

## MSDS610 Week 4 Hive Assignment - Nathan Worsham

### Installing Hive

Again on this week's assignment I started by powering up my VM from week 1. Since the rest of the software on this VM is installed at /home/hadoop I decided to make the Hive installation location no different and unpacked the archive there along with changing ownership of the files.

```
[root@week1 ~]# cd /home/hadoop/
[root@week1 ~]# ls
hadoop hadoopdata hbase hbase-1.1.2 shakespeareoutput.txt shakespeare.txt zookeeper
[root@week1 ~]# wget http://apache.arvixe.com/hive/hive-1.2.1/apache-hive-1.2.1-bin.tar.gz
2016-02-01 21:52:51 -- http://apache.arvixe.com/hive/hive-1.2.1/apache-hive-1.2.1-bin.tar.gz
Resolving apache.arvixe.com (apache.arvixe.com)... 198.58.87.82
Connecting to apache.arvixe.com (apache.arvixe.com)[198.58.87.82]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 92834839 (88M) [application/x-gzip]
Saving to: 'apache-hive-1.2.1-bin.tar.gz'

100%[=====] 92834839 100%
2016-02-01 21:53:00 (6.07 MB/s) - 'apache-hive-1.2.1-bin.tar.gz' saved [92834839/92834839]

[root@week1 ~]# tar -xvf apache-hive-1.2.1-bin.tar.gz
apache-hive-1.2.1-bin/NOTICE
apache-hive-1.2.1-bin/LICENSE
apache-hive-1.2.1-bin/README.txt
apache-hive-1.2.1-bin/RELEASE_NOTES.txt
apache-hive-1.2.1-bin/examples/files/exp.txt
apache-hive-1.2.1-bin/examples/files/type_evolution.sql
apache-hive-1.2.1-bin/examples/files/extract_state_partial.txt
apache-hive-1.2.1-bin/examples/files/lineitem.txt

[root@week1 ~]# ls -l
total 95464
drwxr-xr-x. 8 root root 4096 Feb 1 21:53 apache-hive-1.2.1-bin
-rw-r--r--. 1 root root 92834839 Jun 26 2015 apache-hive-1.2.1-bin.tar.gz
drwxr-xr-x. 18 hadoop hadoop 4096 Jan 12 18:37 hadoopdata
drwxr-xr-x. 3 hadoop hadoop 17 Jan 12 17:44 hadoopdata
drwxr-xr-x. 8 hadoop hadoop 4096 Jan 25 19:36 hbase
drwxr-xr-x. 8 hadoop hadoop 4096 Jan 25 19:12 hbase-1.1.2
-rw-r--r--. 1 hadoop hadoop 356409 Jan 12 20:04 shakespeareoutput.txt
-rw-r--r--. 1 hadoop hadoop 4538523 Aug 16 2011 shakespeare.txt
drwxr-xr-x. 4 hadoop hadoop 51 Jan 25 22:30 zookeeper
[root@week1 ~]# chown hadoop:hadoop apache-hive-1.2.1-bin
[root@week1 ~]# ln -s /home/hadoop/apache-hive-1.2.1-bin hive
[root@week1 ~]# ls -l
total 95464
drwxr-xr-x. 8 hadoop hadoop 4096 Feb 1 21:53 apache-hive-1.2.1-bin
-rw-r--r--. 1 root root 92834839 Jun 26 2015 apache-hive-1.2.1-bin.tar.gz
drwxr-xr-x. 18 hadoop hadoop 4096 Jan 12 18:37 hadoopdata
drwxr-xr-x. 3 hadoop hadoop 17 Jan 12 17:44 hadoopdata
drwxr-xr-x. 8 hadoop hadoop 4096 Jan 25 19:36 hbase
drwxr-xr-x. 8 hadoop hadoop 4096 Jan 25 19:12 hbase-1.1.2
-rwxr-xr-x. 1 root root 34 Feb 1 21:54 hive -> /home/hadoop/apache-hive-1.2.1-bin
-rw-r--r--. 1 hadoop hadoop 356409 Jan 12 20:04 shakespeareoutput.txt
-rw-r--r--. 1 hadoop hadoop 4538523 Aug 16 2011 shakespeare.txt
drwxr-xr-x. 4 hadoop hadoop 51 Jan 25 22:30 zookeeper
[root@week1 ~]#
```

