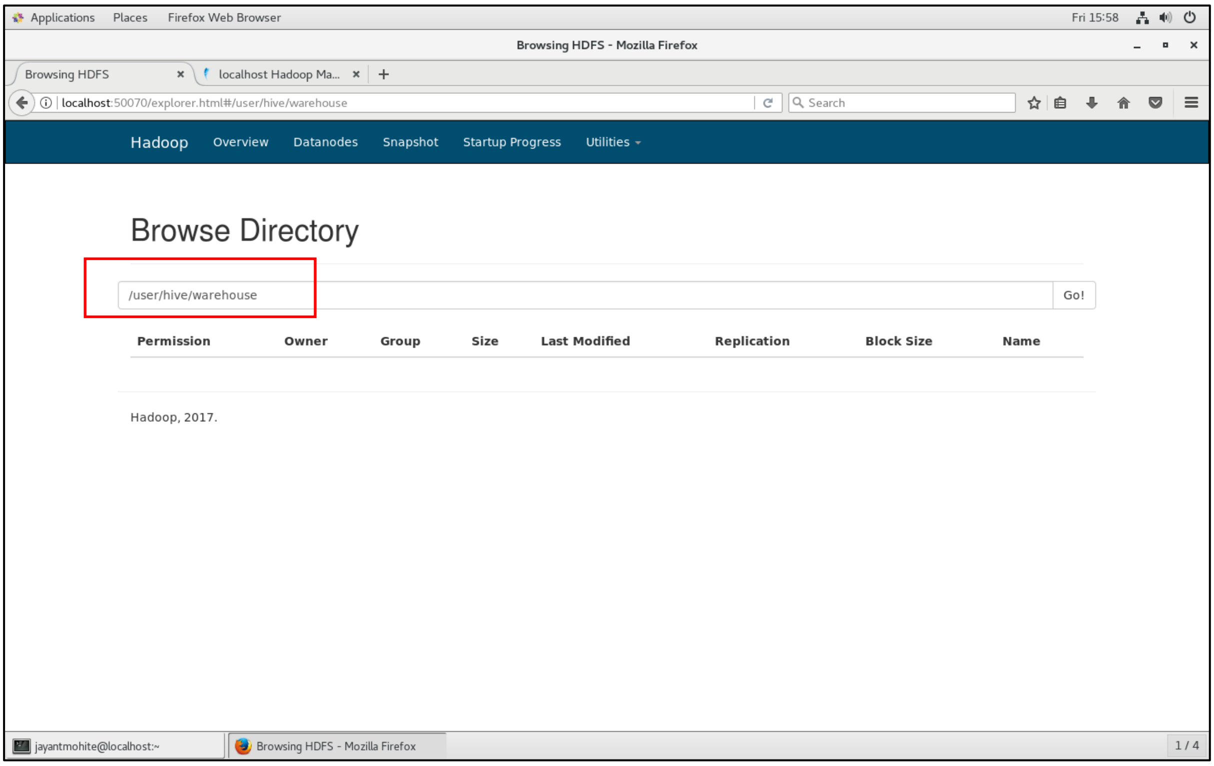
Step Visualization 3:



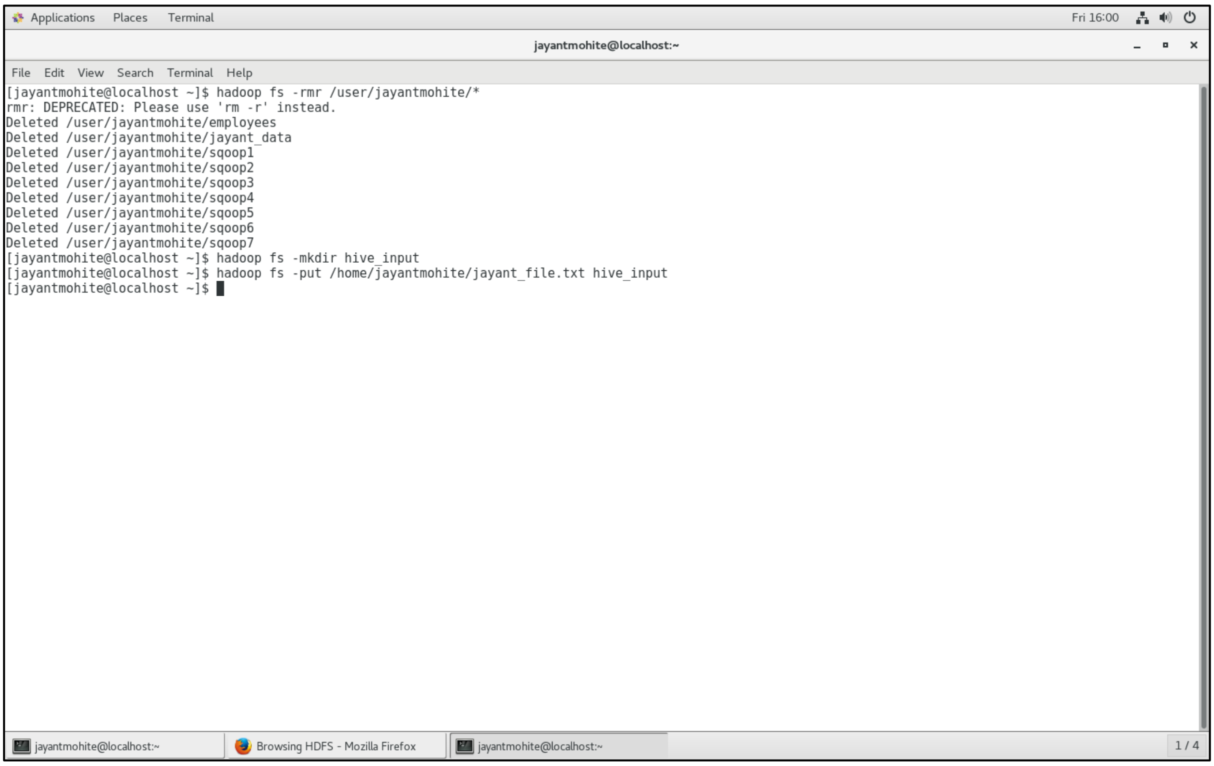
```
[jayantmohite@localhost ~]$ hive
which: no hbase in (/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin:/home/jayantmohite/.local/bin:/home/jayantmohite/bin)

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

Author: Jayant Mohite

Understanding Working of Hive

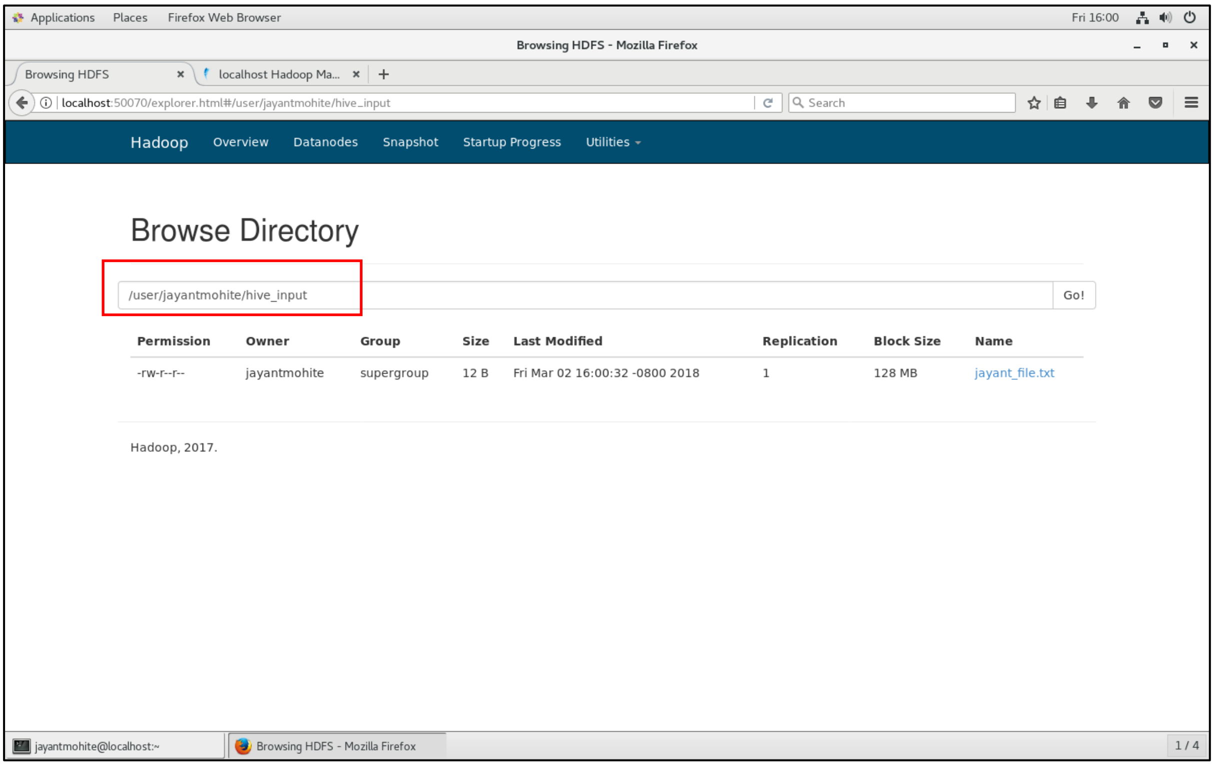


You can see that at this point of time the, warehouse is empty as we have not started working with Hive yet. Now, we will use the file jayant_file.txt as input for our initial operations.


Applications Places Terminal
Fri 16:00
jayantmohite@localhost:~
File Edit View Search Terminal Help
jayantmohite@localhost:~
Browsing HDFS - Mozilla Firefox
jayantmohite@localhost:~
1 / 4

```
[jayantmohite@localhost ~]$ hadoop fs -rmr /user/jayantmohite/*
rmr: DEPRECATED: Please use 'rm -r' instead.
Deleted /user/jayantmohite/employees
Deleted /user/jayantmohite/jayant_data
Deleted /user/jayantmohite/sqoop1
Deleted /user/jayantmohite/sqoop2
Deleted /user/jayantmohite/sqoop3
Deleted /user/jayantmohite/sqoop4
Deleted /user/jayantmohite/sqoop5
Deleted /user/jayantmohite/sqoop6
Deleted /user/jayantmohite/sqoop7
[jayantmohite@localhost ~]$ hadoop fs -mkdir hive_input
[jayantmohite@localhost ~]$ hadoop fs -put /home/jayantmohite/jayant_file.txt hive_input
[jayantmohite@localhost ~]$ 
```

Observe that the file is now available at HDFS location /user/jayantmohite/hive_input


Applications Places Firefox Web Browser Fri 16:00
Browsing HDFS - Mozilla Firefox
Browsing HDFS
localhost Hadoop Ma...
localhost:50070/explorer.html#/user/jayantmohite/hive_input
Search
Hadoop Overview Datanodes Snapshot Startup Progress Utilities
jayantmohite@localhost:~
Browsing HDFS - Mozilla Firefox
1 / 4

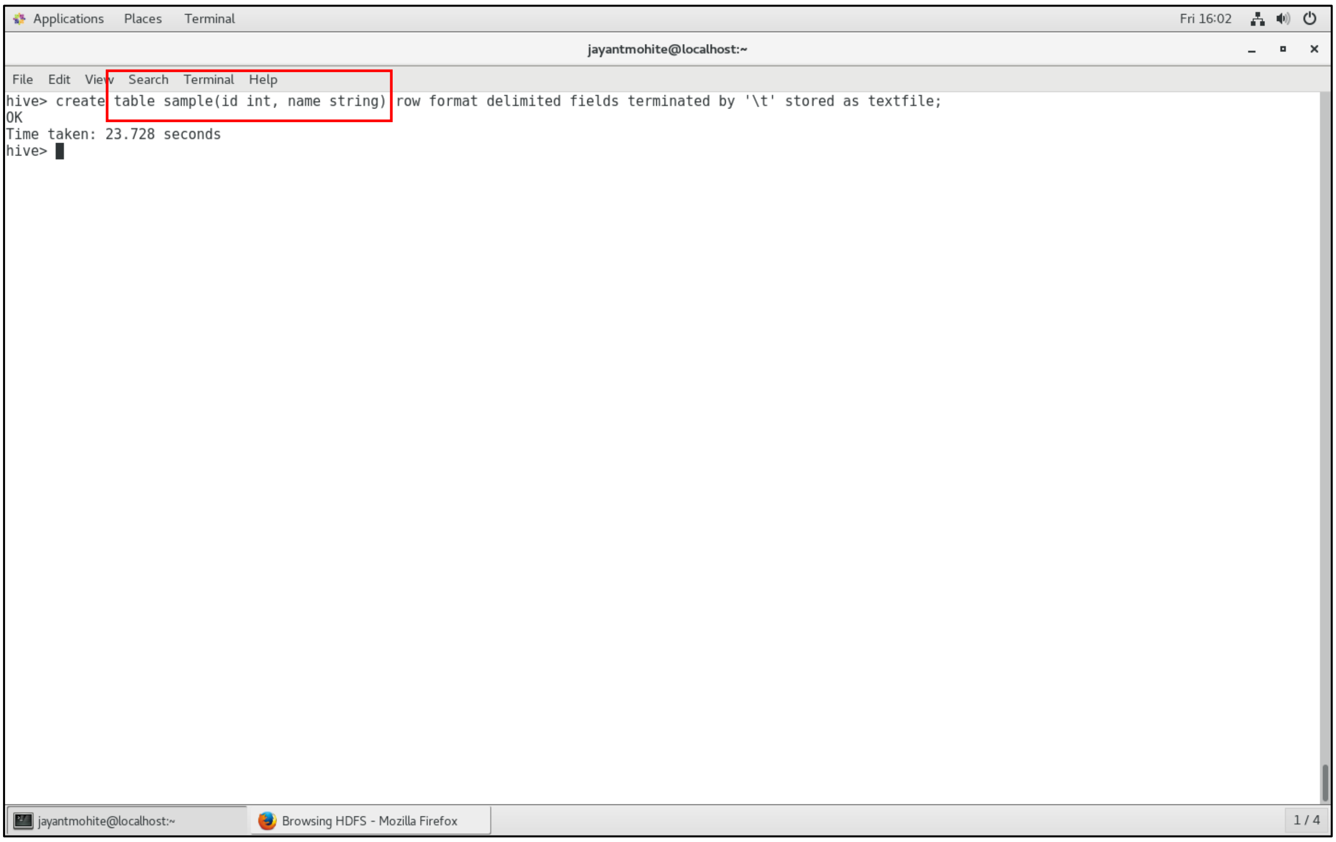
Browse Directory

/user/jayantmohite/hive_input Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	jayantmohite	supergroup	12 B	Fri Mar 02 16:00:32 -0800 2018	1	128 MB	jayant_file.txt

Hadoop, 2017.

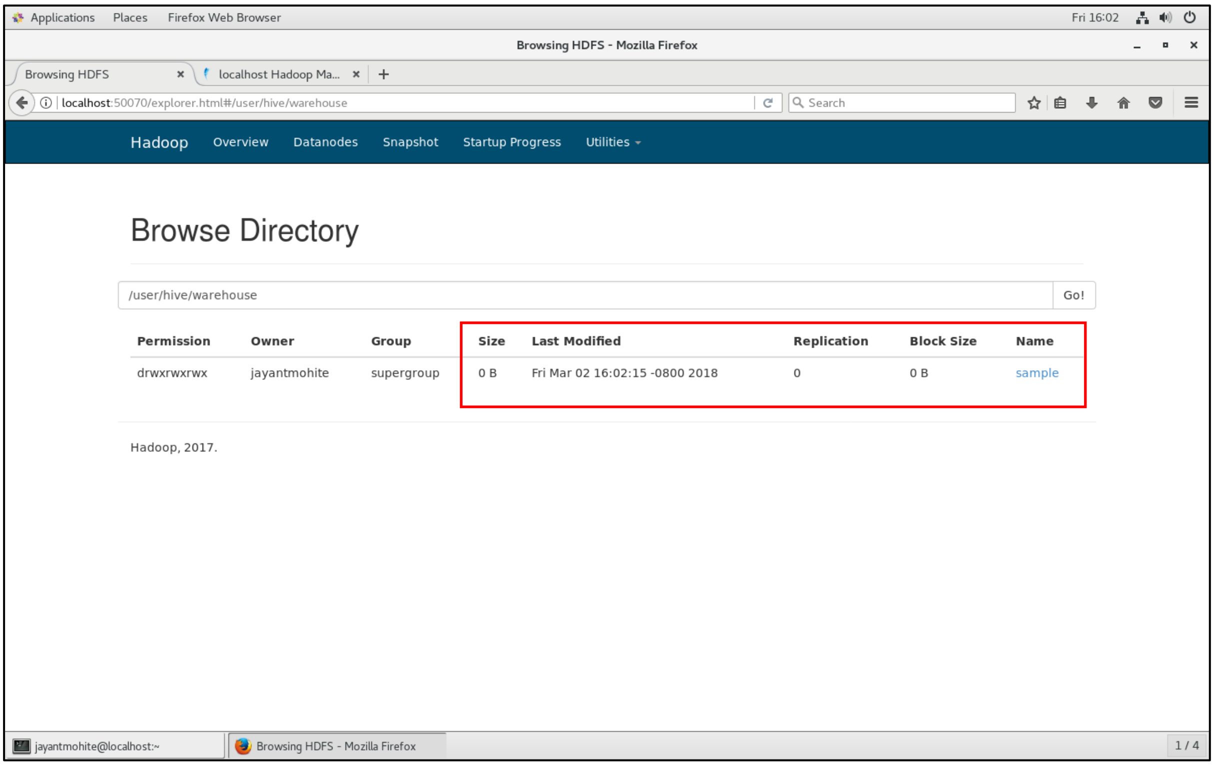
We will now enter our hive shell and create a table as we would do in any RDBMS.


Applications Places Terminal Fri 16:02
jayantmohite@localhost:~
File Edit View Search Terminal Help
jayantmohite@localhost:~
Browsing HDFS - Mozilla Firefox
1 / 4

