Practical 8

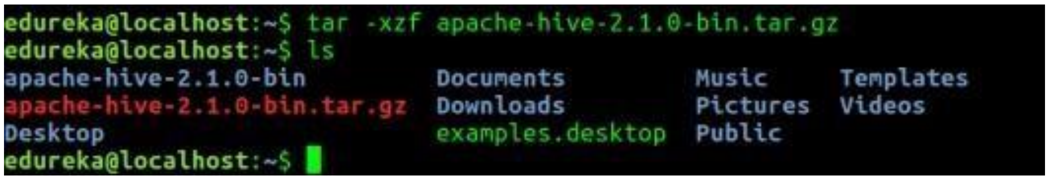
AIM:- Installation of Hive & data aggregation using hive

Theory: Apache Hive is a data warehouse infrastructure that facilitates querying and managing large
data sets which resides in distributed storage system. It is built on top of Hadoop and
developed by Facebook. Hive provides a way to query the data using a SQL-like query
language called HiveQL(Hive query Language).
Internally, a compiler translates HiveQL statements into MapReduce jobs, which are then
submitted to Hadoop framework for execution.

Steps:

Step 1: Download Hive tar.
Command: wget http://archive.apache.org/dist/hive/hive-2.1.0/apache-hive-2.1.0-bin.tar.gz

Step 2: Extract the tar file.
Command: tar -xzf apache-hive-2.1.0-bin.tar.gz
Command: ls

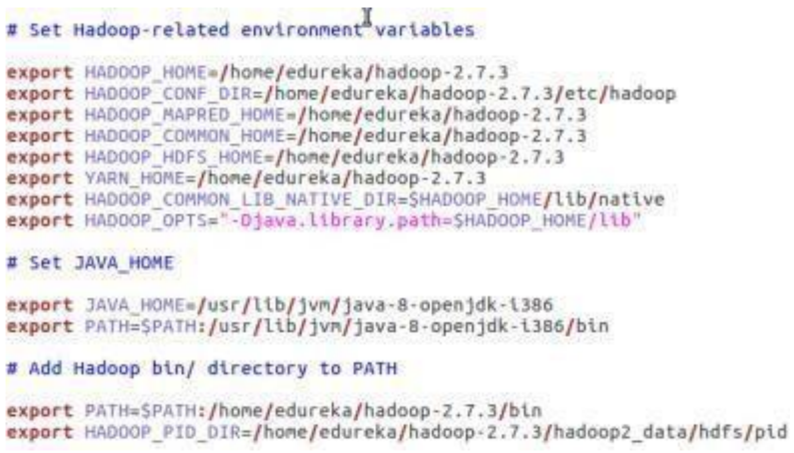


```
edureka@localhost:~$ tar -xzf apache-hive-2.1.0-bin.tar.gz
edureka@localhost:~$ ls
apache-hive-2.1.0-bin         Documents         Music     Templates
apache-hive-2.1.0-bin.tar.gz  Downloads         Pictures  Videos
Desktop                       examples.desktop  Public
edureka@localhost:~$
```

Step 3: Edit the ".bashrc" file to update the environment variables for user.
Command: sudo gedit .bashrc
Add the following at the end of the file:

