Big Data And Hadoop

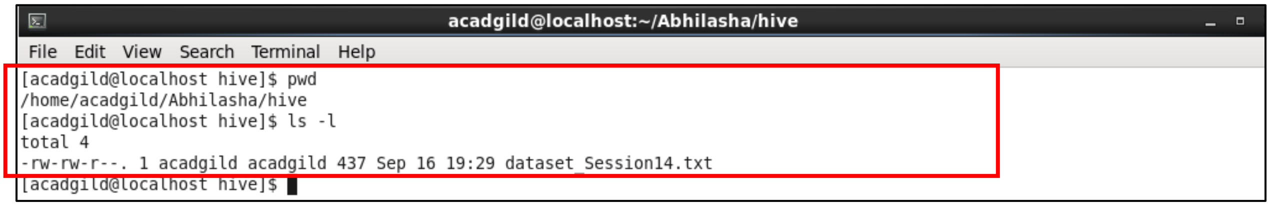
Session 14 - Assignment 1

**Problem Statement:**

Create a database named 'custom'.
Create a table named temperature_data inside custom having below fields:
1. date (mm-dd-yyyy) format
2. zip code
3. temperature
The table will be loaded from comma-delimited file.
Load the dataset.txt (which is ',' delimited) in the table.

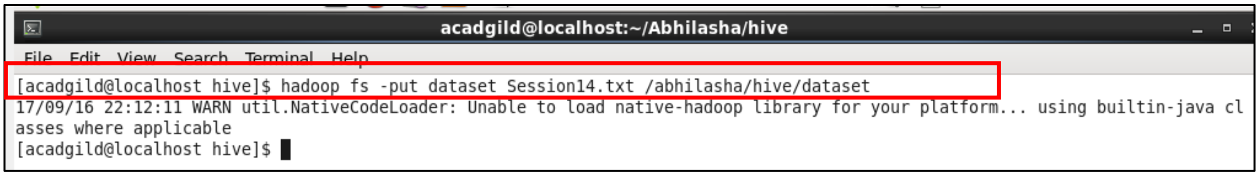
**Solution:**

**Input File:** The input file is downloaded and placed on the local system at **/home/acadgild/Abhilasha/hive** as shown below:


acadgild@localhost:~/Abhilasha/hive
File Edit View Search Terminal Help

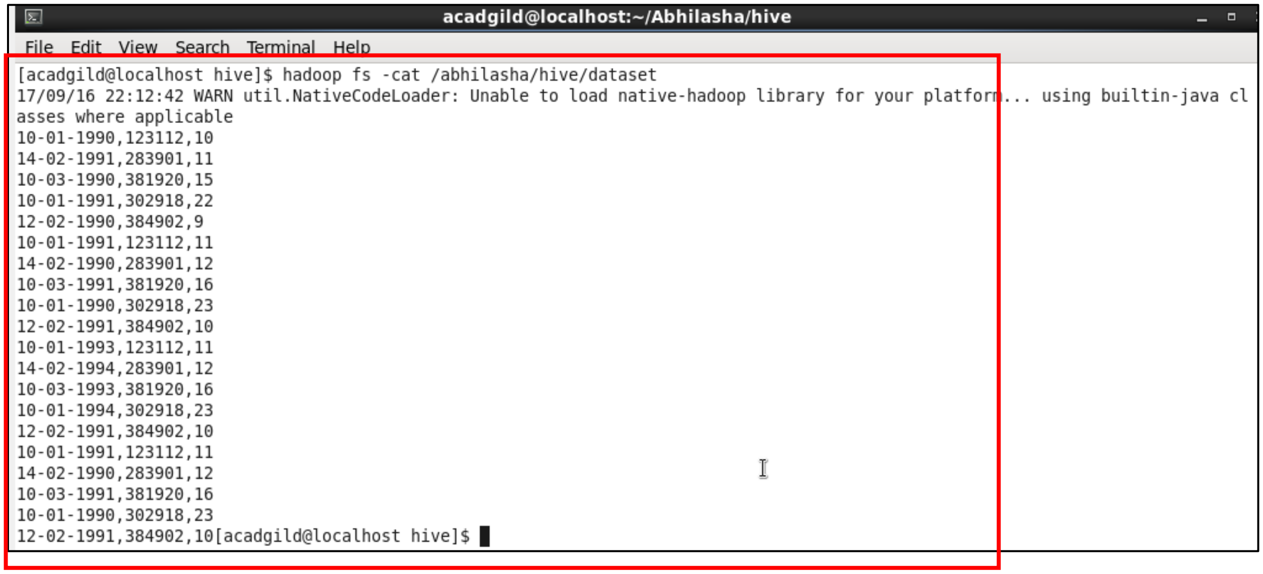
```
[acadgild@localhost hive]$ pwd
/home/acadgild/Abhilasha/hive
[acadgild@localhost hive]$ ls -l
total 4
-rw-rw-r--. 1 acadgild acadgild 437 Sep 16 19:29 dataset_Session14.txt
[acadgild@localhost hive]$
```