```
hive> create table sample(id int, name string) row format delimited fields terminated by '\t' stored as textfile;
OK
Time taken: 23.728 seconds
hive> 
```

Author: Jayant Mohite

With this we expect that a table should be created but what actually happens is a directory by the name of the table is created in the warehouse.



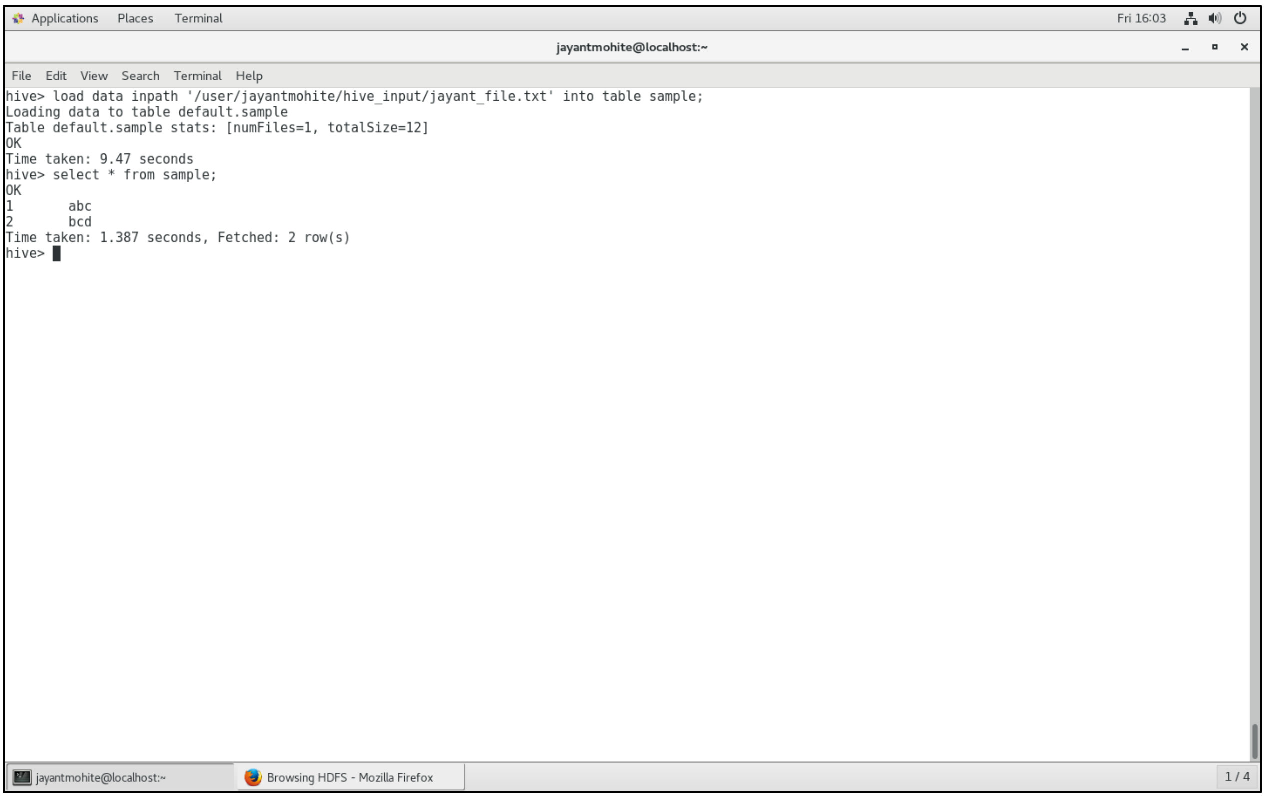
Browse Directory

/user/hive/warehouse

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxrwxrwx	jayantmohite	supergroup	0 B	Fri Mar 02 16:02:15 -0800 2018	0	0 B	sample

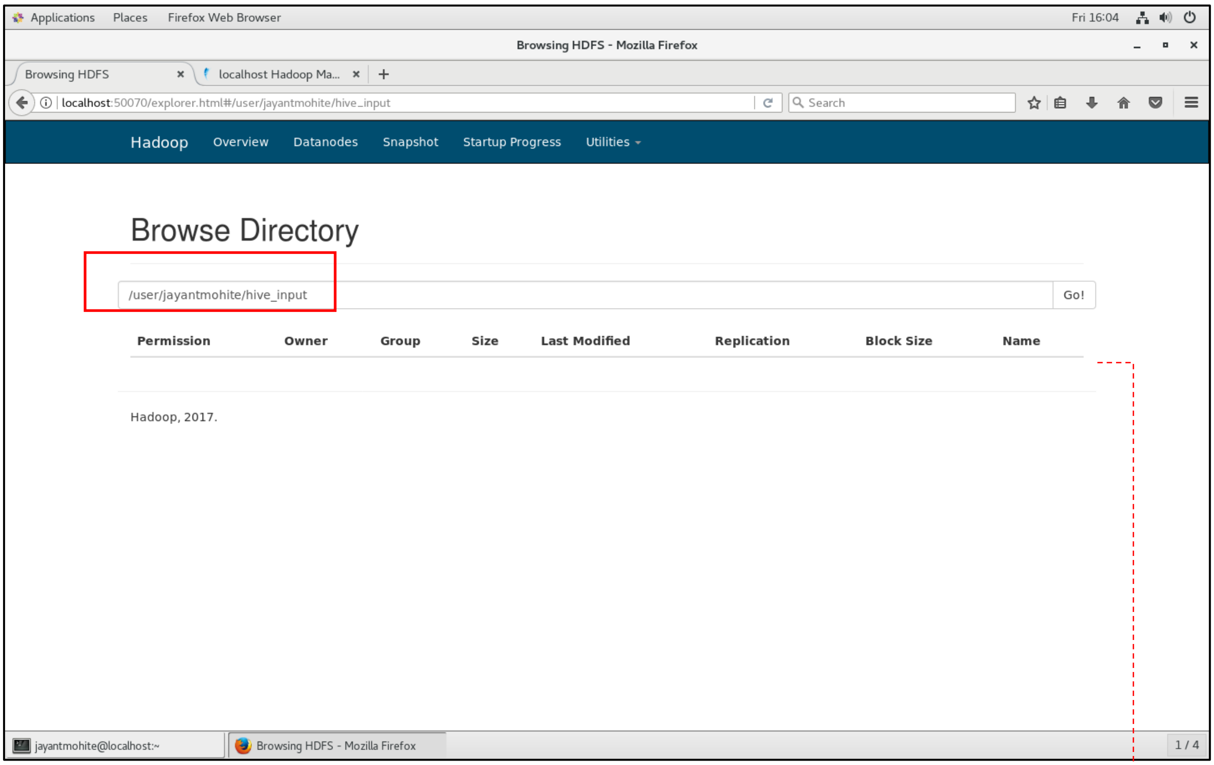
Hadoop, 2017.

You can observe that at this very point the directory created is empty. Now we will load data in our newly created table.



```
hive> load data inpath '/user/jayantmohite/hive_input/jayant_file.txt' into table sample;
Loading data to table default.sample
Table default.sample stats: [numFiles=1, totalSize=12]
OK
Time taken: 9.47 seconds
hive> select * from sample;
OK
1       abc
2       bcd
Time taken: 1.387 seconds, Fetched: 2 row(s)
hive>
```

But what actually happens is data is cut from the source location and is pasted in the hive warehouse location.

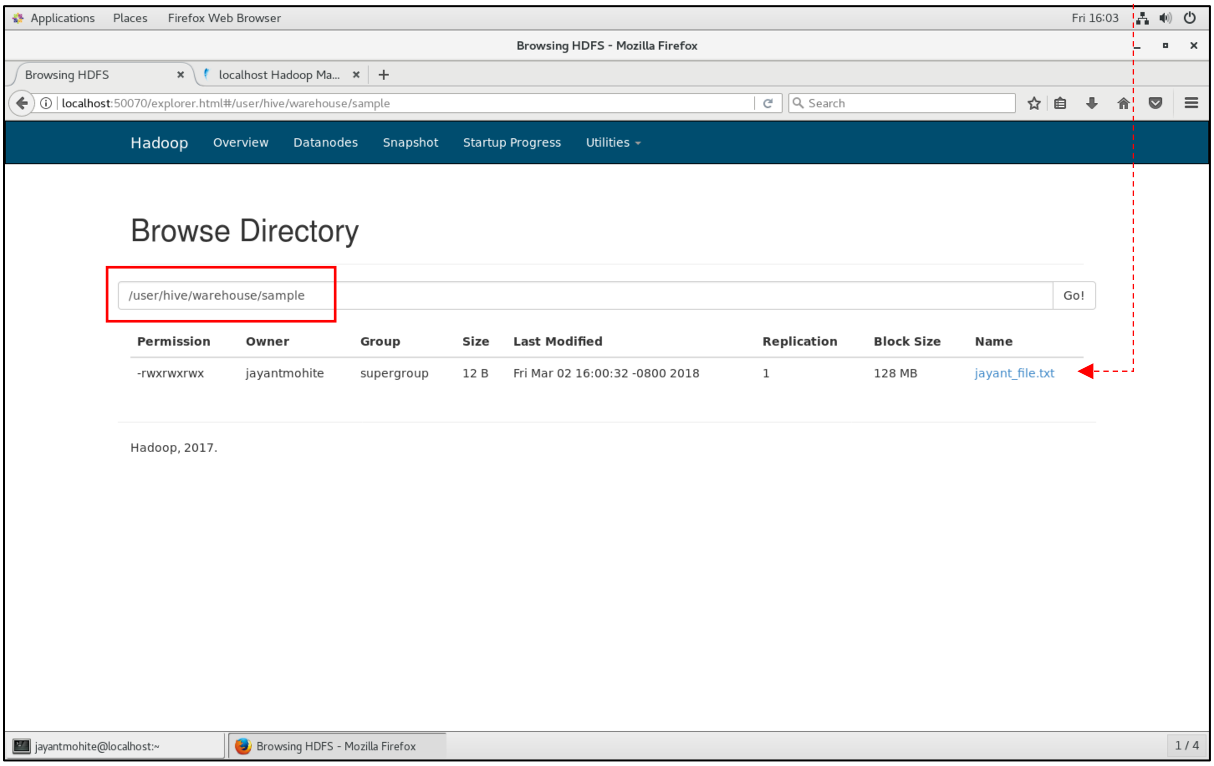


Browse Directory

/user/jayantmohite/hive_input

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name

Hadoop, 2017.



Browse Directory

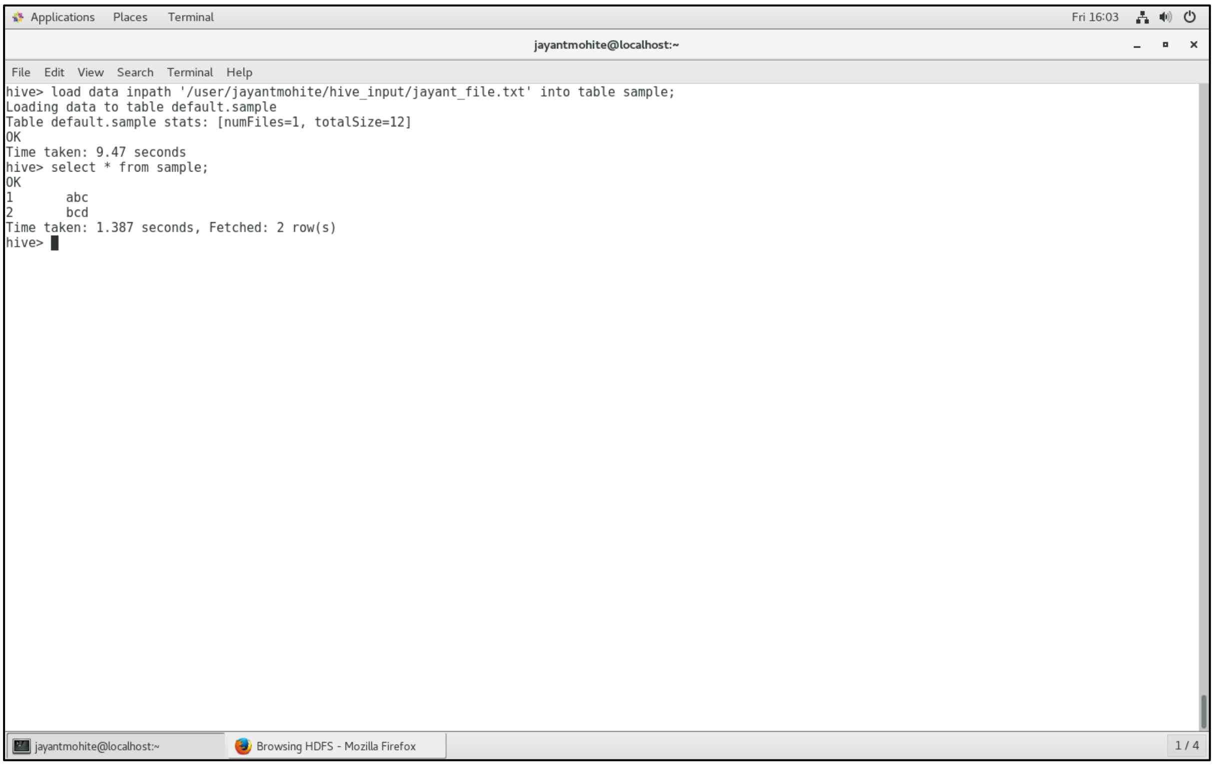
/user/hive/warehouse/sample

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rwxrwxrwx	jayantmohite	supergroup	12 B	Fri Mar 02 16:00:32 -0800 2018	1	128 MB	jayant_file.txt

Hadoop, 2017.

Author: Jayant Mohite

Now you can use the table and execute any SQL queries you wish. But what will actually happen at the backend is, when you execute a query on the table, a Map Reduce program will be executed on the directory in the warehouse location.