- ***# Set HIVE_HOME***

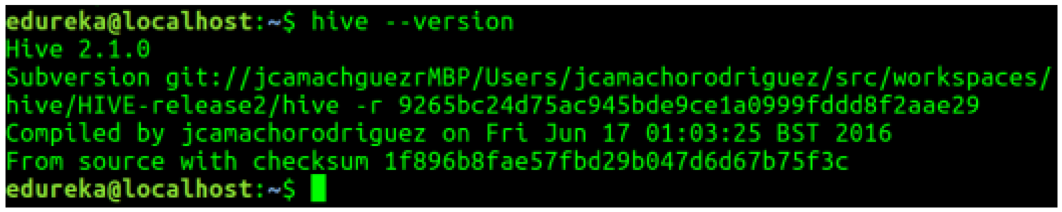
export HIVE_HOME=/home/edureka/apache-hive-2.1.0-bin export PATH=$PATH:/home/edureka/apache-hive-2.1.0-bin/bin Also, make sure that hadoop path is also set.



```
# Set Hadoop-related environment variables

export HADOOP_HOME=/home/edureka/hadoop-2.7.3
export HADOOP_CONF_DIR=/home/edureka/hadoop-2.7.3/etc/hadoop
export HADOOP_MAPRED_HOME=/home/edureka/hadoop-2.7.3
export HADOOP_COMMON_HOME=/home/edureka/hadoop-2.7.3
export HADOOP_HDFS_HOME=/home/edureka/hadoop-2.7.3
export YARN_HOME=/home/edureka/hadoop-2.7.3
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"

# Set JAVA_HOME

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-i386
export PATH=$PATH:/usr/lib/jvm/java-8-openjdk-i386/bin

# Add Hadoop bin/ directory to PATH

export PATH=$PATH:/home/edureka/hadoop-2.7.3/bin
export HADOOP_PID_DIR=/home/edureka/hadoop-2.7.3/hadoop2_data/hdfs/pid
```

Run below command to make the changes work in same terminal.

Command: source .bashrc



```
edureka@localhost:~$ hive --version
Hive 2.1.0
Subversion git://jcamachguezrMBP/Users/jcamachorodriguez/src/workspaces/
hive/HIVE-release2/hive -r 9265bc24d75ac945bde9ce1a0999fddd8f2aae29
Compiled by jcamachorodriguez on Fri Jun 17 01:03:25 BST 2016
From source with checksum 1f896b8fae57fbd29b047d6d67b75f3c
edureka@localhost:~$
```

Step 4: **Check hive version.**

Step 5: Create **Hive** directories within **HDFS**. The directory **'warehouse'** is the location to store the table or data related to hive.

Command:

- hdfs dfs -mkdir -p /user/hive/warehouse

- hdfs dfs -mkdir /tmp

Step 6: Set read/write permissions for table.

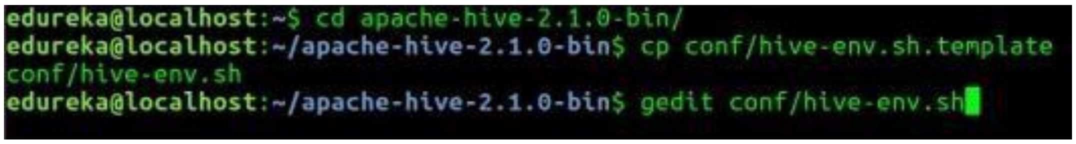
Command:

In this command, we are giving write permission to the group:

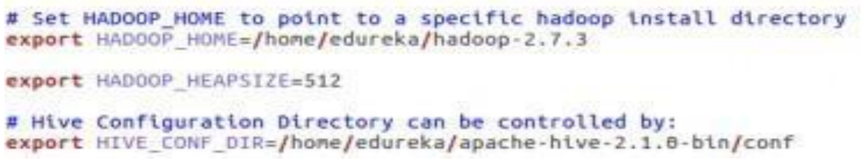
- hdfs dfs -chmod g+w /user/hive/warehouse
- hdfs dfs -chmod g+w /tmp

Step 7: Set **Hadoop** path in **hive-env.sh** **Command:** cd apache-hive-2.1.0-bin/ **Command:** gedit conf/hive-env.sh

Set the parameters as shown in the below snapshot.



```
edureka@localhost:~$ cd apache-hive-2.1.0-bin/
edureka@localhost:~/apache-hive-2.1.0-bin$ cp conf/hive-env.sh.template
conf/hive-env.sh
edureka@localhost:~/apache-hive-2.1.0-bin$ gedit conf/hive-env.sh
```



```
# Set HADOOP_HOME to point to a specific hadoop install directory
export HADOOP_HOME=/home/edureka/hadoop-2.7.3

export HADOOP_HEAPSIZE=512

# Hive Configuration Directory can be controlled by:
export HIVE_CONF_DIR=/home/edureka/apache-hive-2.1.0-bin/conf
```