We put this file on HDFS using the **put** command at location **/abhilasha/hive** and renamed the file to **dataset** as follows:


acadgild@localhost:~/Abhilasha/hive
File Edit View Search Terminal Help

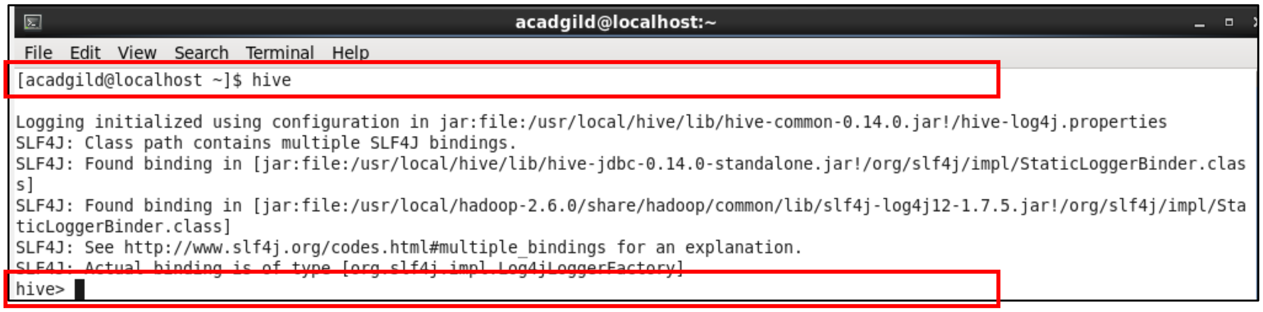
```
[acadgild@localhost hive]$ hadoop fs -put dataset Session14.txt /abhilasha/hive/dataset
17/09/16 22:12:11 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
[acadgild@localhost hive]$
```

The contents of the dataset can be seen using cat command as follows:


acadgild@localhost:~/Abhilasha/hive
File Edit View Search Terminal Help

```
[acadgild@localhost hive]$ hadoop fs -cat /abhilasha/hive/dataset
17/09/16 22:12:42 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java cl
asses where applicable
10-01-1990,123112,10
14-02-1991,283901,11
10-03-1990,381920,15
10-01-1991,302918,22
12-02-1990,384902,9
10-01-1991,123112,11
14-02-1990,283901,12
10-03-1991,381920,16
10-01-1990,302918,23
12-02-1991,384902,10
10-01-1993,123112,11
14-02-1994,283901,12
10-03-1993,381920,16
10-01-1994,302918,23
12-02-1991,384902,10
10-01-1991,123112,11
14-02-1990,283901,12
10-03-1991,381920,16
10-01-1990,302918,23
12-02-1991,384902,10[acadgild@localhost hive]$
```

