Exp5: **Installation of Hive on Ubuntu**

Aim:

To Download and install Hive, Understanding Startup scripts, Configuration files.

Procedure:

Step 1: Download and extract it

Download the Apache hive and extract it use tar, the commands given below:

$wgethttps://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz

$ tar –xvf apache-hive-3.1.2-bin.tar.gz

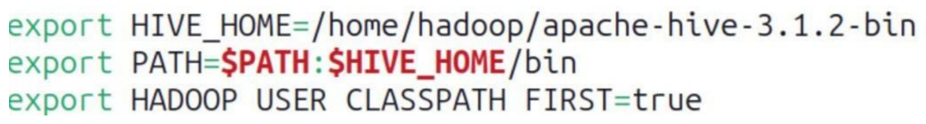
Step 2: Place different configuration properties in Apache Hive

In this step, we are going to do two things

- Placing Hive Home path in bashrc file

  $nano .bashrc

And append the below lines in it



```
export HIVE_HOME=/home/hadoop/apache-hive-3.1.2-bin
export PATH=$PATH:$HIVE_HOME/bin
export HADOOP USER CLASSPATH FIRST=true
```

2. Exporting **Hadoop path in Hive-config.sh** (To communicate with the Hadoop eco system we are defining Hadoop Home path in hive config field) **Open the hive-config.sh as shown in below**

   $cd apache-hive-3.1.2-bin/bin

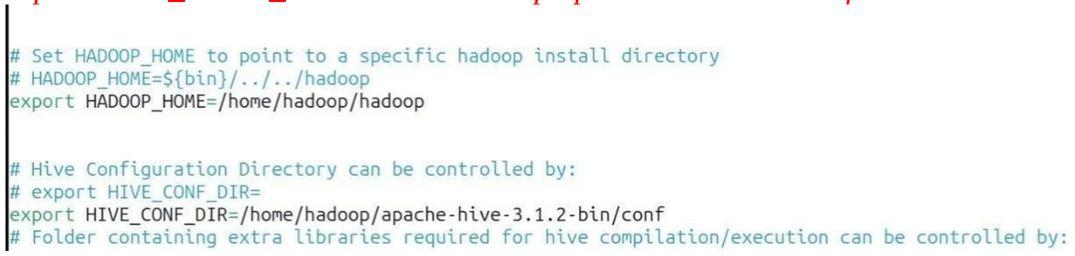
   $cp hive-env.sh.template hive-env.sh

   $nano hive-env.sh

Append the below commands on it

   export HADOOP_HOME=/home/Hadoop/Hadoop

   export HIVE_CONF_DIR=/home/Hadoop/apache-hive-3.1.2/conf



```
# Set HADOOP_HOME to point to a specific hadoop install directory
# HADOOP_HOME=${bin}/../../hadoop
export HADOOP_HOME=/home/hadoop/hadoop


# Hive Configuration Directory can be controlled by:
# export HIVE_CONF_DIR=
export HIVE_CONF_DIR=/home/hadoop/apache-hive-3.1.2-bin/conf
# Folder containing extra libraries required for hive compilation/execution can be controlled by:
```