Step 8: Edit hive-site.xml

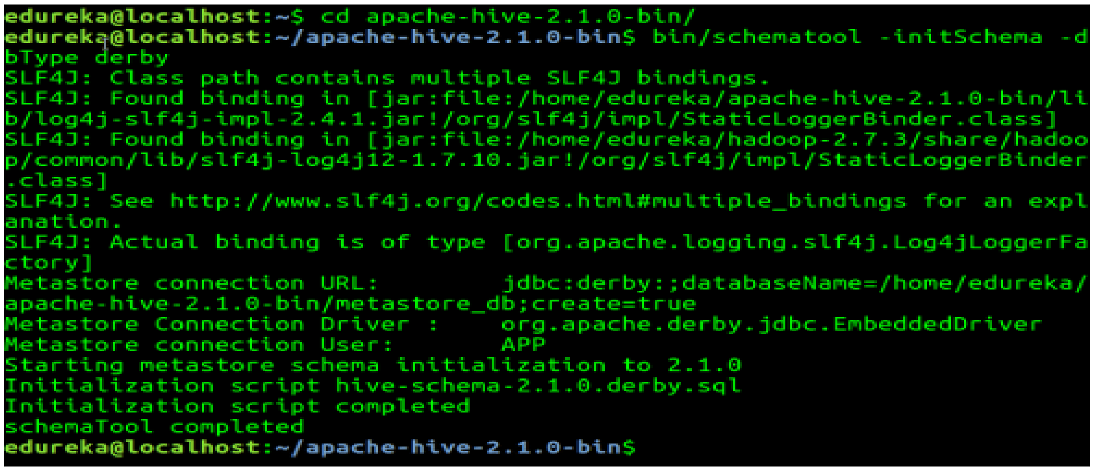
Command: gedit conf/hive-site.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?><!-- 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.
-->
<configuration>
<property>
<name>javax.jdo.option.ConnectionURL</name>
<value>jdbc:derby:;databaseName=/home/edureka/apache-hive-2.1.0-bin/metastore_db;create=true</value>
<description>
JDBC connect string for a JDBC metastore.
To use SSL to encrypt/authenticate the connection, provide database-specific SSL flag in the connection URL.
For example, jdbc:postgresql://myhost/db?ssl=true for postgres database.
</description>
</property>
<property>
<name>hive.metastore.warehouse.dir</name>
```