```
hive> load data inpath '/user/jayantmohite/hive_input/jayant_file.txt' into table sample;
Loading data to table default.sample
Table default.sample stats: [numFiles=1, totalSize=12]
OK
Time taken: 9.47 seconds
hive> select * from sample;
OK
1       abc
2       bcd
Time taken: 1.387 seconds, Fetched: 2 row(s)
hive> 
```

Commands:

hive> create table <enter table name as sample> (id int, name string) row format delimited fields terminated by '\t' stored as textfile;
(in this command, row format delimited fields terminated by '\t' refer to the delimiter options used in our input file. Hive supports multiple file formats for internally handling the storage of the table, one of which is textfile.)

hive> load data inpath <enter input data location as /user/jayantmohite/hive_input/jayant_file.txt> into table <enter table name as sample>;
(this will cut data from source location /user/jayantmohite/hive_input and will paste it in destination location /user/hive/warehouse/sample)

Hive supports multiple file formats for internally handling the storage of the table's data. These file formats are

- Text File
- Sequence File
- RCFile
- ORCFile
- Avro Files
- Parquet

Now lets consider we have the following type of data

1	abc	100
2	bcd	200
3	cde	300
4	def	400
5	efg	500

Text File Representation

[1,abc,100]
[2,bcd,200]
[3,cde,300]
[4,def,400]
[5,efg,500]

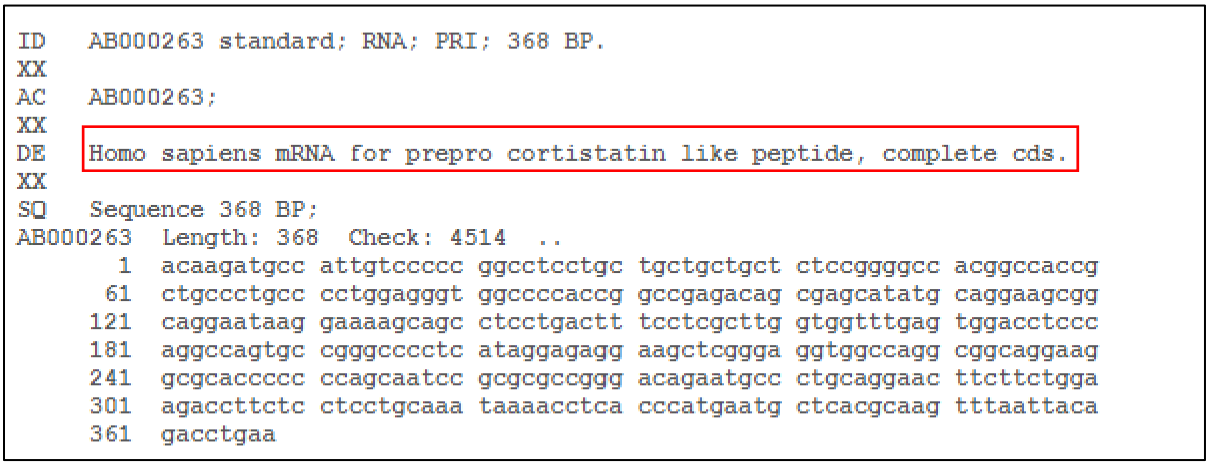
RCFile Representation

[1,2,3,4,5]
[abc, bcd, cde, def, efg]
[100, 200, 300, 400, 500]

Sequence File Representation

[1, abc, 100, 2, bcd, 200, 3, cde, 300, 4, def, 400, 5, efg, 500]

Sample Sequence File representation for more understanding

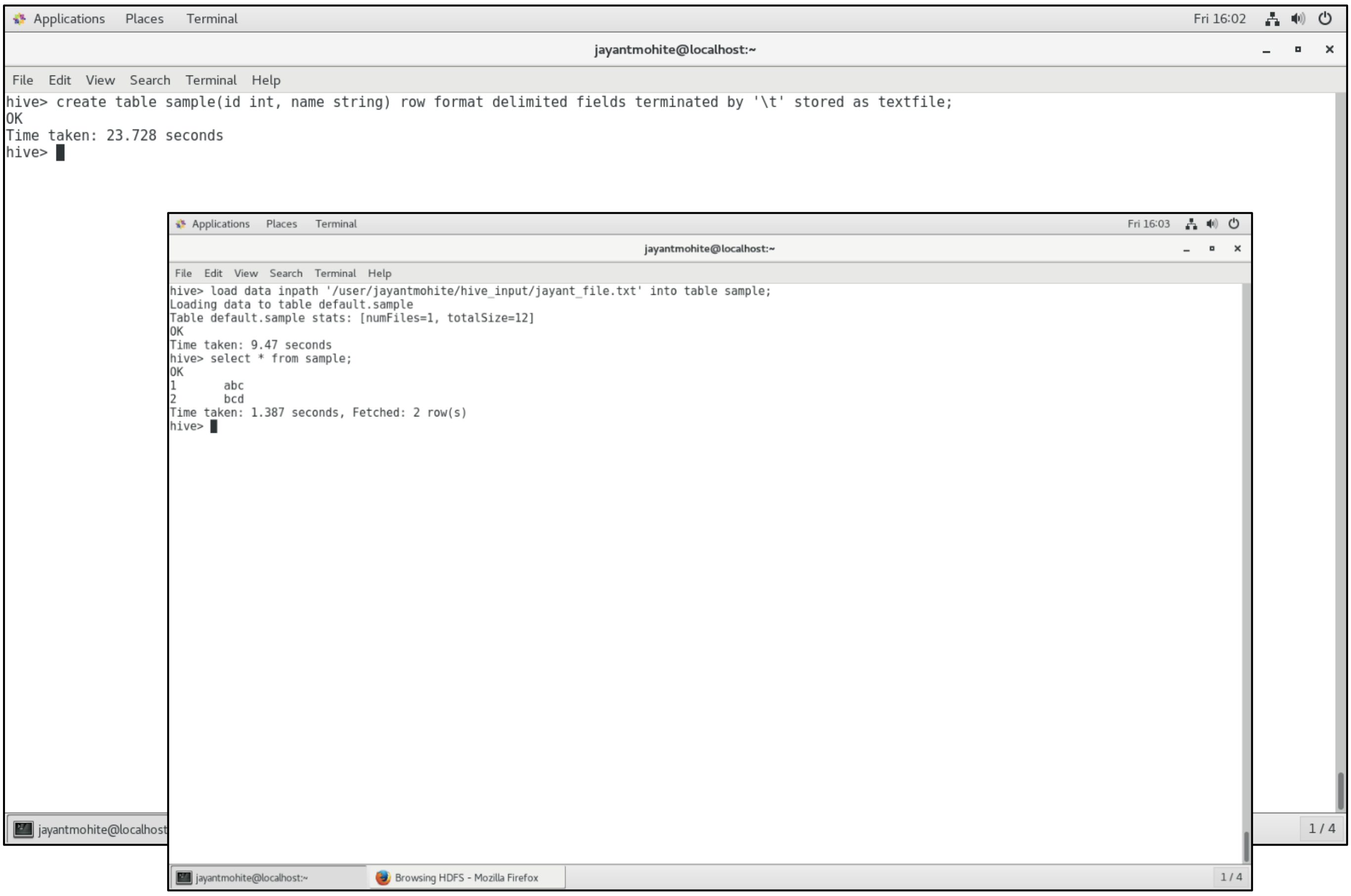


```
ID   AB000263 standard; RNA; PRI; 368 BP.
XX
AC   AB000263;
XX
DE   Homo sapiens mRNA for prepro cortistatin like peptide, complete cds.
XX
SQ   Sequence 368 BP;
AB000263  Length: 368  Check: 4514  ..
       1  acaagatgcc attgtccccc ggcctcctgc tgctgctgct ctccggggcc acggccaccg
      61  ctgccctgcc cctggagggt ggccccaccg gccgagacag cgagcatatg caggaagcgg
     121  caggaataag gaaaagcagc ctcctgactt tcctcgcttg gtggtttgag tggacctccc
     181  aggccagtgc cgggcccctc ataggagagg aagctcggga ggtggccagg cggcaggaag
     241  gcgcaccccc ccagcaatcc gcgcgccggg acagaatgcc ctgcaggaac ttcttctgga
     301  agaccttctc ctcctgcaaa taaaacctca cccatgaatg ctcacgcaag tttaattaca
     361  gacctgaa
```

Working with textfile format

hive> create table sample(id int, name string) row format delimited fields terminated by '\t' stored as textfile;

hive> load data inpath '/user/jayantmohite/hive_input/jayant_file.txt' into table sample



Applications Places Terminal
Fri 16:02
jayantmohite@localhost:~
File Edit View Search Terminal Help
jayantmohite@localhost
1 / 4
Fri 16:03
Browsing HDFS - Mozilla Firefox

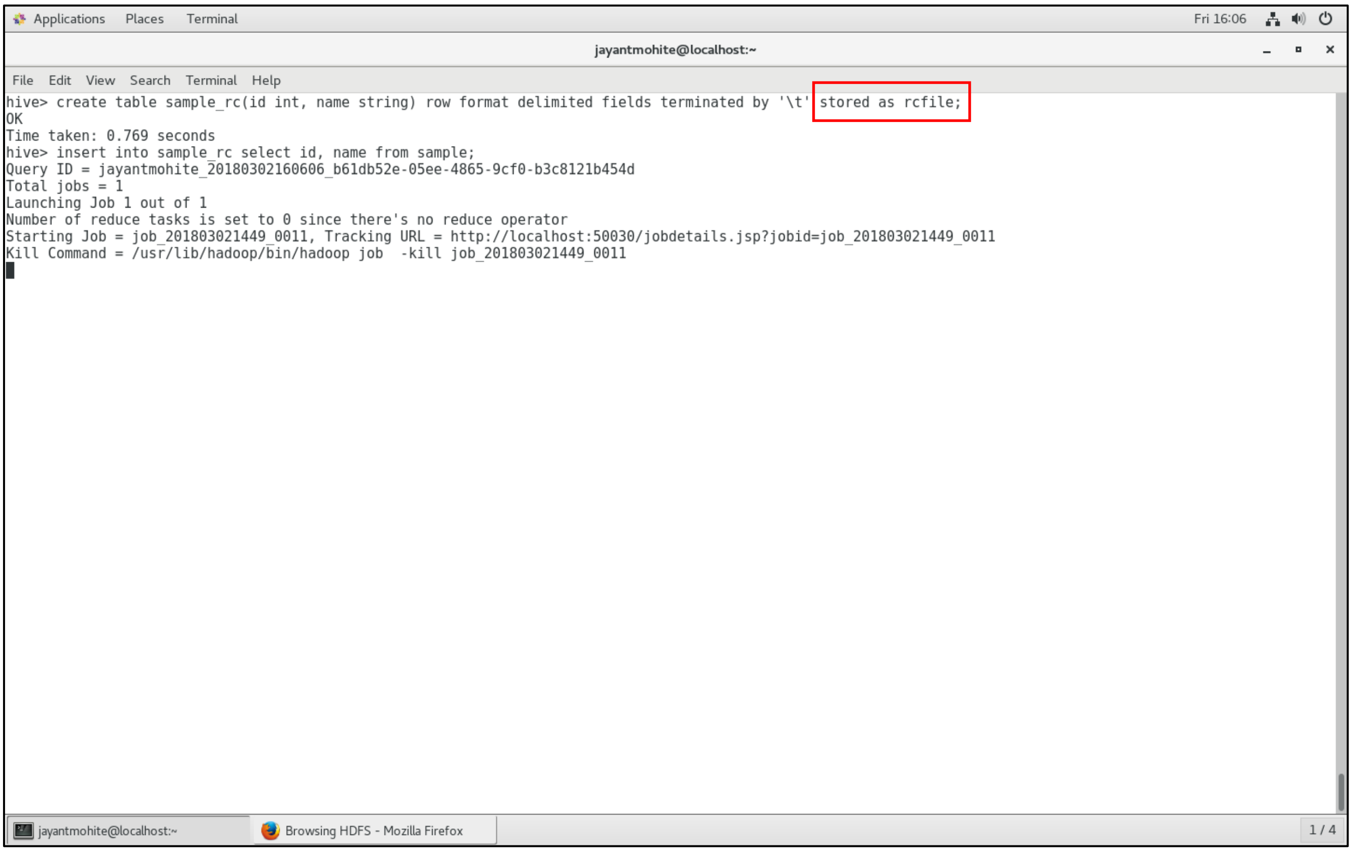
```
hive> create table sample(id int, name string) row format delimited fields terminated by '\t' stored as textfile;
OK
Time taken: 23.728 seconds
hive> 
```

```
hive> load data inpath '/user/jayantmohite/hive_input/jayant_file.txt' into table sample;
Loading data to table default.sample
Table default.sample stats: [numFiles=1, totalSize=12]
OK
Time taken: 9.47 seconds
hive> select * from sample;
OK
1       abc
2       bcd
Time taken: 1.387 seconds, Fetched: 2 row(s)
hive> 
```

Author: Jayant Mohite

Working with other file formats is slightly different than working with the textfile format. You cannot load data directly from a input file into these tables. You need to have a textfile format table and then you can load data from this table to your table of any other file format.

Example RCFile Table