Step 3: Install mysql

1. Install mysql in Ubuntu by running this command:

$sudo apt update

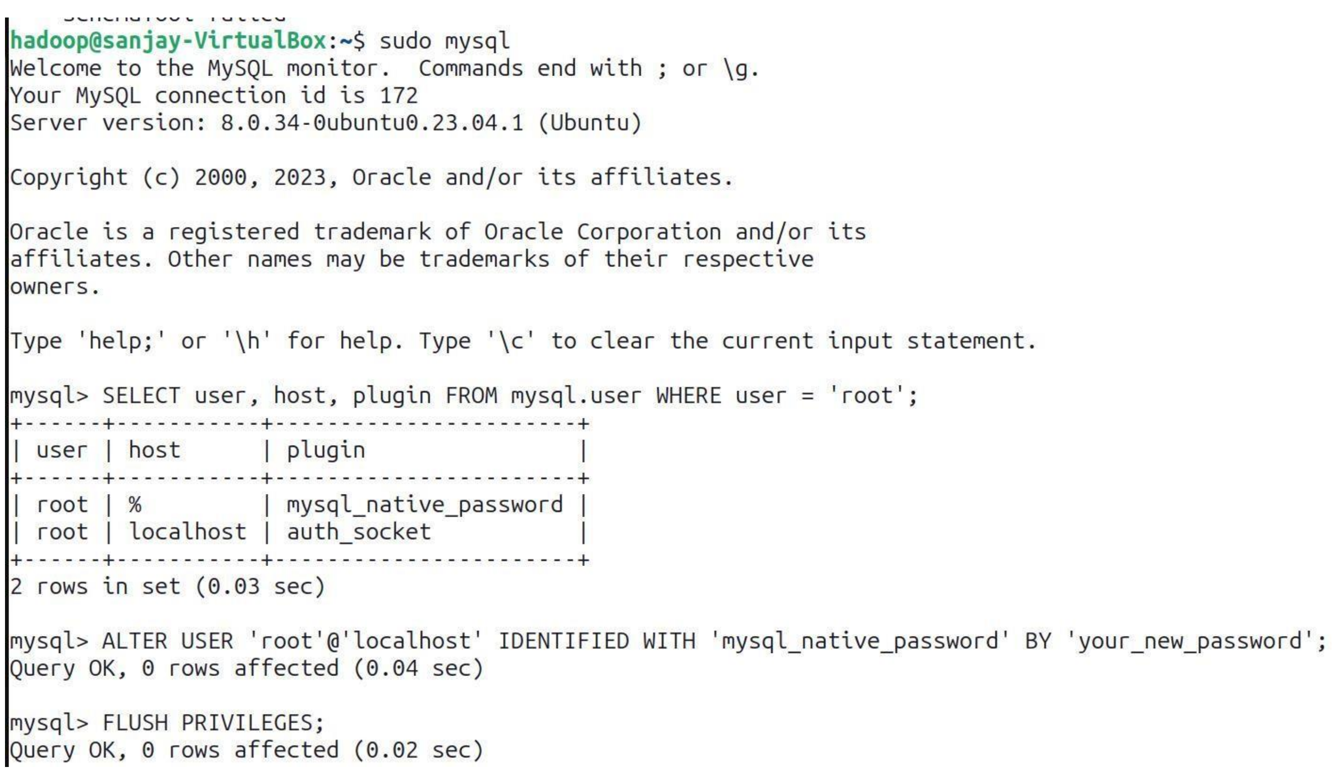
$sudo apt install mysql-server

2. *Alter username and password for MySQLby running below commands:*

   $sudomysql

Pops command line interface for MySQLand run the below SQL queries to change username and set password

mysql> SELECT user, host, plugin FROM mysql.user WHERE user = 'root';



```
hadoop@sanjay-VirtualBox:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 172
Server version: 8.0.34-0ubuntu0.23.04.1 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SELECT user, host, plugin FROM mysql.user WHERE user = 'root';
+------+-----------+-----------------------+
| user | host      | plugin                |
+------+-----------+-----------------------+
| root | %         | mysql_native_password |
| root | localhost | auth_socket           |
+------+-----------+-----------------------+
2 rows in set (0.03 sec)

mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH 'mysql_native_password' BY 'your_new_password';
Query OK, 0 rows affected (0.04 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.02 sec)
```

mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH 'mysql_native_password' BY 'your_new_password';
mysql> FLUSH PRIVILEGES;

Step 4:Config hive-site.xml

Config the hive-site.xml by appending this xml code and change the username and password according to your MySQL.