```
<value>/user/hive/warehouse</value>
<description>location of default database for the warehouse</description>
</property>
<property>
<name>hive.metastore.uris</name>
<value/>
<description>Thrift URI for the remote metastore. Used by metastore client to connect to remote metastore.</description>
</property>
<property>
<name>javax.jdo.option.ConnectionDriverName</name>
<value>org.apache.derby.jdbc.EmbeddedDriver</value>
<description>Driver class name for a JDBC metastore</description>
</property>
<property>
<name>javax.jdo.PersistenceManagerFactoryClass</name>
<value>org.datanucleus.api.jdo.JDOPersistenceManagerFactory</value>
<description>class implementing the jdo persistence</description>
</property>
</configuration>
```

Step 9: By default, Hive uses Derby database. Initialize Derby database.

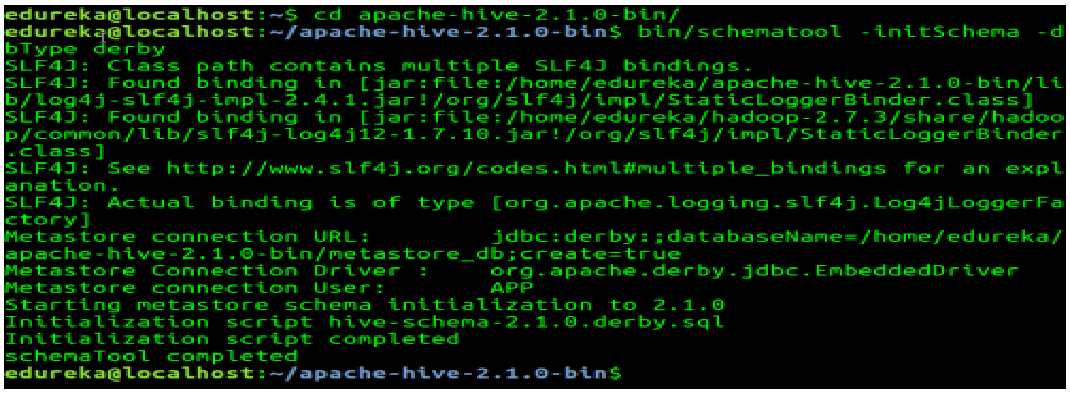
Command: bin/schematool -initSchema -dbType derby



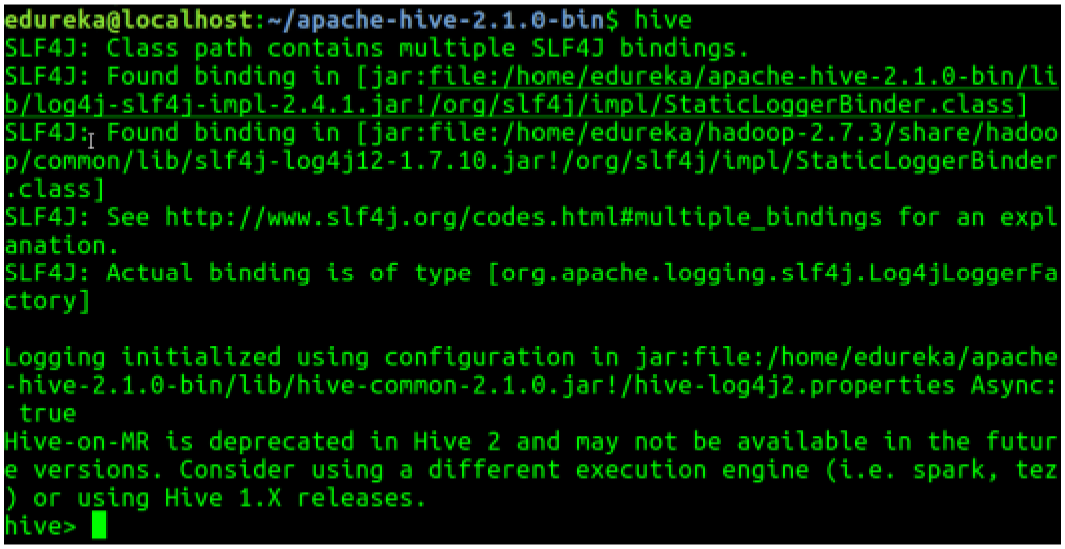
```
edureka@localhost:~$ cd apache-hive-2.1.0-bin/
edureka@localhost:~/apache-hive-2.1.0-bin$ bin/schematool -initSchema -d
bType derby
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/edureka/apache-hive-2.1.0-bin/li
b/log4j-slf4j-impl-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/edureka/hadoop-2.7.3/share/hadoo
p/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder
.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an expl
anation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFa
ctory]
Metastore connection URL:        jdbc:derby:;databaseName=/home/edureka/
apache-hive-2.1.0-bin/metastore_db;create=true
Metastore Connection Driver :    org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:       APP
Starting metastore schema initialization to 2.1.0
Initialization script hive-schema-2.1.0.derby.sql
Initialization script completed
schemaTool completed
edureka@localhost:~/apache-hive-2.1.0-bin$
```

Step 10: Launch **Hive. (In case of Cloudera , we can lunch hive directly on terminal by writing hive command)**

Command: hive