```
hive> create table sample_rc(id int, name string) row format delimited fields terminated by '\t' stored as rcfile;
OK
Time taken: 0.769 seconds
hive> insert into sample_rc select id, name from sample;
Query ID = jayantmohite_20180302160606_b61db52e-05ee-4865-9cf0-b3c8121b454d
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201803021449_0011, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201803021449_0011
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_201803021449_0011
```

Commands:

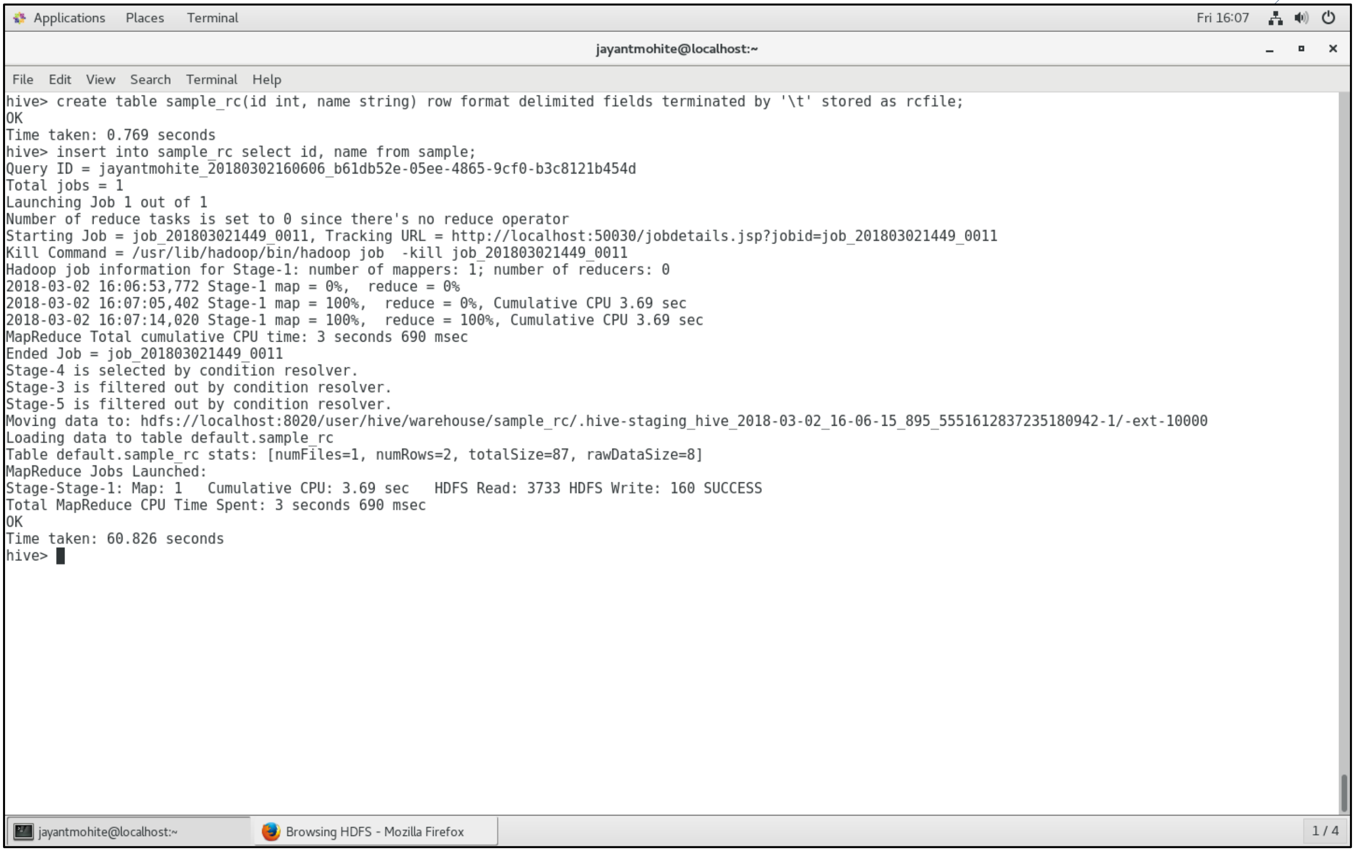
hive> create table sample_rc(id int, name string) row format delimited fields terminated by '\t' stored as RCFile;

(do note that everything else in the syntax is same. What has changed is only the file format.)

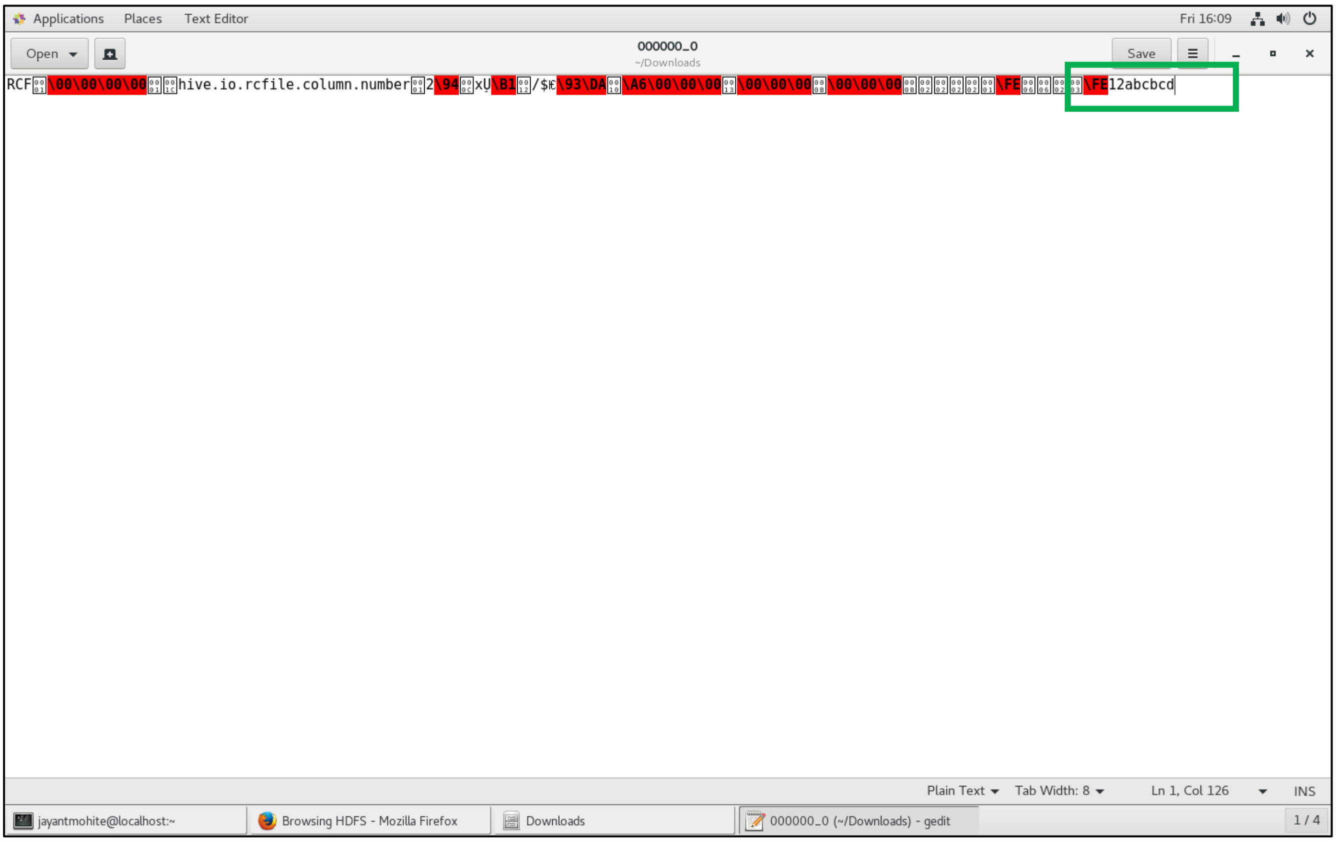
hive> insert into table <enter your RCFile table name as sample_rc> select id, name from <enter your textfile table name as sample>;

(this will load data from the textfile table into the newly created RCFile table)

Author: Jayant Mohite



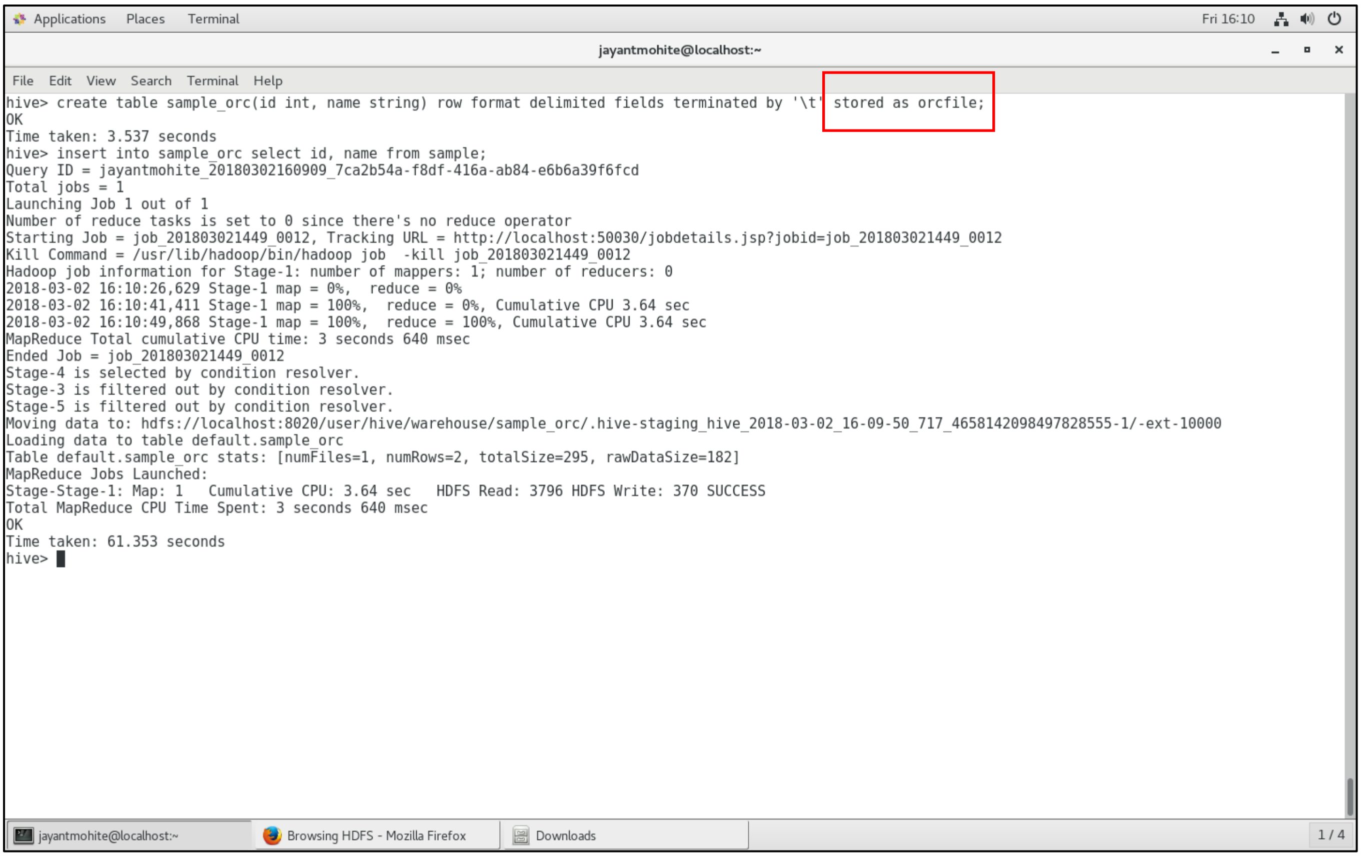
```
hive> create table sample_rc(id int, name string) row format delimited fields terminated by '\t' stored as rcfile;
OK
Time taken: 0.769 seconds
hive> insert into sample_rc select id, name from sample;
Query ID = jayantmohite_20180302160606_b61db52e-05ee-4865-9cf0-b3c8121b454d
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201803021449_0011, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201803021449_0011
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_201803021449_0011
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2018-03-02 16:06:53,772 Stage-1 map = 0%,  reduce = 0%
2018-03-02 16:07:05,402 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.69 sec
2018-03-02 16:07:14,020 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 3.69 sec
MapReduce Total cumulative CPU time: 3 seconds 690 msec
Ended Job = job_201803021449_0011
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://localhost:8020/user/hive/warehouse/sample_rc/.hive-staging_hive_2018-03-02_16-06-15_895_5551612837235180942-1/-ext-10000
Loading data to table default.sample_rc
Table default.sample_rc stats: [numFiles=1, numRows=2, totalSize=87, rawDataSize=8]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 3.69 sec   HDFS Read: 3733 HDFS Write: 160 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 690 msec
OK
Time taken: 60.826 seconds
hive> 
```



```
RCF▯\00\00\00\00▯▯hive.io.rcfile.column.number▯2\94▯xU\B1▯/$€\93\DA▯\A6\00\00\00▯\00\00\00▯\00\00\00▯▯▯▯▯\FE▯▯▯▯\FE12abcbcd
```

Author: Jayant Mohite

Example ORCFile table



```
hive> create table sample_orc(id int, name string) row format delimited fields terminated by '\t' stored as orcfile;
OK
Time taken: 3.537 seconds
hive> insert into sample_orc select id, name from sample;
Query ID = jayantmohite_20180302160909_7ca2b54a-f8df-416a-ab84-e6b6a39f6fcd
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201803021449_0012, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201803021449_0012
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_201803021449_0012
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2018-03-02 16:10:26,629 Stage-1 map = 0%,  reduce = 0%
2018-03-02 16:10:41,411 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.64 sec
2018-03-02 16:10:49,868 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 3.64 sec
MapReduce Total cumulative CPU time: 3 seconds 640 msec
Ended Job = job_201803021449_0012
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://localhost:8020/user/hive/warehouse/sample_orc/.hive-staging_hive_2018-03-02_16-09-50_717_4658142098497828555-1/-ext-10000
Loading data to table default.sample_orc
Table default.sample_orc stats: [numFiles=1, numRows=2, totalSize=295, rawDataSize=182]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 3.64 sec   HDFS Read: 3796 HDFS Write: 370 SUCCESS
Total MapReduce CPU Time Spent: 3 seconds 640 msec
OK
Time taken: 61.353 seconds
hive>
```

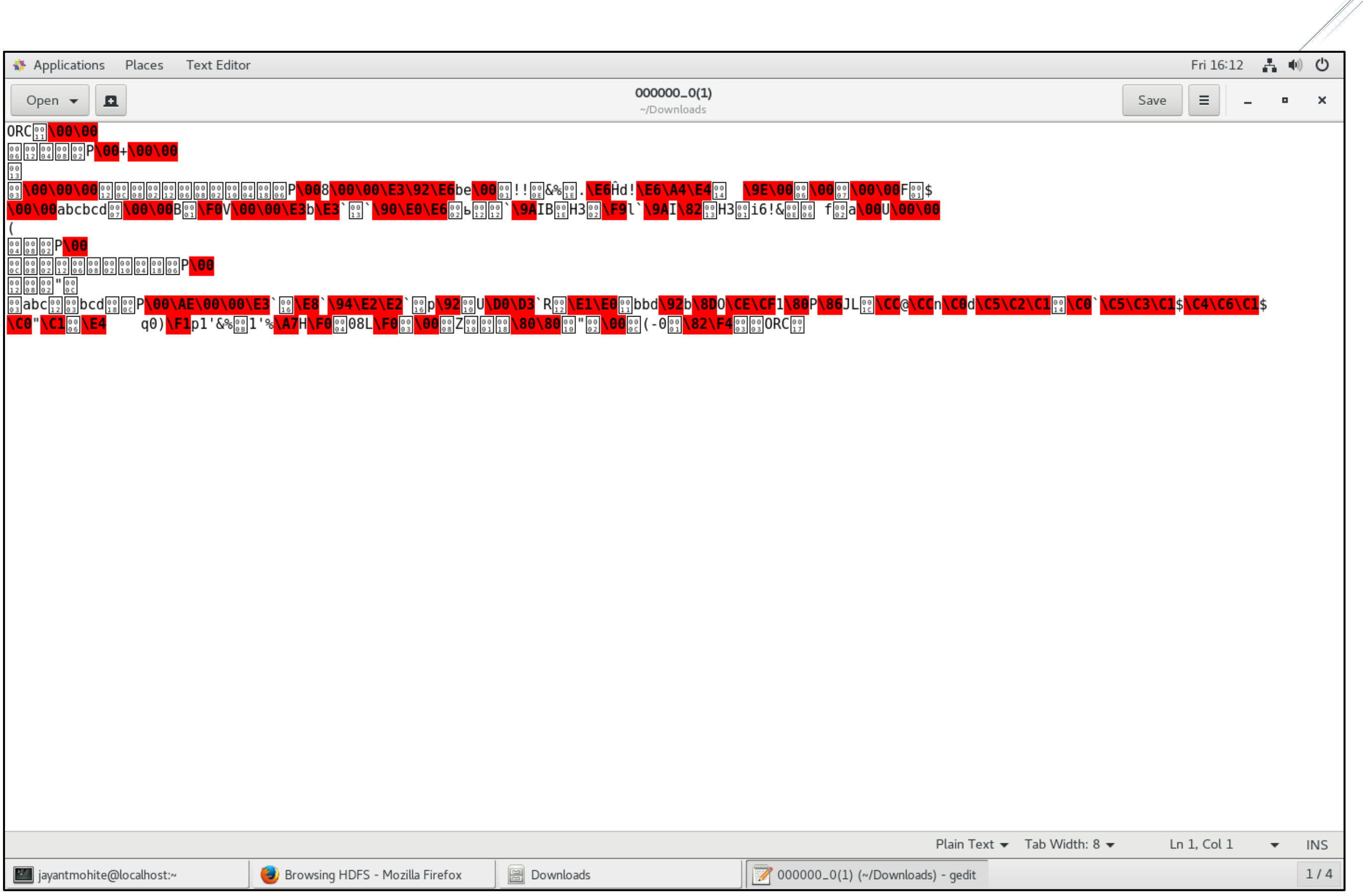
Commands:

hive> create table sample_orc(id int, name string) row format delimited fields terminated by ’\t’ stored as ORCFile;

(do note that everything else in the syntax is same. What has changed is only the file format.)

hive> insert into table <enter your ORCFile table name as sample_orc> select id, name from <enter your textfile table name as sample>;

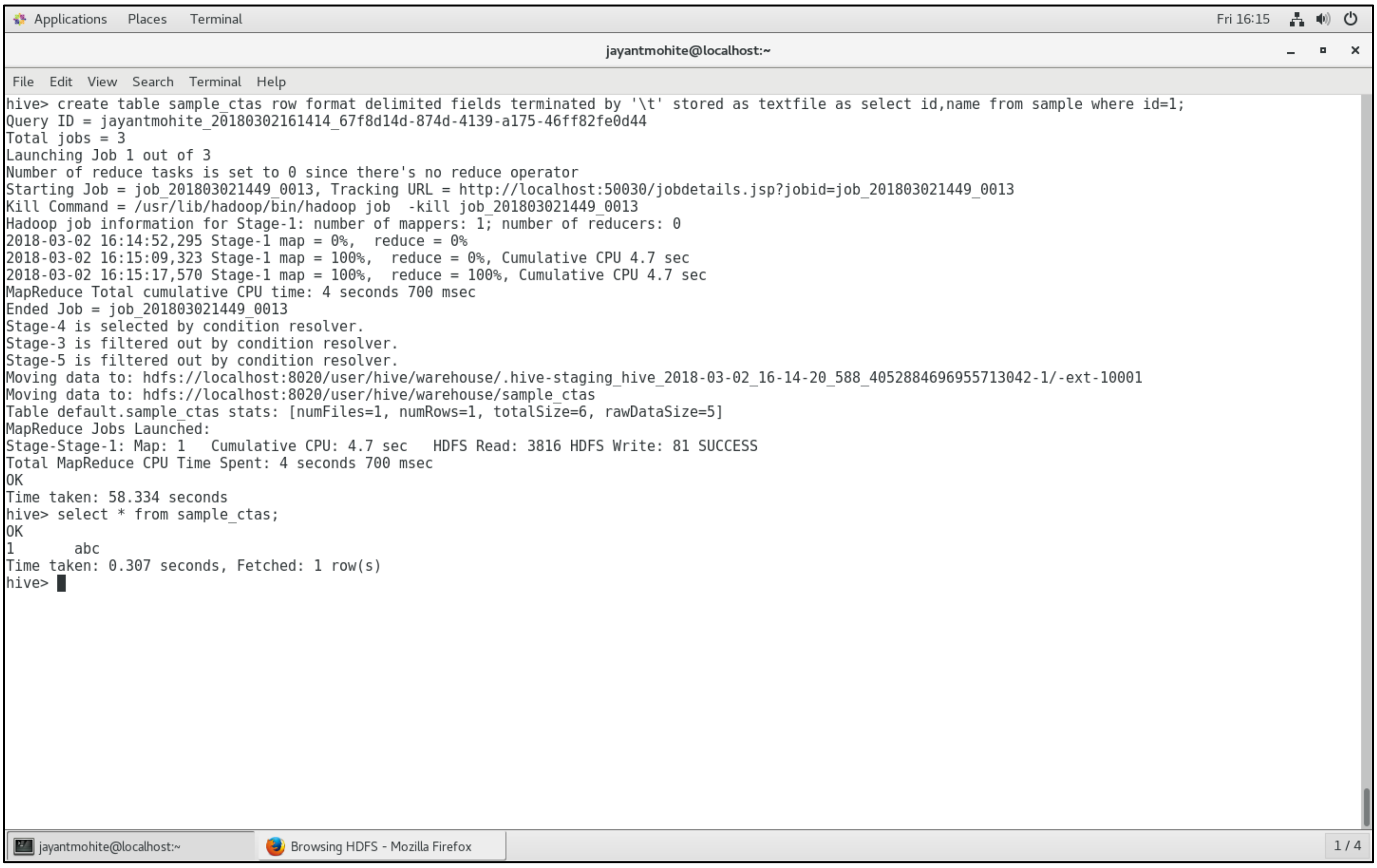
(this will load data from the textfile table into the newly created ORCFile table)