$cd apache-hive-3.1.2-bin/bin
$cp hive-default.xml.template hive-site.xml
$nano hive-site.xml

Append these lines into it
Replace root as your username of MySQL
Replaceyour_new_password as with your password of MySQL

```
<configuration>
<property>
        <name>javax.jdo.option.ConnectionURL</name>
        <value>jdbc:mysql://localhost/metastore?createDatabaseIfNotExist=true</value>
        </property>

        <property>
        <name>javax.jdo.option.ConnectionDriverName</name>
        <value>com.mysql.cj.jdbc.Driver</value>
        </property>

        <property>
        <name>javax.jdo.option.ConnectionUserName</name>
        <value>root</value>
        </property>
        <property>
        <name>javax.jdo.option.ConnectionPassword</name>
        <value>your_new_password</value>
        </property>

        <property>
        <name>datanucleus.autoCreateSchema</name>
        <value>true</value>
        </property>

        <property>
        <name>datanucleus.fixedDatastore</name>
        <value>true</value>
        </property>

        <property>
        <name>datanucleus.autoCreateTables</name>
        <value>True</value>
        </property>

</configuration>
```

Step 5: Setup MySQL java connector:

First, you'll need to download the MySQL Connector/J, which is the JDBC driver for MySQL. You can download it from the below link

https://drive.google.com/file/d/1QFhB7Kvcat7a4LzDRe6GcmZva1yAxKz-/view?usp=drive_link

Copy the downloaded MySQL Connector/J JAR file to the Hive library directory. By default, the Hive library directory is usually located at*/path/to/apache-hive-3.1.2/lib/*on Ubuntu. Use the following command to copy the JAR file:

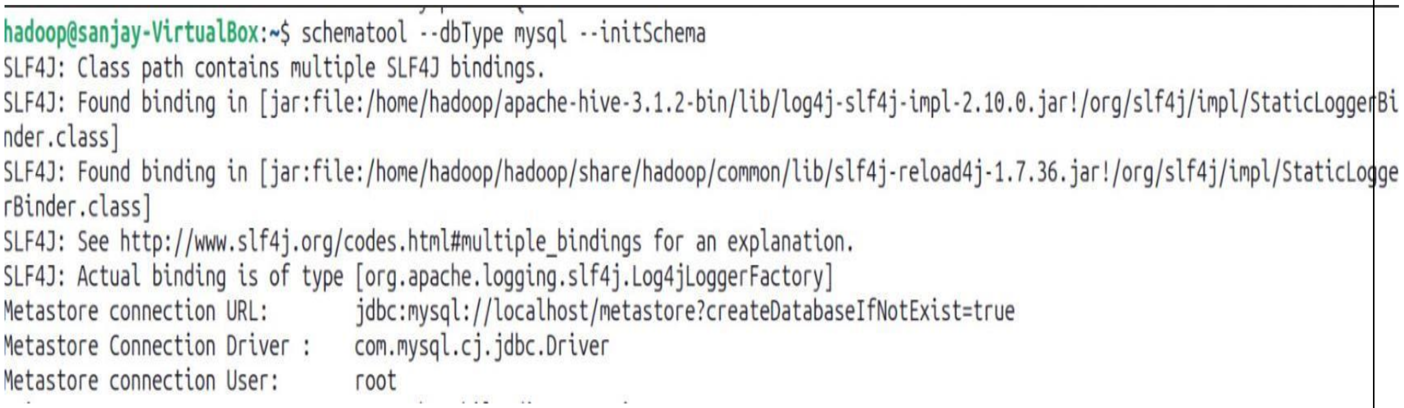
$sudo cp /path/to/mysql-connector-java-8.0.15.jar /path/to/apache-hive-3.1.2/lib/

Replace /path/to/ with the actual path to the JAR file.

Step 6:Initialize the Hive Metastore Schema:

Run the following command to initialize the Hive metastore schema:

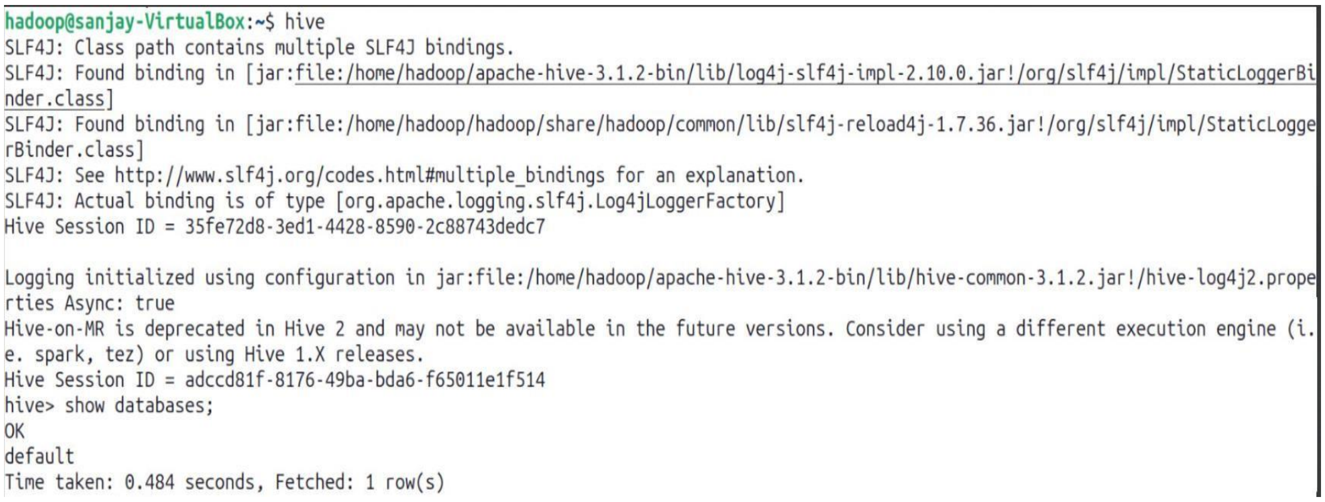
$$HIVE_HOME/bin/schematool -initSchema -dbTypemysql



```
hadoop@sanjay-VirtualBox:~$ schematool --dbType mysql --initSchema
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBi
nder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/hadoop/share/hadoop/common/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLogge
rBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Metastore connection URL:        jdbc:mysql://localhost/metastore?createDatabaseIfNotExist=true
Metastore Connection Driver :    com.mysql.cj.jdbc.Driver
Metastore connection User:       root
```

Step 7: Start hive:

You can test Hive by running the Hive shell: Copy code hive You should be able to run Hive queries, and metadata will be stored in your MySQL database.
$hive



```
hadoop@sanjay-VirtualBox:~$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerBi
nder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/hadoop/share/hadoop/common/lib/slf4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLogge
rBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 35fe72d8-3ed1-4428-8590-2c88743dedc7

Logging initialized using configuration in jar:file:/home/hadoop/apache-hive-3.1.2-bin/lib/hive-common-3.1.2.jar!/hive-log4j2.prope
rties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.
e. spark, tez) or using Hive 1.X releases.
Hive Session ID = adccd81f-8176-49ba-bda6-f65011e1f514
hive> show databases;
OK
default
Time taken: 0.484 seconds, Fetched: 1 row(s)
```

Result:

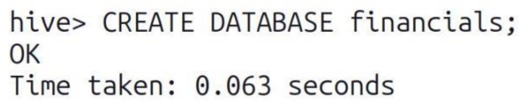
Thus, the Apache Hive installation is completed successfully on Ubuntu.

Exp5a: Design and test various schema models to optimize data storage and retrieval Using Hive.

Aim:
To Design and test various schema models to optimize data storage and retrieval Using Hbase.

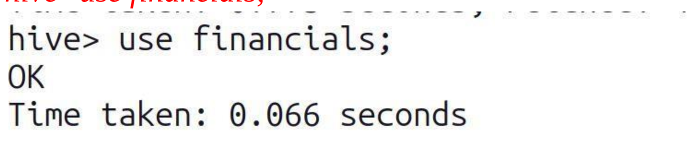
Procedure:
Step 1: Start Hive
Open a terminal and start Hive by running:
$hive

Step 2: Create a Database
Create a new database in Hive:
hive>CREATE DATABASE financials;