**Start hive:** We start the hive command line by executing the command **hive** as shown below:

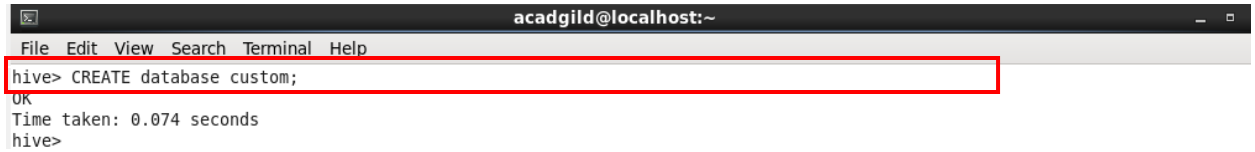


```
[acadgild@localhost ~]$ hive

Logging initialized using configuration in jar:file:/usr/local/hive/lib/hive-common-0.14.0.jar!/hive-log4j.properties
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hive/lib/hive-jdbc-0.14.0-standalone.jar!/org/slf4j/impl/StaticLoggerBinder.clas
s]
SLF4J: Found binding in [jar:file:/usr/local/hadoop-2.6.0/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/Sta
ticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
hive>
```

The above snapshot also shows that hive prompt has started. A pre-requisite to use hive is to start mysql server. This was done using the command **sudo service mysqld start.**

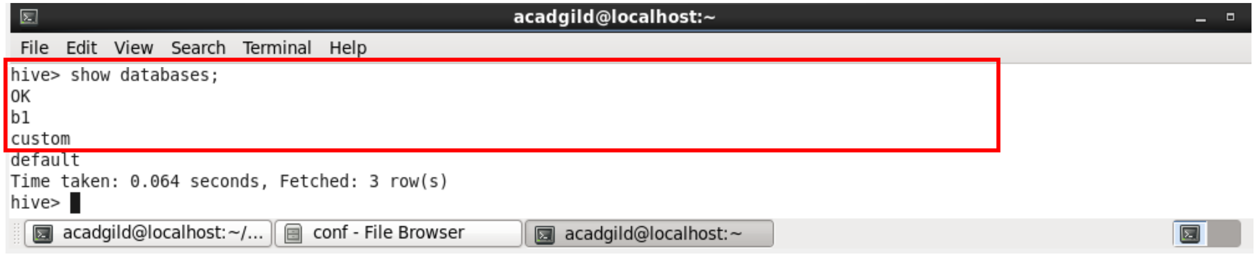
**Solution to the problem statement:**

i. First we need to create a database.
Databases are used to logically group production tables.
Command used to create database is **CREATE DATABASE custom;**
This resulted in creation of database named **custom.**



```
hive> CREATE database custom;
OK
Time taken: 0.074 seconds
hive>
```

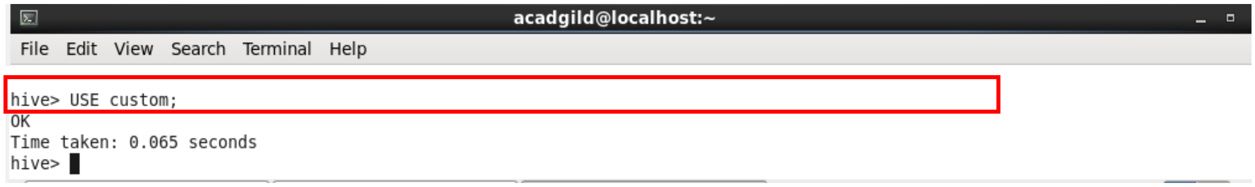
**ii.** The database that got created can be listed using the command **SHOW DATABASES;**



```
hive> show databases;
OK
b1
custom
default
Time taken: 0.064 seconds, Fetched: 3 row(s)
hive>
```

**Custom** database appeared in the list.

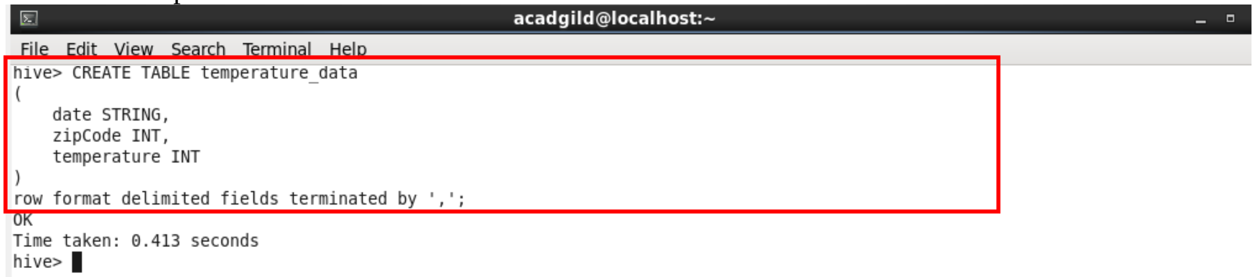
iii. Next is to mention which database we want to work on. This is done using the command **USE custom;**



```
hive> USE custom;
OK
Time taken: 0.065 seconds
hive>
```

iv. Next we create the table temperature_data inside custom having below fields:

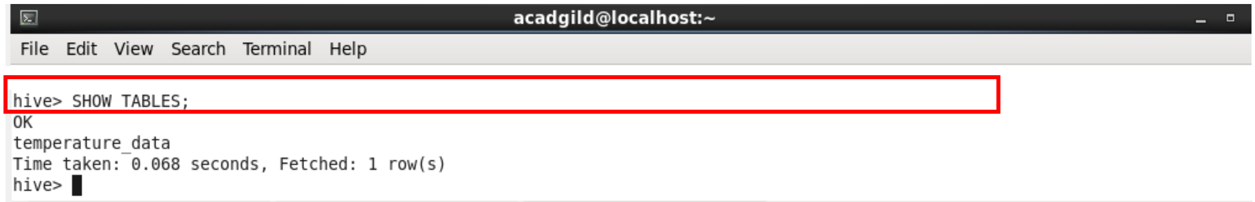
a. date (mm-dd-yyyy) format
b. zip code
c. temperature



```
hive> CREATE TABLE temperature_data
(
    date STRING,
    zipCode INT,
    temperature INT
)
row format delimited fields terminated by ',';
OK
Time taken: 0.413 seconds
hive>
```

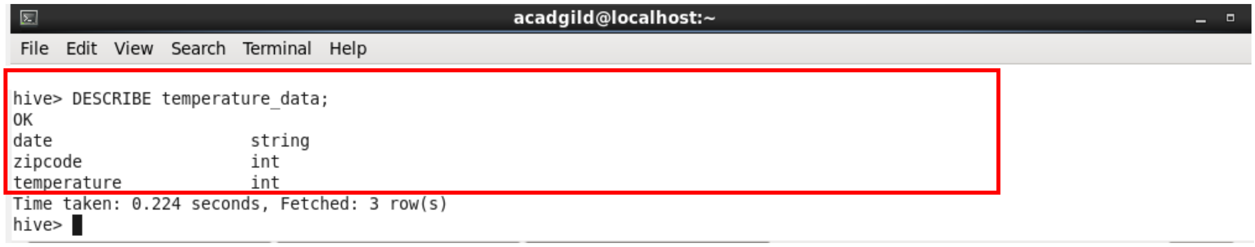
We have also specified the delimiter for the fields to be ‘,’.

iv. **SHOW TABLES;** command lists all the tables in the current database and the table we created now also appears in the list as follows:



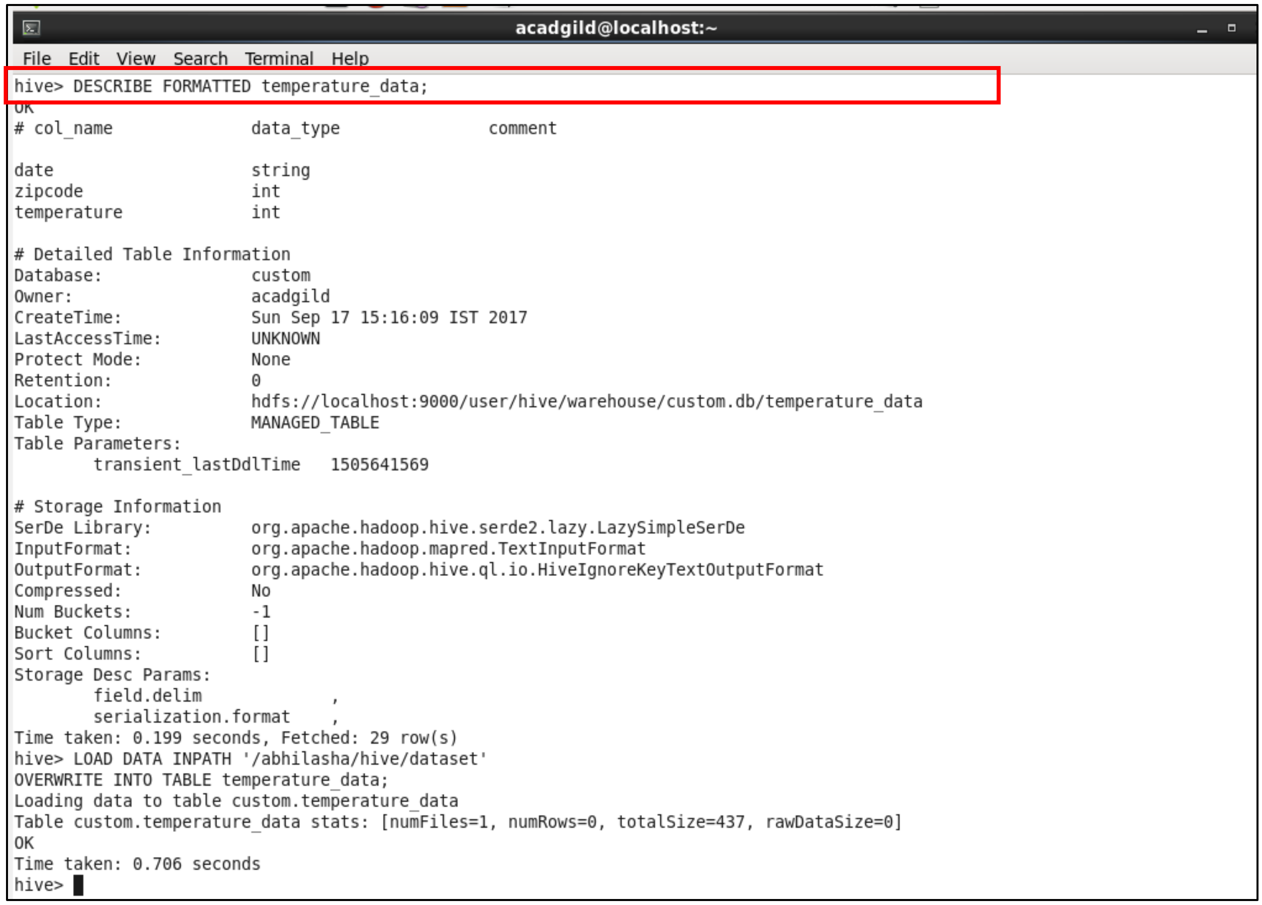
```
hive> SHOW TABLES;
OK
temperature_data
Time taken: 0.068 seconds, Fetched: 1 row(s)
hive>
```

v. Using **DESCRIBE** command gives the schema of the table as shown below:



```
hive> DESCRIBE temperature_data;
OK
date                    string
zipcode                 int
temperature             int
Time taken: 0.224 seconds, Fetched: 3 row(s)
hive>
```