```
ORC□\00\00
□□□□□P\00+\00\00
□
□\00\00\00□□□□□□□□□□□□P\008\00\00\E3\92\E6be\00□!!□&%□.\E6Ĥd!\E6\A4\E4□  \9E\00□\00□\00\00F□$
\00\00abcbcd□\00\00B□\F0V\00\00\E3b\E3`□`\90\E0\E6□ь□□`\9AIB□H3□\F9l`\9AI\82□H3□i6!&□□ f□a\00U\00\00
(
□□□P\00
□□□□□□□□□□□P\00
□□□"□
□abc□□bcd□□P\00\AE\00\00\E3`□\E8`\94\E2\E2`□p\92□U\D0\D3`R□\E1\E0□bbd\92b\8DO\CE\CF1\80P\86JL□\CC@\CCn\C0d\C5\C2\C1□\C0`\C5\C3\C1$\C4\C6\C1$
\C0"\C1□\E4    q0)\F1p1'&%□1'%\A7H\F0□08L\F0□\00□Z□□□\80\80□"□\00□(-0□\82\F4□□ORC□
```

Example of Create Table As Select

In this kind of operation we will be creating a new table from the result set of a query



```
hive> create table sample_ctas row format delimited fields terminated by '\t' stored as textfile as select id,name from sample where id=1;
Query ID = jayantmohite_20180302161414_67f8d14d-874d-4139-a175-46ff82fe0d44
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201803021449_0013, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201803021449_0013
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_201803021449_0013
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2018-03-02 16:14:52,295 Stage-1 map = 0%,  reduce = 0%
2018-03-02 16:15:09,323 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 4.7 sec
2018-03-02 16:15:17,570 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.7 sec
MapReduce Total cumulative CPU time: 4 seconds 700 msec
Ended Job = job_201803021449_0013
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://localhost:8020/user/hive/warehouse/.hive-staging_hive_2018-03-02_16-14-20_588_4052884696955713042-1/-ext-10001
Moving data to: hdfs://localhost:8020/user/hive/warehouse/sample_ctas
Table default.sample_ctas stats: [numFiles=1, numRows=1, totalSize=6, rawDataSize=5]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 4.7 sec   HDFS Read: 3816 HDFS Write: 81 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 700 msec
OK
Time taken: 58.334 seconds
hive> select * from sample_ctas;
OK
1       abc
Time taken: 0.307 seconds, Fetched: 1 row(s)
hive> █
```

Commands:

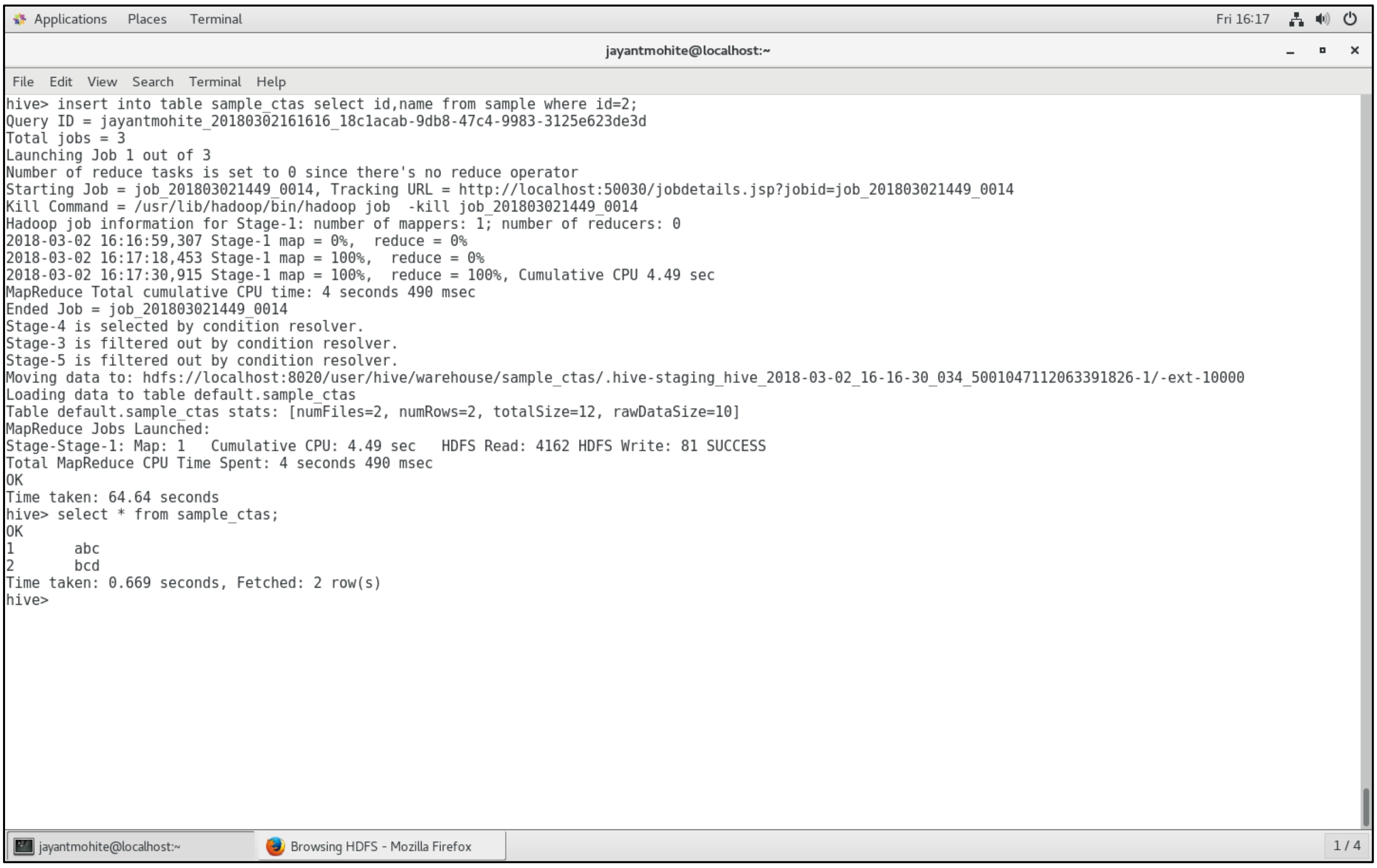
hive> create table sample_ctas row format delimited fileds terminated by '\t' stored as textfile as select id, name from sample where id = 1;

In this command the query select id, name from sample where id = 1 will be execute first. The schema that this query returns will become the schema of the new table and the data that this query returns will become the data of the new table.

In this command both create table and load data commands are combined together.

Example of Hive Insert Into

In this example, we will append some data to an existing table from another existing table.



Applications Places Terminal
Fri 16:17
jayantmohite@localhost:~
File Edit View Search Terminal Help
jayantmohite@localhost:~
Browsing HDFS - Mozilla Firefox
1 / 4

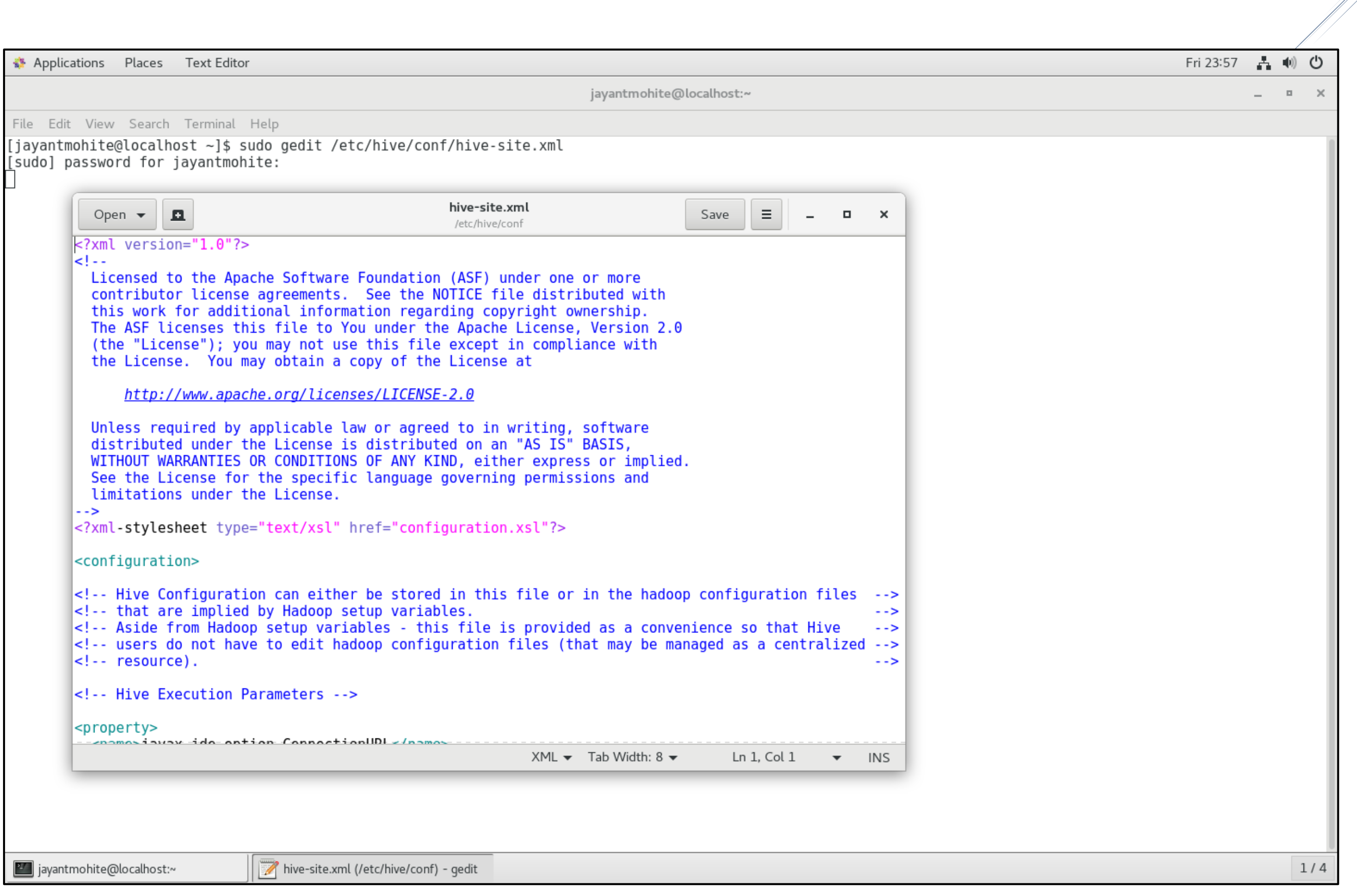
```
hive> insert into table sample_ctas select id,name from sample where id=2;
Query ID = jayantmohite_20180302161616_18c1acab-9db8-47c4-9983-3125e623de3d
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201803021449_0014, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201803021449_0014
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_201803021449_0014
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2018-03-02 16:16:59,307 Stage-1 map = 0%,  reduce = 0%
2018-03-02 16:17:18,453 Stage-1 map = 100%,  reduce = 0%
2018-03-02 16:17:30,915 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.49 sec
MapReduce Total cumulative CPU time: 4 seconds 490 msec
Ended Job = job_201803021449_0014
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://localhost:8020/user/hive/warehouse/sample_ctas/.hive-staging_hive_2018-03-02_16-16-30_034_5001047112063391826-1/-ext-10000
Loading data to table default.sample_ctas
Table default.sample_ctas stats: [numFiles=2, numRows=2, totalSize=12, rawDataSize=10]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 4.49 sec   HDFS Read: 4162 HDFS Write: 81 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 490 msec
OK
Time taken: 64.64 seconds
hive> select * from sample_ctas;
OK
1       abc
2       bcd
Time taken: 0.669 seconds, Fetched: 2 row(s)
hive>
```

Commands:

hive> insert into table sample_ctas select id, name from sample where id = 2;

In this command the query select id, name from sample where id = 2 will be executed first. The schema of the target table should match the schema of the query result to avoid bad data. The data returned by the query will be appended to the existing data in the target table.

Now before moving into other type of operations in Hive, lets define some properties in the hive configuration file named hive-site.xml located at location /etc/hive/conf/hive-site.xml. Also lets create some input files that we will use in the upcoming examples.

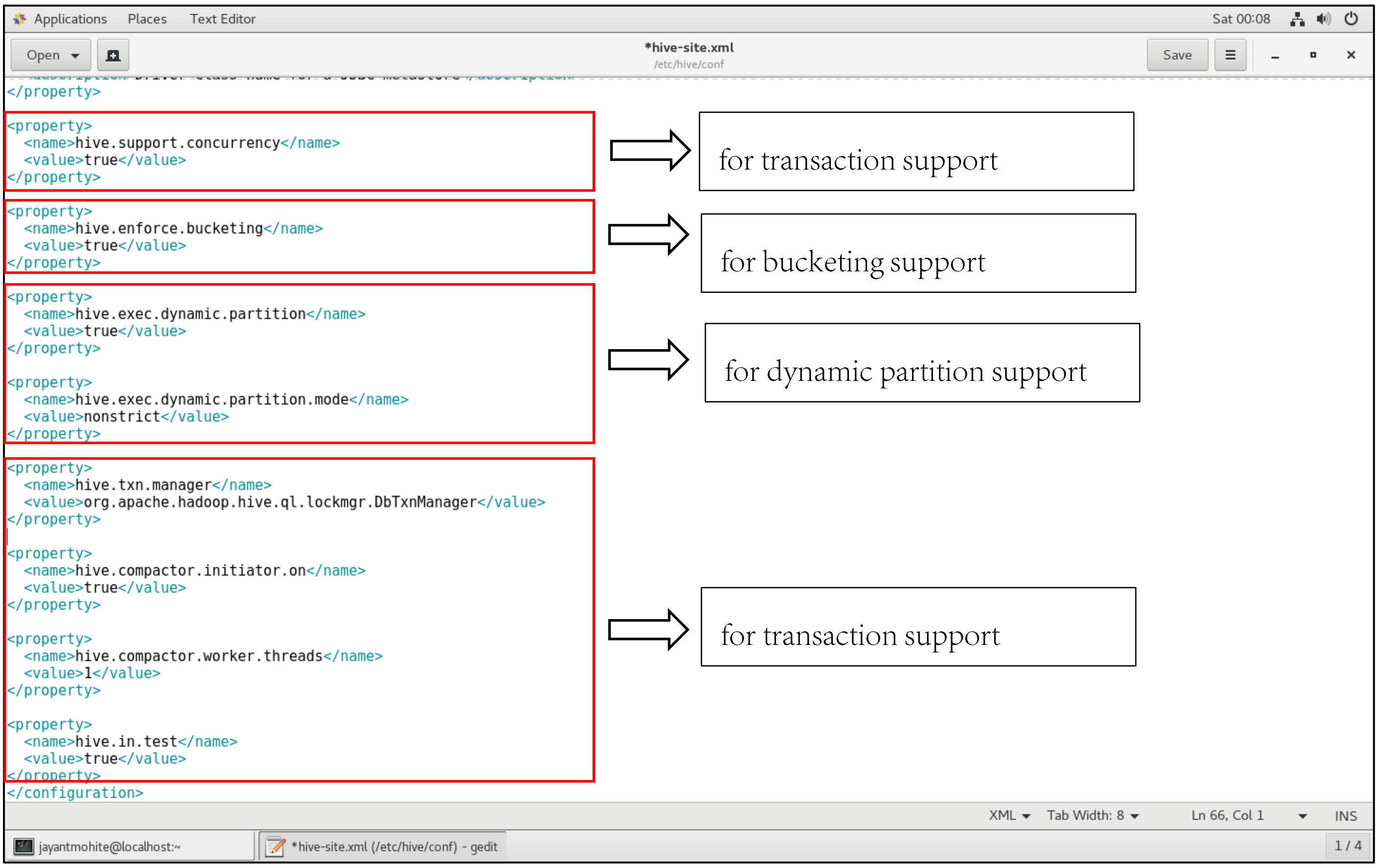


Applications Places Text Editor
Fri 23:57
jayantmohite@localhost:~
File Edit View Search Terminal Help

```
[jayantmohite@localhost ~]$ sudo gedit /etc/hive/conf/hive-site.xml
[sudo] password for jayantmohite:
```

```
<?xml version="1.0"?>
<!--
  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.
-->
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>

<!-- Hive Configuration can either be stored in this file or in the hadoop configuration files  -->
<!-- that are implied by Hadoop setup variables.                                                -->
<!-- Aside from Hadoop setup variables - this file is provided as a convenience so that Hive    -->
<!-- users do not have to edit hadoop configuration files (that may be managed as a centralized -->
<!-- resource).                                                                                 -->

<!-- Hive Execution Parameters -->

<property>
  <name>javax.jdo.option.ConnectionURL</name>
```

XML Tab Width: 8 Ln 1, Col 1 INS
jayantmohite@localhost:~
hive-site.xml (/etc/hive/conf) - gedit
1 / 4



Applications Places Text Editor
Sat 00:08
Open
*hive-site.xml
/etc/hive/conf
Save

```
</property>

<property>
  <name>hive.support.concurrency</name>
  <value>true</value>
</property>

<property>
  <name>hive.enforce.bucketing</name>
  <value>true</value>
</property>

<property>
  <name>hive.exec.dynamic.partition</name>
  <value>true</value>
</property>

<property>
  <name>hive.exec.dynamic.partition.mode</name>
  <value>nonstrict</value>
</property>