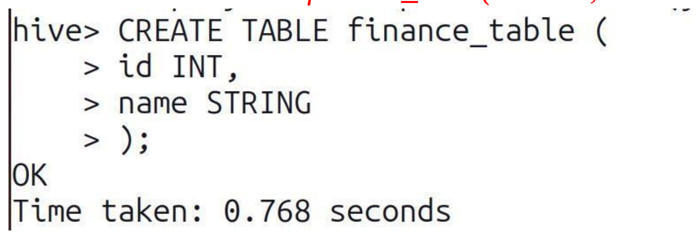
```
hive> CREATE DATABASE financials;
OK
Time taken: 0.063 seconds
```

Step 3: Use the Database*:*
Switch to the newly created database:
hive>use financials;



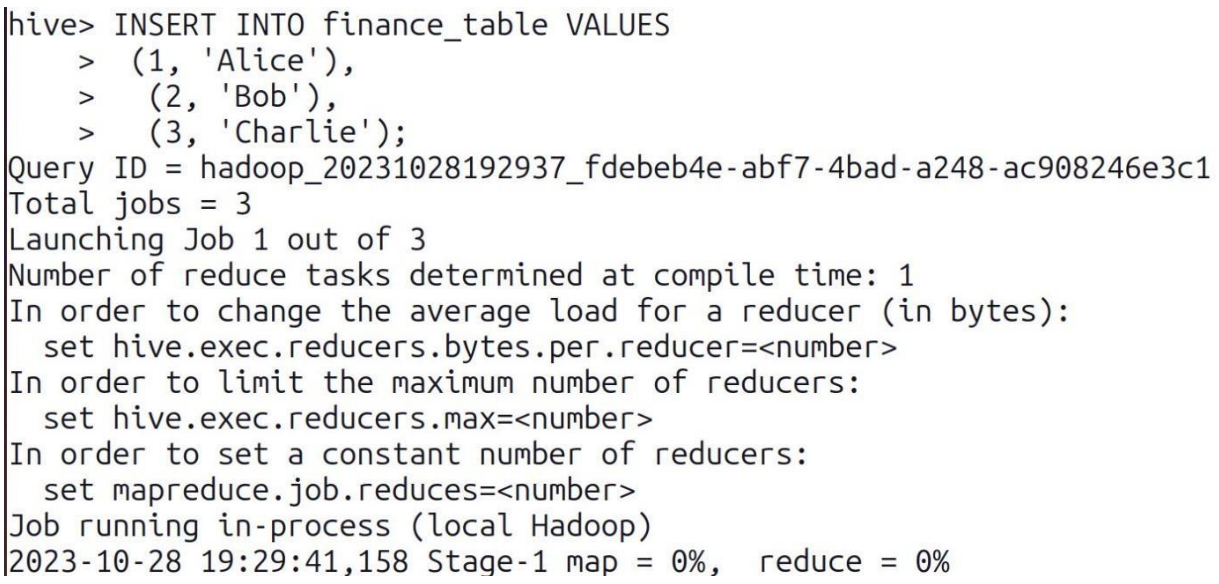
```
hive> use financials;
OK
Time taken: 0.066 seconds
```

Step 4: Create a Table:
Create a simple table in your database:
hive>CREATE TABLE finance_table(id INT,   name STRING);



```
hive> CREATE TABLE finance_table (
    > id INT,
    > name STRING
    > );
OK
Time taken: 0.768 seconds
```

Step 5: Load Sample Data:
You can insert sample data into the table:
hive>INSERT INTO finance_tableVALUES (1, 'Alice'), (2, 'Bob'),   (3, 'Charlie');



```
hive> INSERT INTO finance_table VALUES
    >  (1, 'Alice'),
    >   (2, 'Bob'),
    >   (3, 'Charlie');
Query ID = hadoop_20231028192937_fdebeb4e-abf7-4bad-a248-ac908246e3c1
Total jobs = 3
Launching Job 1 out of 3
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>
Job running in-process (local Hadoop)
2023-10-28 19:29:41,158 Stage-1 map = 0%,  reduce = 0%
```