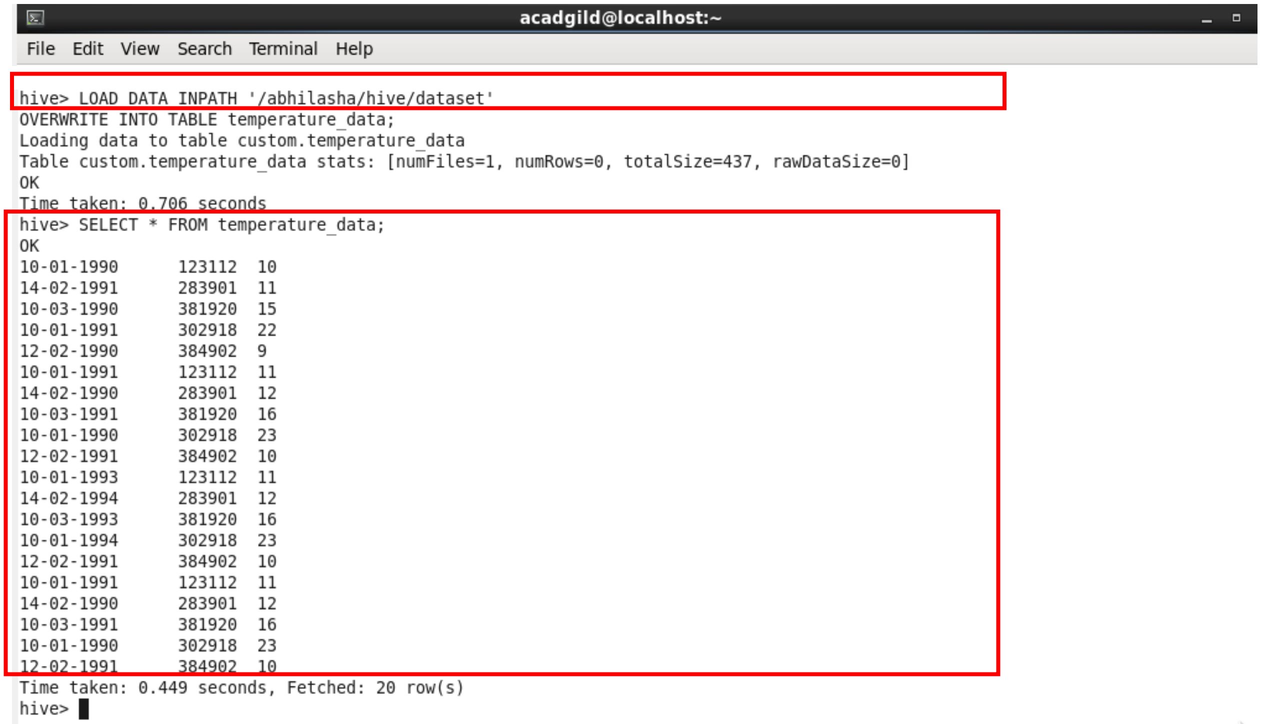
vi. We can also use **DESCRIBE FORMATTED** command to get detailed description of the as follows:



```
hive> DESCRIBE FORMATTED temperature_data;
OK
# col_name              data_type               comment

date                    string
zipcode                 int
temperature             int

# Detailed Table Information
Database:               custom
Owner:                  acadgild
CreateTime:             Sun Sep 17 15:16:09 IST 2017
LastAccessTime:         UNKNOWN
Protect Mode:           None
Retention:              0
Location:               hdfs://localhost:9000/user/hive/warehouse/custom.db/temperature_data
Table Type:             MANAGED_TABLE
Table Parameters:
        transient_lastDdlTime   1505641569

# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:            org.apache.hadoop.mapred.TextInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        field.delim             ,
        serialization.format    ,
Time taken: 0.199 seconds, Fetched: 29 row(s)
hive> LOAD DATA INPATH '/abhilasha/hive/dataset'
OVERWRITE INTO TABLE temperature_data;
Loading data to table custom.temperature_data
Table custom.temperature_data stats: [numFiles=1, numRows=0, totalSize=437, rawDataSize=0]
OK
Time taken: 0.706 seconds
hive>
```

vii. Now, we load the input file into the table created using the **LOAD** command. Since, the file was located in HDFS at **/abhilasha/hive,** we have specified this path. We can also load a file that is located on local file system using the keyword **LOCAL** in the command.

We also use the query **SELECT * from temperature_data;** to verify if the data is loaded as expected into the database.



```
hive> LOAD DATA INPATH '/abhilasha/hive/dataset'
OVERWRITE INTO TABLE temperature_data;
Loading data to table custom.temperature_data
Table custom.temperature_data stats: [numFiles=1, numRows=0, totalSize=437, rawDataSize=0]
OK
Time taken: 0.706 seconds
hive> SELECT * FROM temperature_data;
OK
10-01-1990      123112  10
14-02-1991      283901  11
10-03-1990      381920  15
10-01-1991      302918  22
12-02-1990      384902  9
10-01-1991      123112  11
14-02-1990      283901  12
10-03-1991      381920  16
10-01-1990      302918  23
12-02-1991      384902  10
10-01-1993      123112  11
14-02-1994      283901  12
10-03-1993      381920  16
10-01-1994      302918  23
12-02-1991      384902  10
10-01-1991      123112  11
14-02-1990      283901  12
10-03-1991      381920  16
10-01-1990      302918  23
12-02-1991      384902  10
Time taken: 0.449 seconds, Fetched: 20 row(s)
hive> 
```