<property>
  <name>hive.txn.manager</name>
  <value>org.apache.hadoop.hive.ql.lockmgr.DbTxnManager</value>
</property>

<property>
  <name>hive.compactor.initiator.on</name>
  <value>true</value>
</property>

<property>
  <name>hive.compactor.worker.threads</name>
  <value>1</value>
</property>

<property>
  <name>hive.in.test</name>
  <value>true</value>
</property>
</configuration>
```

for transaction support

for bucketing support

for dynamic partition support

for transaction support

XML Tab Width: 8 Ln 66, Col 1 INS
jayantmohite@localhost:~
*hive-site.xml (/etc/hive/conf) - gedit
1 / 4

We will be creating 3 input files with almost same data. Major difference between these file will be that one of them will have four columns including one column as the country and other two won't have the country column.

For assumption we consider that we have collected one data set from a global domain so we need have the country column into it and rest we have collected from our India and US servers so we know the country and hence that column is not required.

In hive the Map Reduce program is executed on the entire directory of the table. Lets assume that there are 10 file with total of 10 million records. Out of which only 100 records belong to India. If we try to fire a query over this table, still it will query all 10 files as at the backend what happens is simply a Map Reduce program on the entire directory. So the time required for this operation will not be optimized. But what if we are able to created sub directories under our table directory and every sub directory contains data of distinct countries. In this case whenever we execute a query with a where clause on the country column, the operation will be more optimized as the operation will execute on the countries specific sub directory rather than executing on the complete directory. This is called as Partitioning.

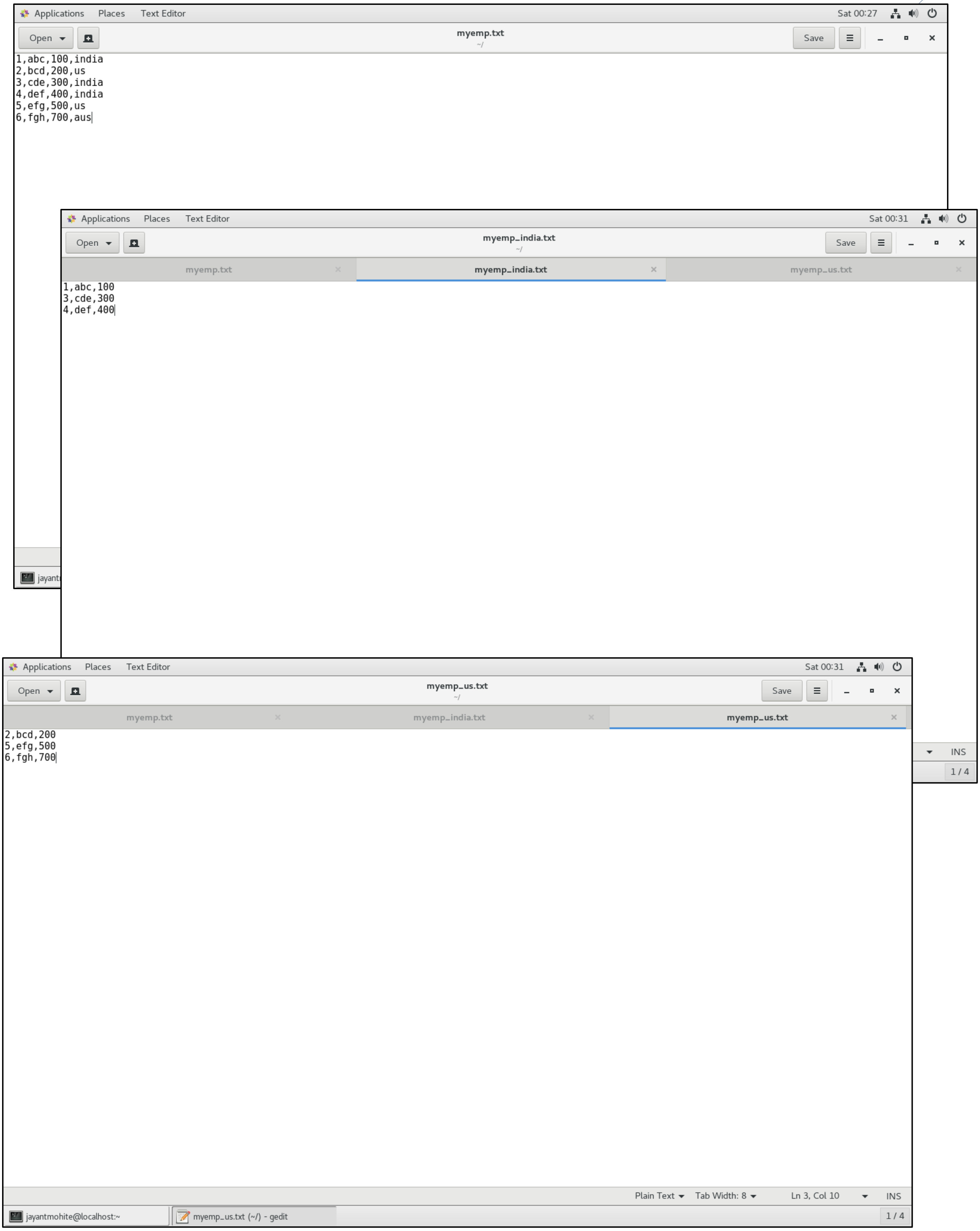
But there can be two cases in which we can get the data.

Case 1: The value of the partition column is available in the data itself. As in case of our first input file with the country column.

In this case we switch to a type of partitioning which automatically identifies the partitions and is called as Dynamic Partitioning

Case 2: The value of the partition column is not available in the data but we have separate inputs for each value as in case of our remaining two input files.

In this case we switch to a type of partitioning in which we manually specify the value of each partition and this is called as static partitioning.

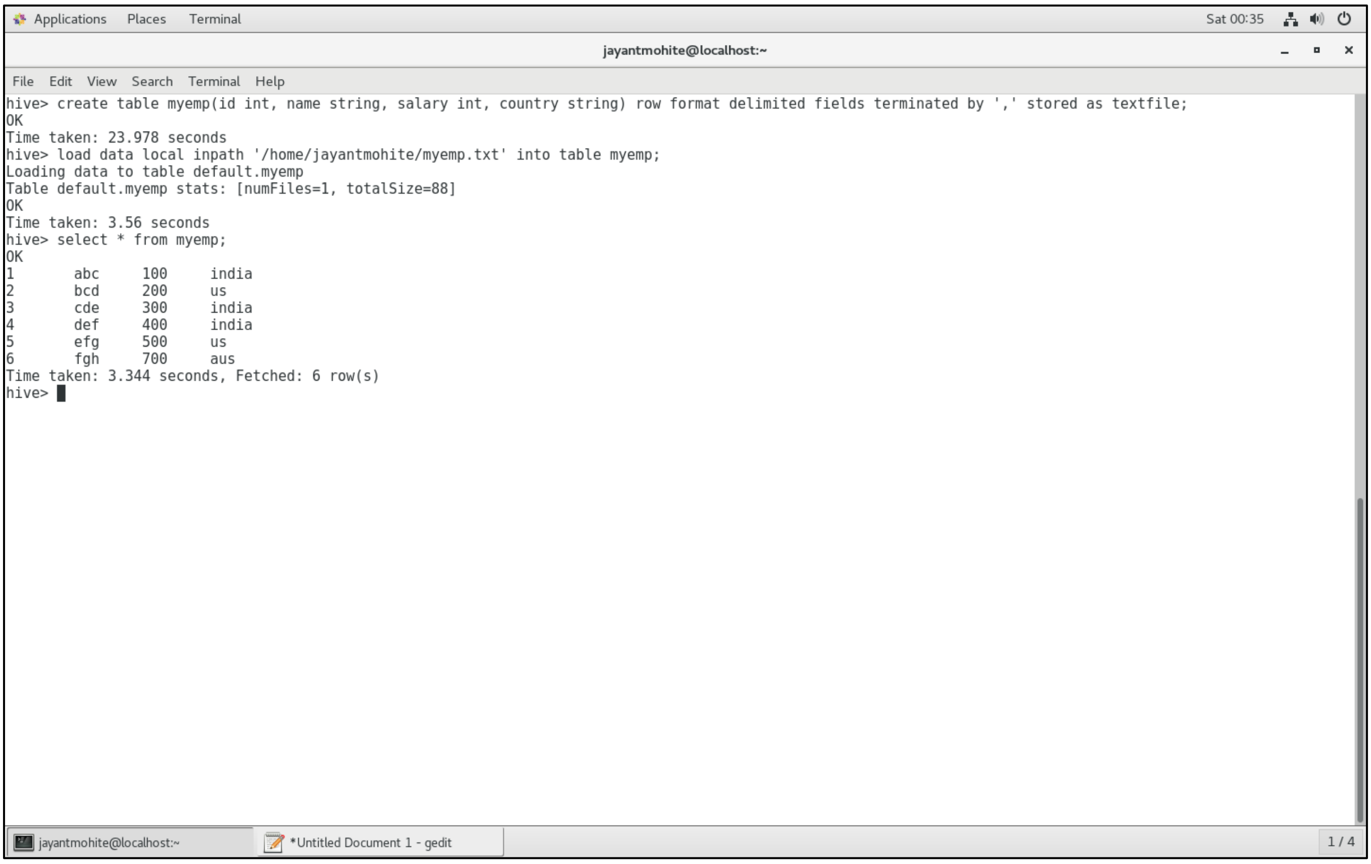


```
1,abc,100,india
2,bcd,200,us
3,cde,300,india
4,def,400,india
5,efg,500,us
6,fgh,700,aus
```

```
1,abc,100
3,cde,300
4,def,400
```

```
2,bcd,200
5,efg,500
6,fgh,700
```

Now lets create our primary table.



```
hive> create table myemp(id int, name string, salary int, country string) row format delimited fields terminated by ',' stored as textfile;
OK
Time taken: 23.978 seconds
hive> load data local inpath '/home/jayantmohite/myemp.txt' into table myemp;
Loading data to table default.myemp
Table default.myemp stats: [numFiles=1, totalSize=88]
OK
Time taken: 3.56 seconds
hive> select * from myemp;
OK
1       abc     100     india
2       bcd     200     us
3       cde     300     india
4       def     400     india
5       efg     500     us
6       fgh     700     aus
Time taken: 3.344 seconds, Fetched: 6 row(s)
hive> 
```

Command for creating a partitioned table

hive> create table myemp_partitioned(id int, name string, salary int) partitioned by (country string) row format delimited fields terminated by ’,’ stored as textfile;

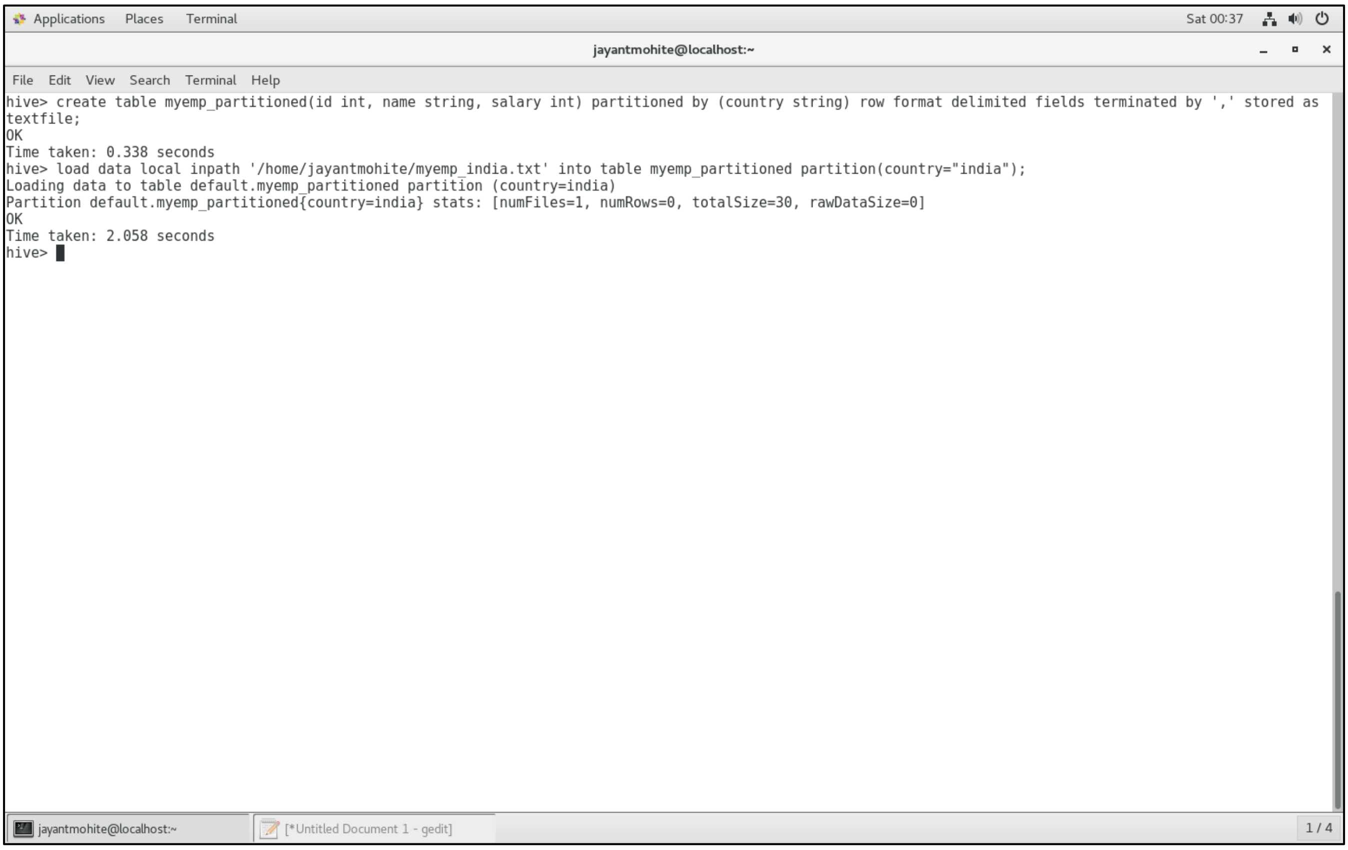
(In this command we are creating a table that has 4 columns and the country column is the one on which the table is partitioned)

Loading data in Static Partitioning

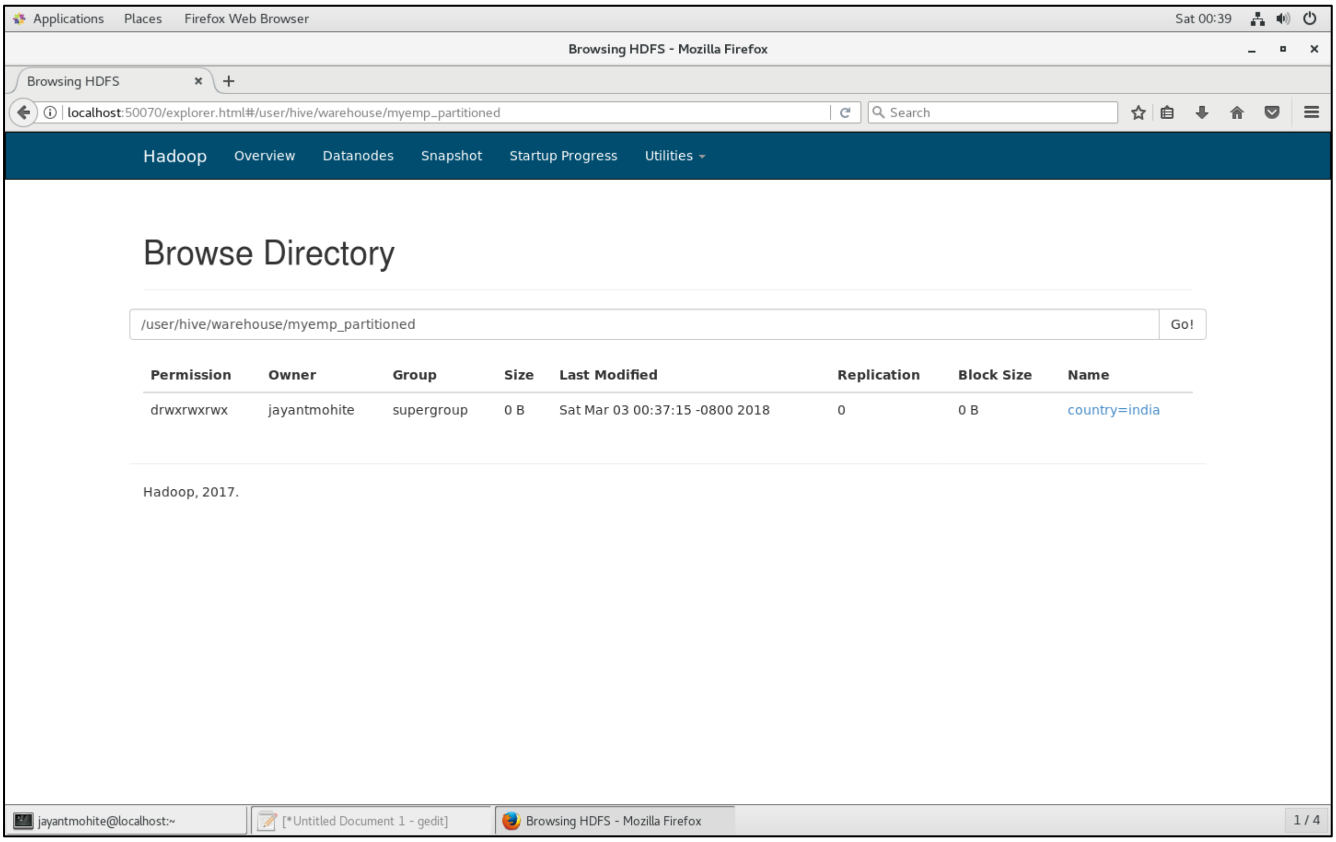
hive> load data local inpath ’/home/jayantmohite/myemp_india.txt’ into table myemp_partitioned partition(country = "india");

(In this command we load data from a file which has 3 columns. The value of the 4th column which is the partition column is provided manually by us.)

hive> load data local inpath ’/home/jayantmohite/myemp_us.txt’ into table myemp_partitioned partition(country = "us");



