

Exp.No.:5**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

```
hadoop@priyav-VirtualBox:~$ wget https://archive.apache.org/dist/hive/hive-3.1.2/
/apache-hive-3.1.2-bin.tar.gz
--2024-09-02 12:26:15-- https://archive.apache.org/dist/hive/hive-3.1.2/apache-
hive-3.1.2-bin.tar.gz
Resolving archive.apache.org (archive.apache.org)... 65.108.204.189, 2a01:4f9:1a
:a084::2
Connecting to archive.apache.org (archive.apache.org)|65.108.204.189|:443... con
nected.
HTTP request sent, awaiting response... 200 OK
Length: 278813748 (266M) [application/x-gzip]
Saving to: 'apache-hive-3.1.2-bin.tar.gz'

apache-hive-3.1.2-b 100%[=====>] 265.90M  1.20MB/s   in 2m 57s

2024-09-02 12:29:13 (1.50 MB/s) - 'apache-hive-3.1.2-bin.tar.gz' saved [27881374
8/278813748]
```

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

```
hadoop@priyav-VirtualBox:~$ tar -xvf apache-hive-3.1.2-bin.tar.gz
apache-hive-3.1.2-bin/LICENSE
apache-hive-3.1.2-bin/NOTICE
apache-hive-3.1.2-bin/RELEASE_NOTES.txt
apache-hive-3.1.2-bin/binary-package-licenses/asm-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.google.protobuf-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.ibm.icu.icu4j-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.sun.jersey-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/com.thoughtworks.paranamer-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/javax.transaction.transaction-api-
LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/javolution-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/jline-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/NOTICE
apache-hive-3.1.2-bin/binary-package-licenses/org.abego.treelayout.core-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org antlr-LICENSE
apache-hive-3.1.2-bin/binary-package-licenses/org antlr antlr4-LICENSE
```

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

```
#HIVE settings
export HIVE_HOME=/home/hadoop/apache-hive-3.1.2
export PATH=$PATH:$HIVE_HOME/bin
#HIVE settings end
```

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 hiveconfig.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 MySQL by running below commands:

```
$sudomysql
```

Opens command line interface for MySQL and run the below SQL queries to change username and set password

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

```
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

Replace your_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/1QFhB7Kvcat7a4LzDRe6GcmZvalyAxKz/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
```

```

enabled set to false
grunt> vboxuser@tamanna:~$ schematool -initSchema -dbType derby
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/vboxuser/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/Static
LoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/vboxuser/hadoop-3.3.6/share/hadoop/common/lib/slf4j-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]
Metastore connection URL:      jdbc:derby;;databaseName=metastore_db;create=true
Metastore Connection Driver :   org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:      APP
Starting metastore schema initialization to 3.1.0
Initialization script hive-schema-3.1.0.derby.sql

```

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*

```

vboxuser@ubuntu20: ~
vboxuser@ubuntu20: ~/hive/bin
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 0 HDFS Write: 208 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 9.375 seconds
hive> CREATE VIEW myview AS SELECT name, id FROM finance_table;
OK
Time taken: 0.352 seconds
hive> SELECT * FROM myview;
OK
Alice 1
Bob 2
Charlie 3
Time taken: 0.413 seconds, Fetched: 3 row(s)
hive> DESCRIBE finance_table;
OK
id          int
name        string
Time taken: 0.13 seconds, Fetched: 2 row(s)
hive> ALTER TABLE finance_table ADD COLUMNS (age INT);
OK
Time taken: 0.165 seconds
hive> quit;
vboxuser@ubuntu20:~$

```

Result:

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

Exp.No.: 5a**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: 0.57 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: 2.013 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_20240911171244_304f3e60-6937-434c-acb2-d71be2797182
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)
2024-09-11 17:12:54,138 Stage-1 map = 0%,  reduce = 0%
2024-09-11 17:12:57,541 Stage-1 map = 100%,  reduce = 100%
Ended Job = job_local1825573535_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 directory hdfs://localhost:9000/user/hive/warehouse/financials.db/finance_table/.hive-staging_hive_2024-
9-11_17-12-44_558_5675160086864575725-1/-ext-10000
Loading data to table financials.finance_table
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 0 HDFS Write: 208 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 13.965 seconds
```

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;

```
hive> CREATE VIEW myview AS SELECT name, id FROM finance_table;
OK
Time taken: 0.244 seconds
```

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: 0.22 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
age               int
Time taken: 0.729 seconds, Fetched: 3 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: 0.188 seconds
```

Step 10: Quit Hive:

To exit the Hive CLI, simply type: *hive>quit;*

Result:

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