Next I configured the environment variables to include HIVE\_HOME as instructed.

```
[hadoop@week1 ~]$ vi .bashrc
[hadoop@week1 ~]$ tail -5 .bashrc
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin

export HIVE=/home/hadoop/hive
export PATH=$PATH:$HIVE/bin
[hadoop@week1 ~]$ ..bashrc
[hadoop@week1 ~]$ printenv PATH
/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/opt/jdk1.8.0_65/bin:/h
5/bin:/home/hadoop/hadoop/sbin:/home/hadoop/hadoop/bin:/home/hadoop/hive/bin:/
```

Now the instructions wanted directories and permissions established on such directories within the HDFS. Not realizing, I tried making the /tmp directory only to find one already there. It seemed the rights were already correct but equally did not seem it would hurt to make sure with the `chmod` command. When I tried to create layers of directories with the command they provided `hadoop fs -mkdir /user/hive/warehouse` it complained of "No such file or directory". Adding the `-p` option like the normal `mkdir` command to "Create intermediate directories as required" worked in the HDFS. I then added the rights to the new directory as well.

```
[hadoop@week1 ~]$ /home/hadoop/hadoop/sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/hadoop/hadoop/logs/hadoop-hadoop-namenode-week1.out
localhost: starting datanode, logging to /home/hadoop/hadoop/logs/hadoop-hadoop-datanode-week1.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/hadoop/hadoop/logs/hadoop-hadoop-secondarynamenode
[hadoop@week1 ~]$ hadoop fs -mkdir /tmp
mkdir: /tmp: File exists
[hadoop@week1 ~]$ hadoop fs -ls /tmp
Found 1 items
drwxr-xr-x. - hadoop supergroup 0 2016-01-12 18:45 /tmp/hadoop-yarn
[hadoop@week1 ~]$ hadoop fs -mkdir /user/hive/warehouse
mkdir: /user/hive/warehouse: No such file or directory
[hadoop@week1 ~]$ hadoop fs -ls /user/hive/warehouse
ls: /user/hive/warehouse: No such file or directory
[hadoop@week1 ~]$ hadoop fs -mkdir -p /user/hive/warehouse
[hadoop@week1 ~]$ hadoop fs -ls /user/hive/warehouse
Found 1 items
drwxr-xr-x. - hadoop supergroup 0 2016-02-01 22:15 /user/hive/warehouse
[hadoop@week1 ~]$ hadoop fs -chmod g+w /tmp
[hadoop@week1 ~]$ hadoop fs -chmod g+w /user/hive/warehouse
```

I was now ready to start hive, which started up fine. The instructions indicate that HiveCLI is deprecated and that instead "HiveServer2 and Beeline" should be used in it's place. When I tried the first command `$HIVE_HOME/bin/hiveserver2` the shell did not come back, it seems this command starts up a process. I used CTRL-Z to stop it after CTRL-C could not break it. Reading further it seems the instructions offer a single command to run both `-$HIVE_HOME/bin/beeline -u jdbc:hive2://` but when I ran it I received errors about "initialising [sic] the database".

```

[hadoop@week1 ~]$ /home/hadoop/hive/bin/hive
Logging initialized using configuration in jar:file:/home/hadoop/apache-hive-1.2.1-bin/lib/hive-common-1
hive> quit;
[hadoop@week1 ~]$ /home/hadoop/hive/bin/hiveserver2
^C
^Z
[1]+  Stopped                  /home/hadoop/hive/bin/hiveserver2
[hadoop@week1 ~]$ /home/hadoop/hive/bin/beeline -u jdbc:hive2://
Connecting to jdbc:hive2://
ERROR Datastore.Schema: Failed initialising database.
Unable to open a test connection to the given database. JDBC url = jdbc:derby://databaseName=metastore_db
your database after your app). Original Exception: -----
java.sql.SQLException: Failed to start database 'metastore_db' with class loader sun.misc.Launcher$AppCl
    at org.apache.derby.impl.jdbc.SQLExceptionFactory40.getSQLException(Unknown Source)
    at org.apache.derby.impl.jdbc.Util.newEmbedSQLException(Unknown Source)
    at org.apache.derby.impl.jdbc.Util.getNextException(Unknown Source)