```
hive> create table myemp_partitioned(id int, name string, salary int) partitioned by (country string) row format delimited fields terminated by ',' stored as
textfile;
OK
Time taken: 0.338 seconds
hive> load data local inpath '/home/jayantmohite/myemp_india.txt' into table myemp_partitioned partition(country="india");
Loading data to table default.myemp_partitioned partition (country=india)
Partition default.myemp_partitioned{country=india} stats: [numFiles=1, numRows=0, totalSize=30, rawDataSize=0]
OK
Time taken: 2.058 seconds
hive> 
```

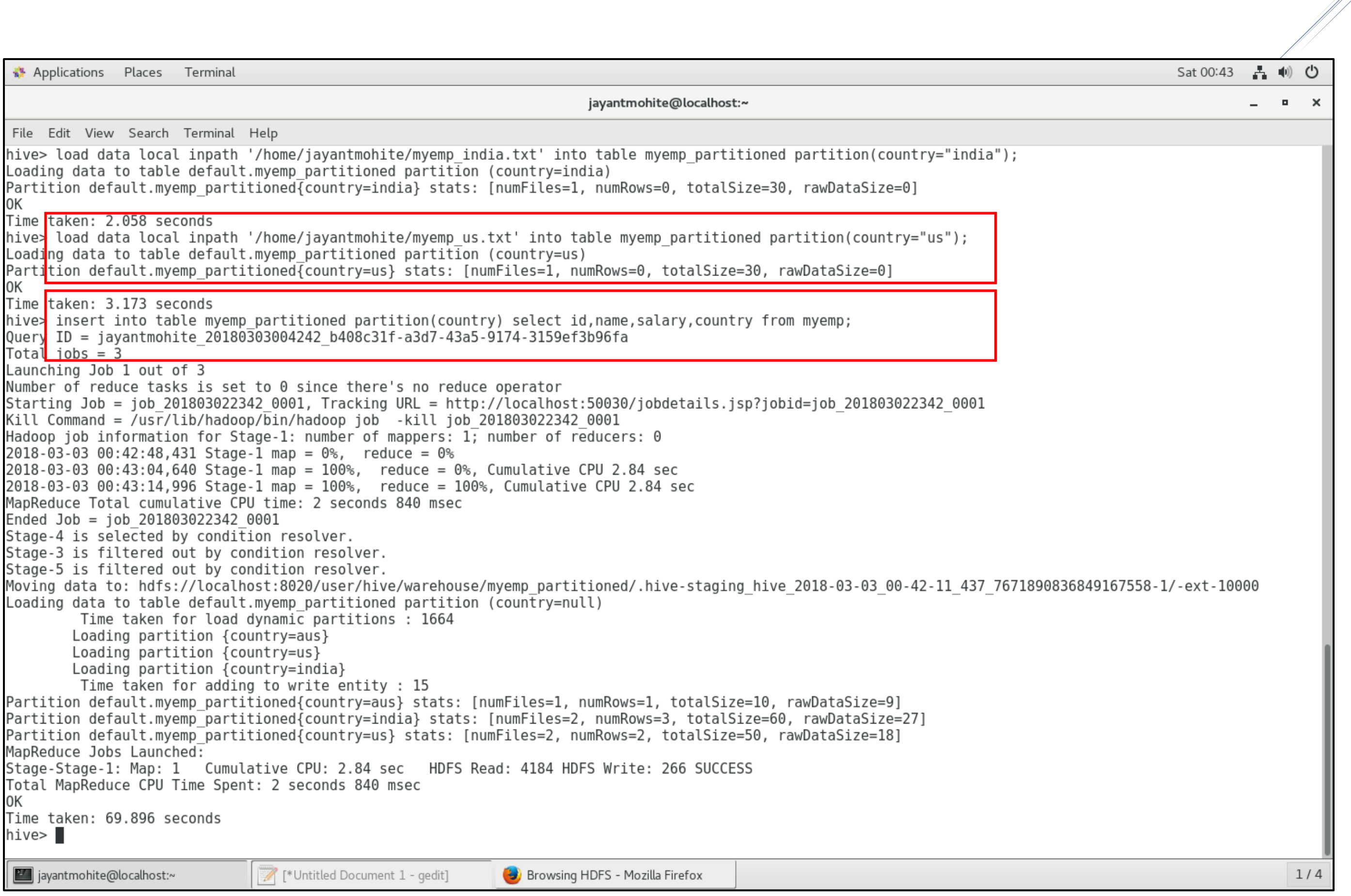

Applications
Places
Firefox Web Browser
Sat 00:39
Browsing HDFS - Mozilla Firefox
Browsing HDFS
localhost:50070/explorer.html#/user/hive/warehouse/myemp_partitioned
Search
Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities
Go!
jayantmohite@localhost:~
[*Untitled Document 1 - gedit]
Browsing HDFS - Mozilla Firefox
1 / 4

Browse Directory

/user/hive/warehouse/myemp_partitioned

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxrwxrwx	jayantmohite	supergroup	0 B	Sat Mar 03 00:37:15 -0800 2018	0	0 B	country=india

Hadoop, 2017.