```
edureka@localhost:~$ cd apache-hive-2.1.0-bin/
edureka@localhost:~/apache-hive-2.1.0-bin$ bin/schematool -initSchema -d
bType derby
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/edureka/apache-hive-2.1.0-bin/li
b/log4j-slf4j-impl-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/edureka/hadoop-2.7.3/share/hadoo
p/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder
.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an expl
anation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFa
ctory]
Metastore connection URL:        jdbc:derby:;databaseName=/home/edureka/
apache-hive-2.1.0-bin/metastore_db;create=true
Metastore Connection Driver :    org.apache.derby.jdbc.EmbeddedDriver
Metastore connection User:       APP
Starting metastore schema initialization to 2.1.0
Initialization script hive-schema-2.1.0.derby.sql
Initialization script completed
schemaTool completed
edureka@localhost:~/apache-hive-2.1.0-bin$
```

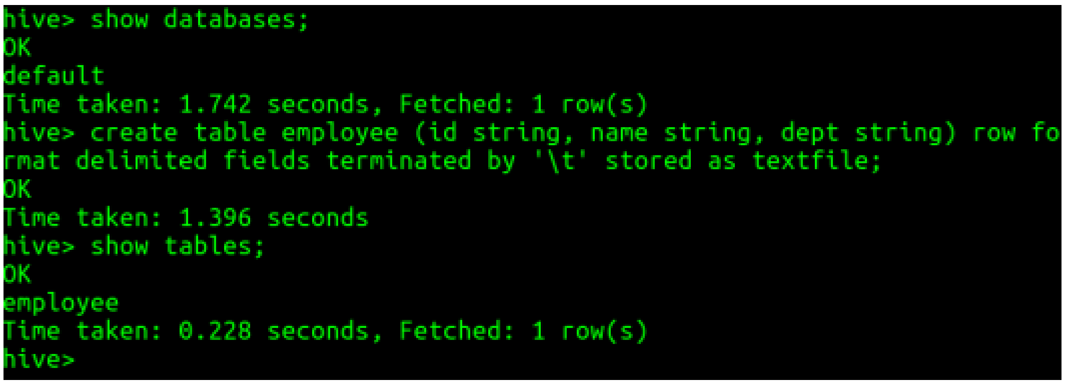


```
edureka@localhost:~/apache-hive-2.1.0-bin$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/edureka/apache-hive-2.1.0-bin/li
b/log4j-slf4j-impl-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/home/edureka/hadoop-2.7.3/share/hadoo
p/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder
.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an expl
anation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFa
ctory]

Logging initialized using configuration in jar:file:/home/edureka/apache
-hive-2.1.0-bin/lib/hive-common-2.1.0.jar!/hive-log4j2.properties Async:
 true
Hive-on-MR is deprecated in Hive 2 and may not be available in the futur
e versions. Consider using a different execution engine (i.e. spark, tez
) or using Hive 1.X releases.
hive>
```

Step 11: Run few queries in Hive shell.

Command: show databases;

Command: create table employee (id string, name string, dept string) row format delimited fields terminated by ‘ ‘ stored as textfile;



```
hive> show databases;
OK
default
Time taken: 1.742 seconds, Fetched: 1 row(s)
hive> create table employee (id string, name string, dept string) row fo
rmat delimited fields terminated by '\t' stored as textfile;
OK
Time taken: 1.396 seconds
hive> show tables;
OK
employee
Time taken: 0.228 seconds, Fetched: 1 row(s)
hive>
```

Command: show tables;

Step 12: To exit from **Hive:**

Command: exit;

Hive Commands :

Data Definition Language (DDL)

DDL statements are used to build and modify the tables and other objects in the database.

Example :

CREATE, DROP, TRUNCATE, ALTER, SHOW, DESCRIBE Statements.

Data Manipulation Language (DML)

DML statements are used to retrieve, store, modify, delete, insert and update data in the database.

Example :

LOAD, INSERT Statements. Syntax :

LOAD data <LOCAL> inpath <file path> into table [tablename]

The Load operation is used to move the data into corresponding Hive table. If the keyword **local** is specified, then in the load command will give the local file system path. If the keyword local is not specified we have to use the HDFS path of the file.