Step 6: Query Your Data

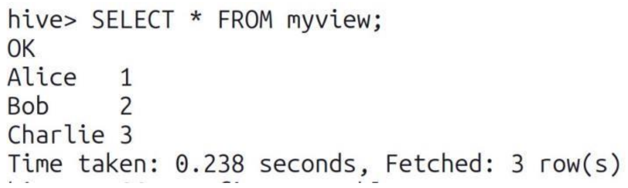
Use SQL-like queries to retrieve data from your table:

hive>CREATE VIEW myview AS SELECT name, id FROM finance_table;

Step 7: View the data:

To see the data in the view, you would need to query the view

*hive>SELECT*FROM myview;*

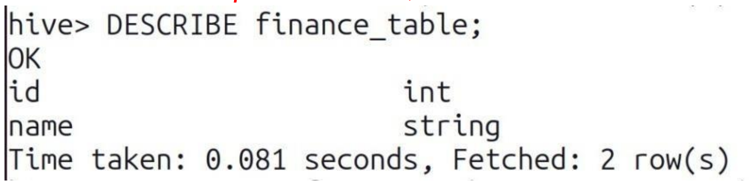


```
hive> SELECT * FROM myview;
OK
Alice   1
Bob     2
Charlie 3
Time taken: 0.238 seconds, Fetched: 3 row(s)
```

Step 8: Describe a Table:

You can describe the structure of a table using the DESCRIBE command:

hive>DESCRIBE finance_table;

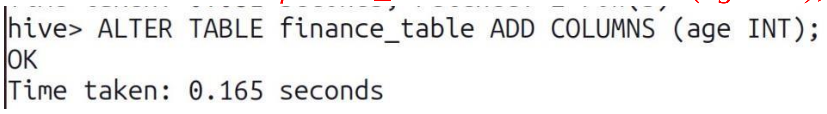


```
hive> DESCRIBE finance_table;
OK
id                      int
name                    string
Time taken: 0.081 seconds, Fetched: 2 row(s)
```

Step 9: Alter a Table:

You can alter the table structure by adding a new column:

hive>ALTER TABLE finance_table ADD COLUMNS (age INT);

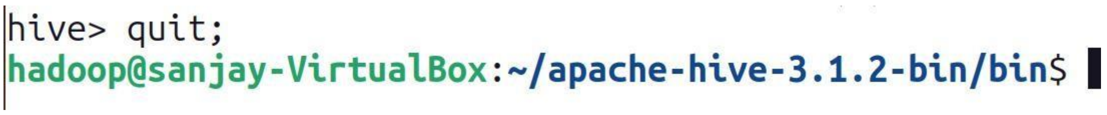


```
hive> ALTER TABLE finance_table ADD COLUMNS (age INT);
OK
Time taken: 0.165 seconds
```

Step 10: Quit Hive:

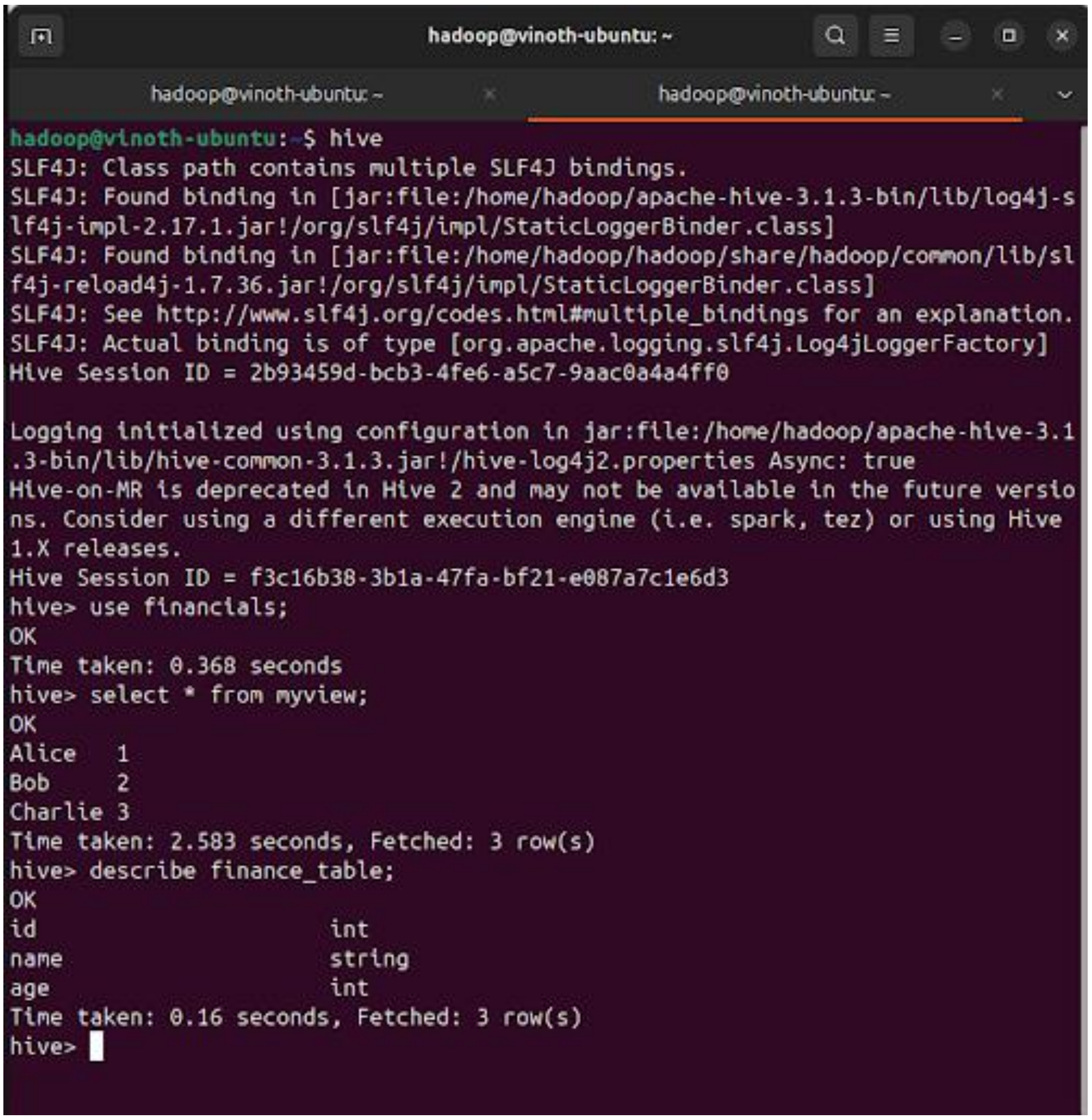
To exit the Hive CLI, simply type:

hive>quit;



```
hive> quit;
hadoop@sanjay-VirtualBox:~/apache-hive-3.1.2-bin/bin$
```

Output:



```
hadoop@vinoth-ubuntu:~$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hadoop/apache-hive-3.1.3-bin/lib/log4j-s
lf4j-impl-2.17.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/hadoop/hadoop/share/hadoop/common/lib/sl
f4j-reload4j-1.7.36.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Hive Session ID = 2b93459d-bcb3-4fe6-a5c7-9aac0a4a4ff0

Logging initialized using configuration in jar:file:/home/hadoop/apache-hive-3.1
.3-bin/lib/hive-common-3.1.3.jar!/hive-log4j2.properties Async: true
Hive-on-MR is deprecated in Hive 2 and may not be available in the future versio
ns. Consider using a different execution engine (i.e. spark, tez) or using Hive
1.X releases.
Hive Session ID = f3c16b38-3b1a-47fa-bf21-e087a7c1e6d3
hive> use financials;
OK
Time taken: 0.368 seconds
hive> select * from myview;
OK
Alice   1
Bob     2
Charlie 3
Time taken: 2.583 seconds, Fetched: 3 row(s)
hive> describe finance_table;
OK
id                      int
name                    string
age                     int
Time taken: 0.16 seconds, Fetched: 3 row(s)
hive>
```

Result:
Thus, the usage of various commands in Hive has been successfully completed.