```

Thinking that when I killed the previous command there was likely some fallout, so I rebooted the VM and tried the command again, this time I was received a new error about not being able to create a directory in /tmp inside the HDFS.

```

[hadoop@week1 ~]$ /home/hadoop/hive/bin/beeline -u jdbc:hive2://
Connecting to jdbc:hive2://
Error applying authorization policy on hive configuration: org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.hdfs.server.namenode.SafeModeException): Cannot create directory /tmp/hive/hadoop/48
bc90-4dd-9c68-30bc923660eb. Name node is in safe mode.
The reported blocks 23 has reached the threshold 0.9990 of total blocks 23. The number of live datanodes 1 has reached the minimum number 0. In safe mode extension, Safe mode will be turned off autom
y in 4 seconds.
    at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkNameNodeSafeMode(FSNamesystem.java:1327)
    at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.mkdirs(FSNamesystem.java:3899)
    at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.mkdirs(NameNodeRpcServer.java:978)
    at org.apache.hadoop.hdfs.protocolPB.ClientNameNodeProtocolServerSideTranslatorPB.mkdirs(ClientNameNodeProtocolServerSideTranslatorPB.java:622)
    at org.apache.hadoop.hdfs.protocol.proto.ClientNameNodeProtocolProtos$ClientNameNodeProtocol$2.callBlockingMethod(ClientNameNodeProtocolProtos.java)
    at org.apache.hadoop.ipc.ProtobufRpcEngine$Server$ProtobufRpcInvoker.call(ProtobufRpcEngine.java:616)
    at org.apache.hadoop.ipc.RPC$Server.call(RPC.java:969)
    at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2049)
    at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:2045)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:422)
    at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1657)
    at org.apache.hadoop.ipc.Server$Handler.run(Server.java:2049)
Beeline version 1.2.1 by Apache Hive
0: jdbc:hive2:// (closed)>

```

It would seem that similarly to the instruction using the `mkdir` command, the `chmod` command instructions did not account for recursive actions. This is because when I did a file listing in the HDFS of /tmp/hive I found that the group portion of the permissions was lacking the write permission the command was supposed to give. So when I ran the command again, but this time targeting that folder, the rights were now there and when I started the combined command for HiveServer2 and Beeline, I was rewarded a CLI command prompt with no errors.

```

[hadoop@week1 ~]$ $HADOOP_HOME/bin/hadoop fs -chmod g+w /tmp
[hadoop@week1 ~]$ hadoop fs -ls /tmp/hive/hadoop
[hadoop@week1 ~]$ hadoop fs -ls /tmp/hive/
Found 1 items
drwx----- - hadoop supergroup          0 2016-02-02 03:24 /tmp/hive/hadoop
[hadoop@week1 ~]$ $HADOOP_HOME/bin/hadoop fs -chmod g+w /tmp/hive/hadoop
[hadoop@week1 ~]$ hadoop fs -ls /tmp/hive/
Found 1 items
drwx-v---- - hadoop supergroup          0 2016-02-02 03:24 /tmp/hive/hadoop
[hadoop@week1 ~]$ /home/hadoop/hive/bin/beeline -u jdbc:hive2://
Connecting to jdbc:hive2://
Connected to: Apache Hive (version 1.2.1)
Driver: Hive JDBC (version 1.2.1)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 1.2.1 by Apache Hive
0: jdbc:hive2://>