```
hive> load data local inpath '/home/jayantmohite/myemp_india.txt' into table myemp_partitioned partition(country="india");
Loading data to table default.myemp_partitioned partition (country=india)
Partition default.myemp_partitioned{country=india} stats: [numFiles=1, numRows=0, totalSize=30, rawDataSize=0]
OK
Time taken: 2.058 seconds
hive> load data local inpath '/home/jayantmohite/myemp_us.txt' into table myemp_partitioned partition(country="us");
Loading data to table default.myemp_partitioned partition (country=us)
Partition default.myemp_partitioned{country=us} stats: [numFiles=1, numRows=0, totalSize=30, rawDataSize=0]
OK
Time taken: 3.173 seconds
hive> insert into table myemp_partitioned partition(country) select id,name,salary,country from myemp;
Query ID = jayantmohite_20180303004242_b408c31f-a3d7-43a5-9174-3159ef3b96fa
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks is set to 0 since there's no reduce operator
Starting Job = job_201803022342_0001, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201803022342_0001
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_201803022342_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0
2018-03-03 00:42:48,431 Stage-1 map = 0%,  reduce = 0%
2018-03-03 00:43:04,640 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.84 sec
2018-03-03 00:43:14,996 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 2.84 sec
MapReduce Total cumulative CPU time: 2 seconds 840 msec
Ended Job = job_201803022342_0001
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to: hdfs://localhost:8020/user/hive/warehouse/myemp_partitioned/.hive-staging_hive_2018-03-03_00-42-11_437_7671890836849167558-1/-ext-10000
Loading data to table default.myemp_partitioned partition (country=null)
         Time taken for load dynamic partitions : 1664
        Loading partition {country=aus}
        Loading partition {country=us}
        Loading partition {country=india}
         Time taken for adding to write entity : 15
Partition default.myemp_partitioned{country=aus} stats: [numFiles=1, numRows=1, totalSize=10, rawDataSize=9]
Partition default.myemp_partitioned{country=india} stats: [numFiles=2, numRows=3, totalSize=60, rawDataSize=27]
Partition default.myemp_partitioned{country=us} stats: [numFiles=2, numRows=2, totalSize=50, rawDataSize=18]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1   Cumulative CPU: 2.84 sec   HDFS Read: 4184 HDFS Write: 266 SUCCESS
Total MapReduce CPU Time Spent: 2 seconds 840 msec
OK
Time taken: 69.896 seconds
hive> 
```

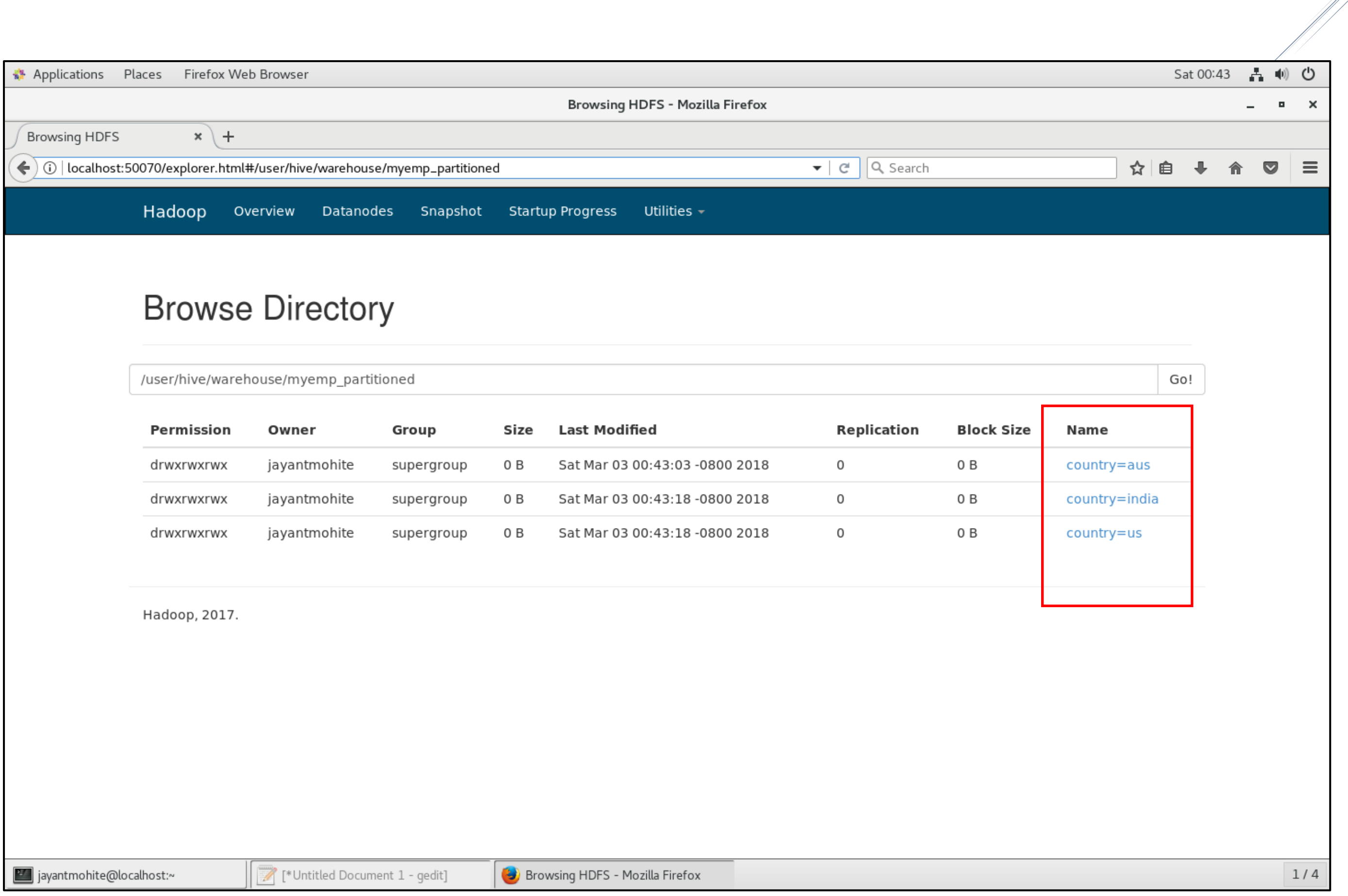
Loading data in Dynamic Partitioning

hive> insert into table myemp_partitioned partition (country) select id, name, salary, country from myemp;

In this command we load data into the partitioned table by name myemp_partitioned from a non-partitioned table by name myemp which we have created previously in this chapter.

Dynamic partitioning will automatically sense the distinct values of the partitioned columns and will create the required sub folders

In case the partition already exists, the data will be appended to the partition and if it does not exist, a new partition will be created.


Applications
Places
Firefox Web Browser
Sat 00:43
Browsing HDFS - Mozilla Firefox
Browsing HDFS
localhost:50070/explorer.html#/user/hive/warehouse/myemp_partitioned
Search
Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities
Go!
jayantmohite@localhost:~
[*Untitled Document 1 - gedit]
Browsing HDFS - Mozilla Firefox
1 / 4

Browse Directory

/user/hive/warehouse/myemp_partitioned

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxrwxrwx	jayantmohite	supergroup	0 B	Sat Mar 03 00:43:03 -0800 2018	0	0 B	country=aus
drwxrwxrwx	jayantmohite	supergroup	0 B	Sat Mar 03 00:43:18 -0800 2018	0	0 B	country=india
drwxrwxrwx	jayantmohite	supergroup	0 B	Sat Mar 03 00:43:18 -0800 2018	0	0 B	country=us

Hadoop, 2017.

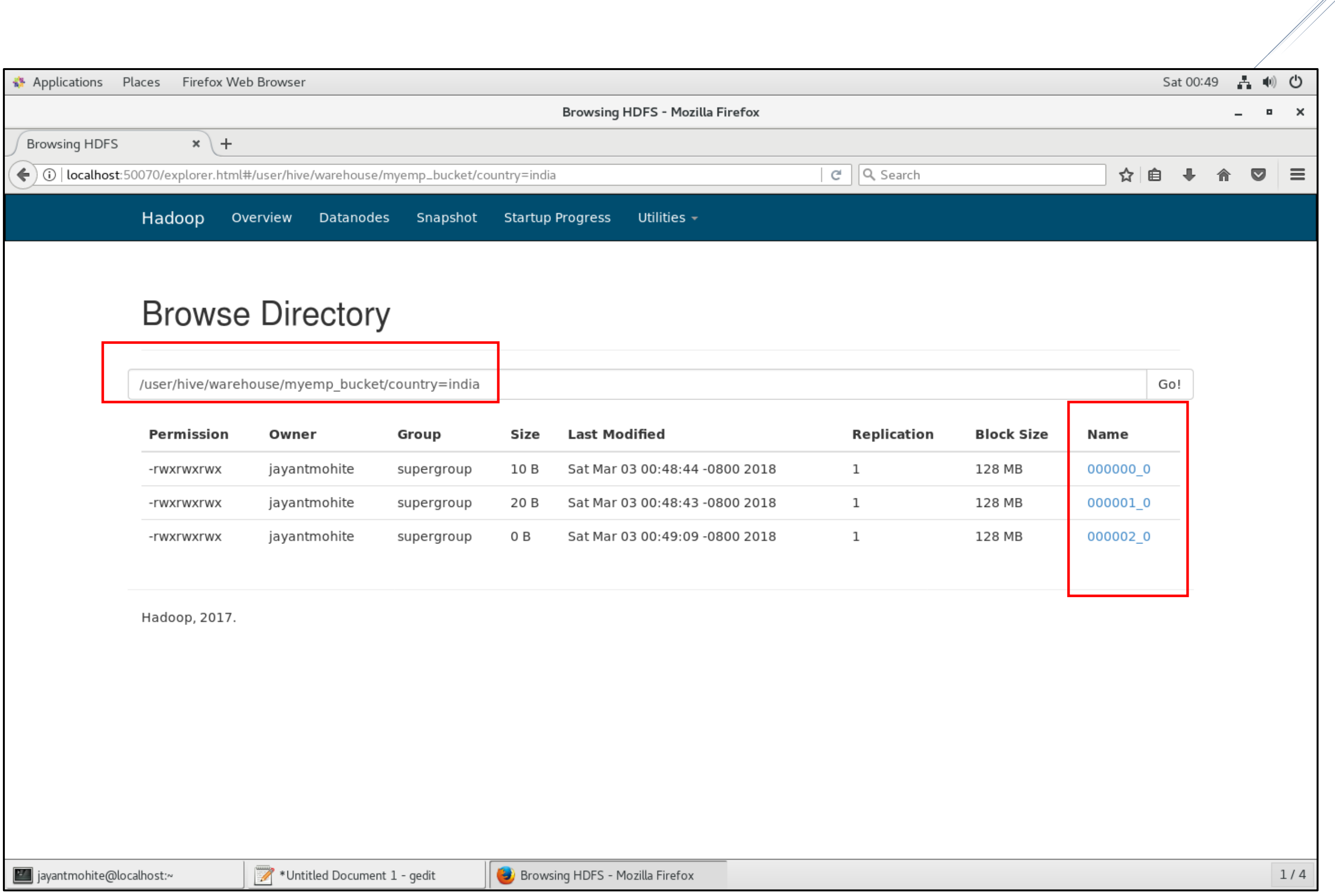
Hive Bucketing Example

Bucketing in Hive can be viewed as partitions created within partitions which is actually unconditional clustering based on number of buckets specified by the user.

Commands:

hive> create table myemp_bucket(id int, name string, salary int) partitioned by (country string) clustered by (id) into 3 buckets row format delimited fields terminated by ',' stored as textfile;

So basically this is the same syntax as for the table partitioning with only difference that we are further dividing all partitions into 3 sub-sections called as buckets. So data in every partition will be further split into 3 parts.



Applications
Places
Firefox Web Browser
Sat 00:49
Browsing HDFS - Mozilla Firefox
Browsing HDFS
localhost:50070/explorer.html#/user/hive/warehouse/myemp_bucket/country=india
Search
Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities
Go!
jayantmohite@localhost:~
*Untitled Document 1 - gedit
Browsing HDFS - Mozilla Firefox
1 / 4

Browse Directory

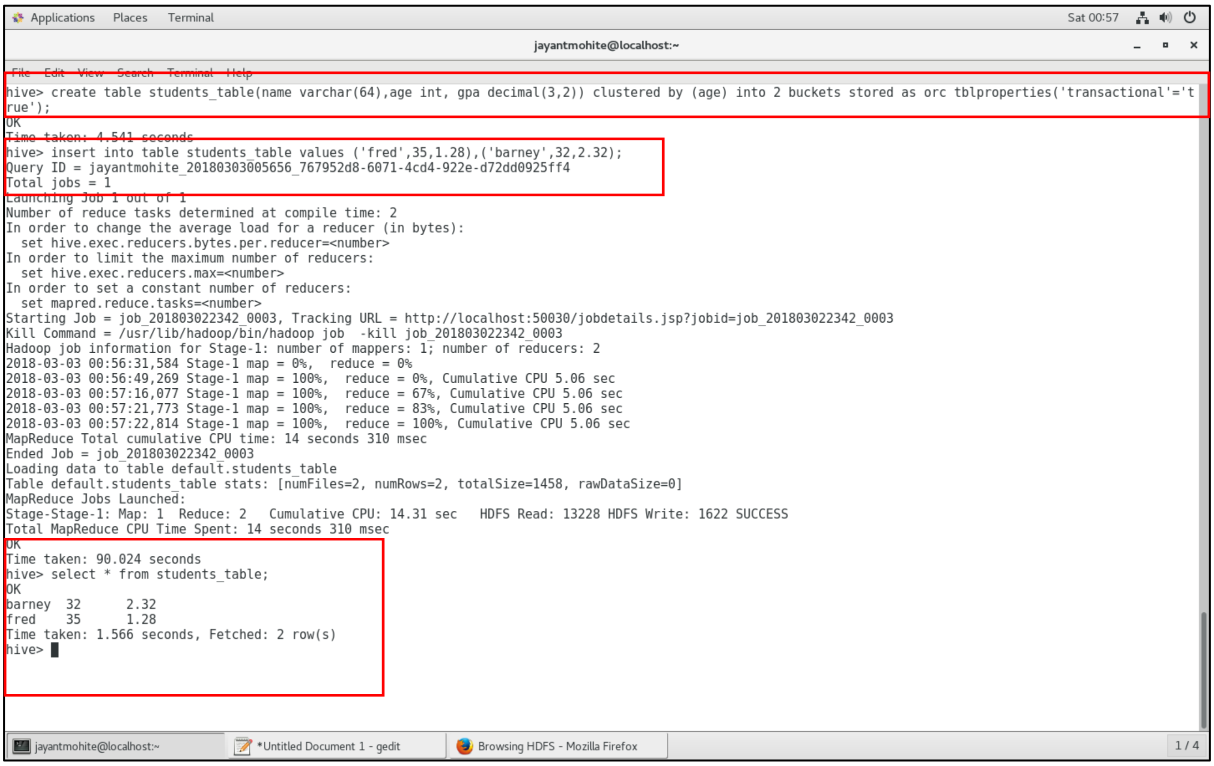
/user/hive/warehouse/myemp_bucket/country=india

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rwxrwxrwx	jayantmohite	supergroup	10 B	Sat Mar 03 00:48:44 -0800 2018	1	128 MB	000000_0
-rwxrwxrwx	jayantmohite	supergroup	20 B	Sat Mar 03 00:48:43 -0800 2018	1	128 MB	000001_0
-rwxrwxrwx	jayantmohite	supergroup	0 B	Sat Mar 03 00:49:09 -0800 2018	1	128 MB	000002_0

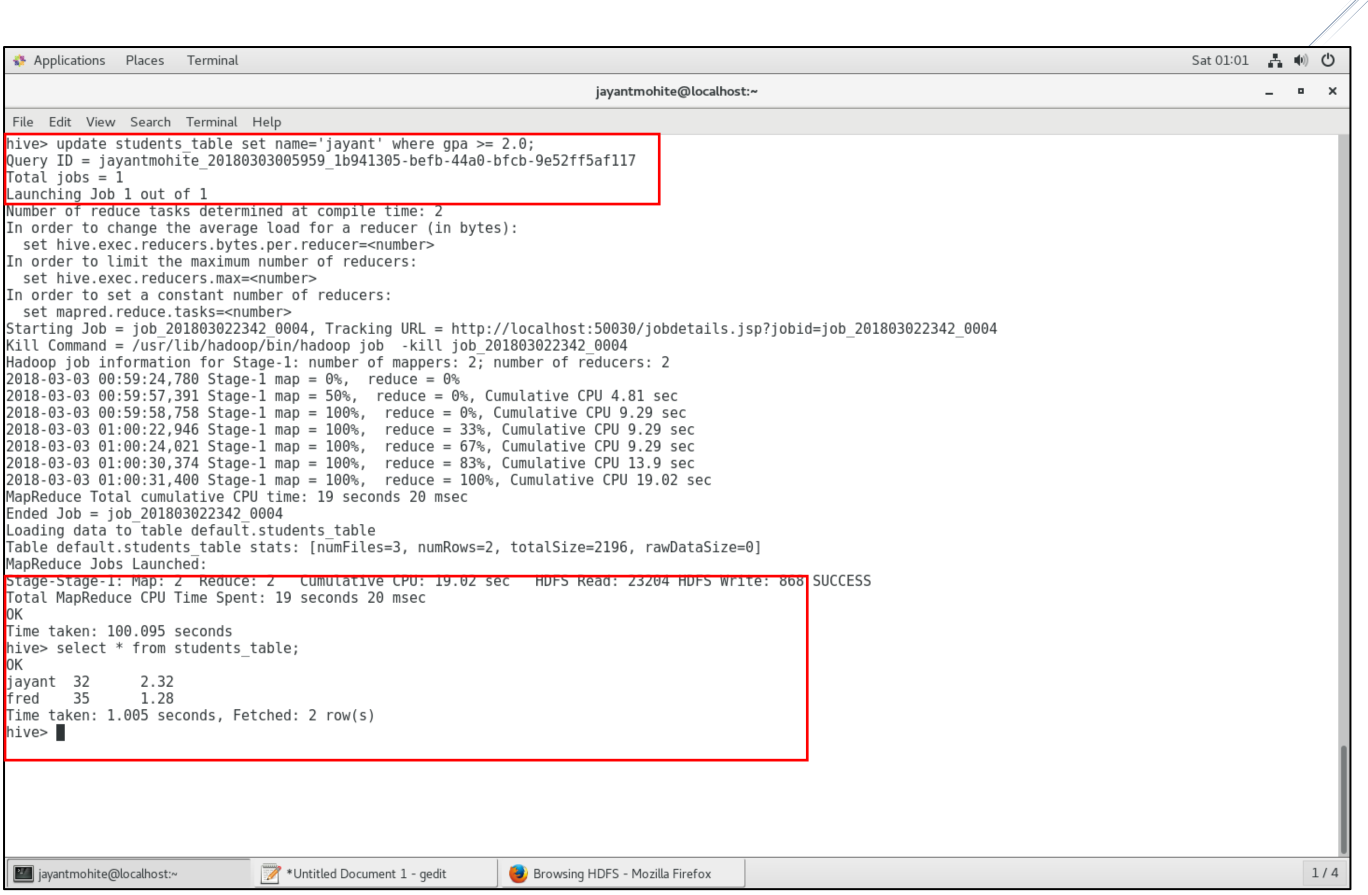
Hadoop, 2017.

Hive Transactions

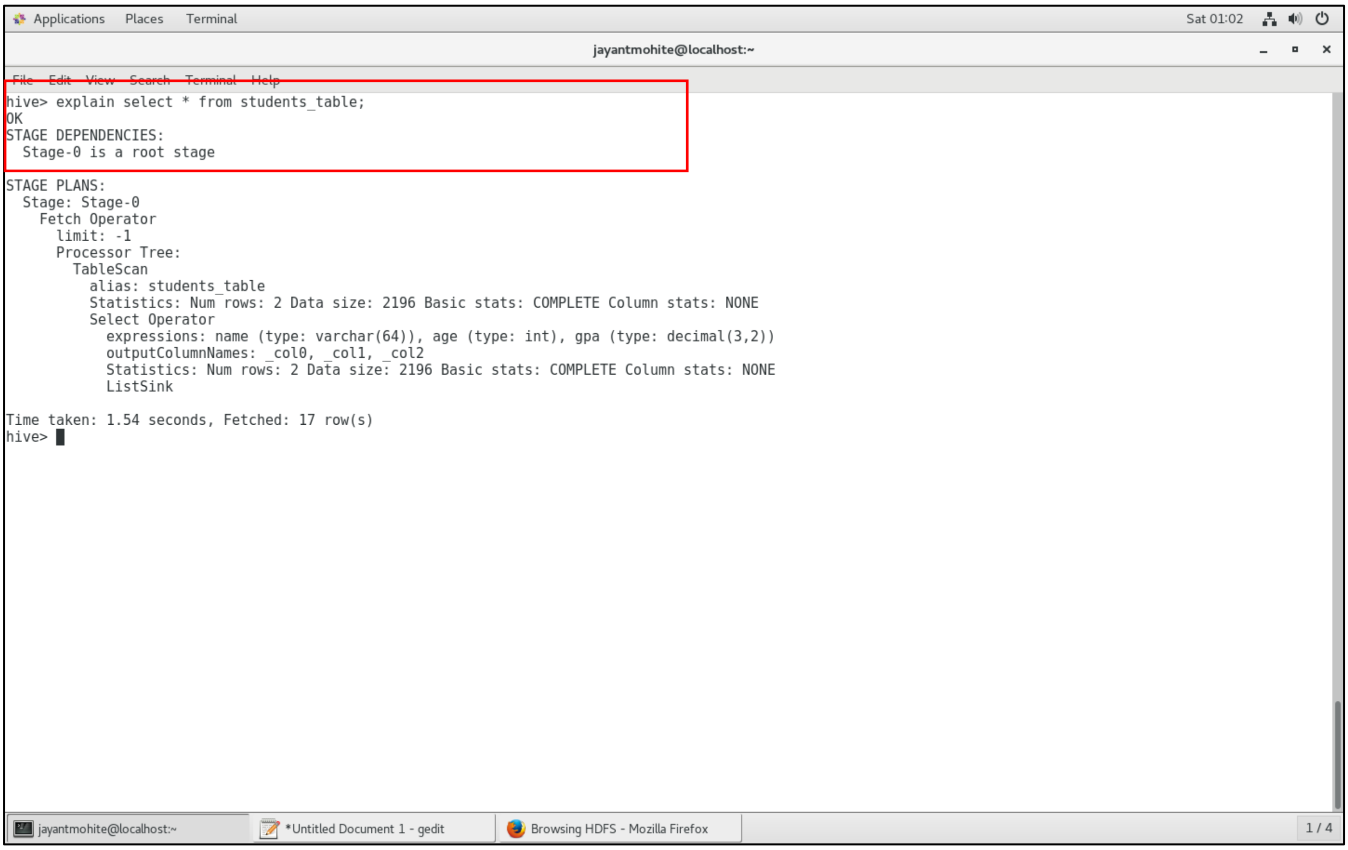
Like any other RDBMS, even Hive supports transactional queries like insert, update and delete.



```
hive> create table students_table(name varchar(64),age int, gpa decimal(3,2)) clustered by (age) into 2 buckets stored as orc tblproperties('transactional'='t
rue');
OK
Time taken: 4.541 seconds
hive> insert into table students_table values ('fred',35,1.28),('barney',32,2.32);
Query ID = jayantmohite_20180303005656_767952d8-6071-4cd4-922e-d72dd0925ff4
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 2
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 mapred.reduce.tasks=<number>
Starting Job = job_201803022342_0003, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201803022342_0003
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_201803022342_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 2
2018-03-03 00:56:31,584 Stage-1 map = 0%,  reduce = 0%
2018-03-03 00:56:49,269 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 5.06 sec
2018-03-03 00:57:16,077 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 5.06 sec
2018-03-03 00:57:21,773 Stage-1 map = 100%,  reduce = 83%, Cumulative CPU 5.06 sec
2018-03-03 00:57:22,814 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 5.06 sec
MapReduce Total cumulative CPU time: 14 seconds 310 msec
Ended Job = job_201803022342_0003
Loading data to table default.students_table
Table default.students_table stats: [numFiles=2, numRows=2, totalSize=1458, rawDataSize=0]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 2   Cumulative CPU: 14.31 sec   HDFS Read: 13228 HDFS Write: 1622 SUCCESS
Total MapReduce CPU Time Spent: 14 seconds 310 msec
OK
Time taken: 90.024 seconds
hive> select * from students_table;
OK
barney  32      2.32
fred    35      1.28
Time taken: 1.566 seconds, Fetched: 2 row(s)
hive> 
```



```
hive> update students_table set name='jayant' where gpa >= 2.0;
Query ID = jayantmohite_20180303005959_1b941305-befb-44a0-bfcb-9e52ff5af117
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 2
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 mapred.reduce.tasks=<number>
Starting Job = job_201803022342_0004, Tracking URL = http://localhost:50030/jobdetails.jsp?jobid=job_201803022342_0004
Kill Command = /usr/lib/hadoop/bin/hadoop job  -kill job_201803022342_0004
Hadoop job information for Stage-1: number of mappers: 2; number of reducers: 2
2018-03-03 00:59:24,780 Stage-1 map = 0%,  reduce = 0%
2018-03-03 00:59:57,391 Stage-1 map = 50%,  reduce = 0%, Cumulative CPU 4.81 sec
2018-03-03 00:59:58,758 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 9.29 sec
2018-03-03 01:00:22,946 Stage-1 map = 100%,  reduce = 33%, Cumulative CPU 9.29 sec
2018-03-03 01:00:24,021 Stage-1 map = 100%,  reduce = 67%, Cumulative CPU 9.29 sec
2018-03-03 01:00:30,374 Stage-1 map = 100%,  reduce = 83%, Cumulative CPU 13.9 sec
2018-03-03 01:00:31,400 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 19.02 sec
MapReduce Total cumulative CPU time: 19 seconds 20 msec
Ended Job = job_201803022342_0004
Loading data to table default.students_table
Table default.students_table stats: [numFiles=3, numRows=2, totalSize=2196, rawDataSize=0]
MapReduce Jobs Launched:
Stage-Stage-1: Map: 2  Reduce: 2   Cumulative CPU: 19.02 sec   HDFS Read: 23204 HDFS Write: 868 SUCCESS
Total MapReduce CPU Time Spent: 19 seconds 20 msec
OK
Time taken: 100.095 seconds
hive> select * from students_table;
OK
jayant  32      2.32
fred    35      1.28
Time taken: 1.005 seconds, Fetched: 2 row(s)
hive> 
```



```
hive> explain select * from students_table;
OK
STAGE DEPENDENCIES:
  Stage-0 is a root stage

STAGE PLANS:
  Stage: Stage-0
    Fetch Operator
      limit: -1
      Processor Tree:
        TableScan
          alias: students_table
          Statistics: Num rows: 2 Data size: 2196 Basic stats: COMPLETE Column stats: NONE
          Select Operator
            expressions: name (type: varchar(64)), age (type: int), gpa (type: decimal(3,2))
            outputColumnNames: _col0, _col1, _col2
            Statistics: Num rows: 2 Data size: 2196 Basic stats: COMPLETE Column stats: NONE
            ListSink

Time taken: 1.54 seconds, Fetched: 17 row(s)
hive> 
```

That’s all from this Hive Tutorials. Do check out the assignments added in this book for additional references.