```

## Hive Test Use Case

Following the instructions for the test use case using MovieLens data, I created the table using Hive CLI (despite the earlier mention of deprecation, as it seems simpler to use). I then exited out of the Hive CLI and downloaded the MovieLens data.

```

[hadoop@week1 ~]$ ./home/hadoop/hive/bin/hive
Logging initialized using configuration in jar:file:/home/hadoop/apache-hive-1.2.1-bin/lib/hive-common-1.2.1.jar!/hive-log4j.properties
hive> CREATE TABLE u_data (
  >   userid INT,
  >   movieid INT,
  >   rating INT,
  >   unixtime STRING)
  > ROW FORMAT DELIMITED
  > FIELDS TERMINATED BY '\t'
  > STORED AS TEXTFILE;
OK
Time taken: 1.711 seconds
hive> quit;
[hadoop@week1 ~]$ ls
apache-hive-1.2.1-bin  derby.log  hadoop  hadoopdata  hbase  hbase-1.1.2  hive  metastore_db  shakespeareoutput.txt
shakespeare.txt  zookeeper
[hadoop@week1 ~]$ wget http://files.grouplens.org/datasets/movielens/ml-100k.zip
--2016-02-02 21:21:18-- http://files.grouplens.org/datasets/movielens/ml-100k.zip
Resolving files.grouplens.org (files.grouplens.org)... 128.101.34.146
Connecting to files.grouplens.org (files.grouplens.org)[128.101.34.146]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4924029 (4.7M) [application/zip]
Saving to: 'ml-100k.zip'

100%[=====] 4,924,029  2.32MB/s  in 2.0s

2016-02-02 21:21:20 (2.32 MB/s) - 'ml-100k.zip' saved [4924029/4924029]

[hadoop@week1 ~]$

```

After the download, I tried to unpack the data but realized I needed to first go and grab the package for unzip as it was not installed. After getting it installed and unpacking the data, I was ready to load the data into the table using the LOAD DATA function to load the u.data file. According to the README file, the u.data file is the "full u data set, 100000 ratings by 943 users on 1682 items".

```

[hadoop@week1 ~]$ unzip ml-100k.zip
Archive: ml-100k.zip
  creating: ml-100k/
  inflating: ml-100k/allbut.pl
  inflating: ml-100k/aku.sh
  inflating: ml-100k/README
  inflating: ml-100k/u.data
  inflating: ml-100k/u.genre
  inflating: ml-100k/u.info
  inflating: ml-100k/u.item
  inflating: ml-100k/u.occupation
  inflating: ml-100k/u.user
  inflating: ml-100k/u1.base
  inflating: ml-100k/u1.test
  inflating: ml-100k/u2.base
  inflating: ml-100k/u2.test
  inflating: ml-100k/u3.base
  inflating: ml-100k/u3.test
  inflating: ml-100k/u4.base
  inflating: ml-100k/u4.test
  inflating: ml-100k/u5.base
  inflating: ml-100k/u5.test
  inflating: ml-100k/ua.base
  inflating: ml-100k/ua.test
  inflating: ml-100k/ub.base
  inflating: ml-100k/ub.test
[hadoop@week1 ~]$ ls
apache-hive-1.2.1-bin  hadoop  hbase  hive  ml-100k  shakespeareou
derby.log  hadoopdata  hbase-1.1.2  metastore_db  ml-100k.zip  shakespeare.t
[hadoop@week1 ~]$ hive/bin/hive

Logging initialized using configuration in jar:file:/home/hadoop/apache-hive-1.2.1-bin/lib
hive> LOAD DATA LOCAL INPATH '/home/hadoop/ml-100k/u.data'
  > OVERWRITE INTO TABLE u_data;
Loading data to table default.u_data
Table default.u_data stats: [numFiles=1, numRows=0, totalSize=1979173, rawDataSize=0]
OK
Time taken: 2.795 seconds
hive>

```

Next I counted the rows in the table. Here I can see that it is running a mapreduce job to accomplish the task. I admit I was a little concerned this wasn't going to return anything because the previous step when I loaded the data in had a message about "numRows=0", but the query came back with the correct number of 100000.

```

hive> SELECT COUNT(*) FROM u_data;
Query ID = hadoop_20160202213515_f8695084-c435-43da-a9c8-cd83c7d576a9
Total jobs = 1
Launching Job 1 out of 1
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)
2016-02-02 21:35:18.252 Stage-1 map = 0%, reduce = 0%
2016-02-02 21:35:19.268 Stage-1 map = 100%, reduce = 100%
Ended Job = job_local1388229904_0001
MapReduce Jobs Launched:
Stage-Stage-1:  HDFS Read: 3958346 HDFS Write: 3958346 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
100000
Time taken: 3.778 seconds, Fetched: 1 row(s)
hive>

```

Now the instructions were to make a python script for the "mapper script". I went ahead and created

a scripts directory just for tidiness and built the script there. Looks like the script removes line endings, then segments out each line placing each part in a variable but most importantly the unixtime variable. It then takes that unixtime variable and converts it to a day of the week as a number (1-7) and then reformats the line back out again separated by tabs. Next I created another table and added the python script as a resource.

```
[hadoop@week1 ~]$ hive/bin/hive
Logging initialized using configuration in jar:file:/home/
hive> CREATE TABLE u_data_new (
>   userid INT,
>   moviewid INT,
>   rating INT,
>   weekday INT)
> ROW FORMAT DELIMITED
> FIELDS TERMINATED BY '\t';
OK
Time taken: 1.499 seconds
hive> add FILE /home/hadoop/scripts/weekday_mapper.py
> ;
Added resources: [/home/hadoop/scripts/weekday_mapper.py]
```

I ran the command that would "transform" the data from the original table to the data into the new table with the day of the week as a number. Though the thought did occur to me of why the exercise went to the the trouble of making another table instead of just adding a column with the converted days of the week value to the original table? After this completed I was then able to count from this data and GROUP BY the day of the week.

```
Hive> INSERT OVERWRITE TABLE u_data_new
> SELECT
>   TRANSFORM (userid, moviewid, rating, unixtime)
>   USING 'python weekday_mapper.py'
>   AS (userid, moviewid, rating, weekday)
> FROM u_data;
Query ID = hadoop_20160202215043_7c933dd7-b04d-4e85-6ad1-35a515999069
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks is set to 0 since there's no reduce operator
Job running in-process (local Hadoop)
2016-02-02 21:50:47,193 Stage-1 map = 0%, reduce = 0%
2016-02-02 21:50:49,210 Stage-1 map = 100%, reduce = 0%
Ended Job = job_local1981465775_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:9000/user/hive/warehouse/u_data_new/hive-staging_hive_2016-02-02_21-50-
Loading data to table default.u_data_new
Table default.u_data_new stats: [numFiles=1, numRows=100000, totalSize=1179173, rawDataSize=1079173]
MapReduce Jobs Launched:
Stage-Stage-1: HDFS Read: 1979173 HDFS Write: 1179256 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
Time taken: 6.513 seconds
Hive> SELECT weekday, COUNT(*)
> FROM u_data_new
> GROUP BY weekday;
Query ID = hadoop_20160202215106_1eab2f7c-1760-4f8d-6de6-59465b6420da
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 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)
2016-02-02 21:51:08,044 Stage-1 map = 100%, reduce = 100%
Ended Job = job_local1958194460_0002
MapReduce Jobs Launched:
Stage-Stage-1: HDFS Read: 6316858 HDFS Write: 2358512 SUCCESS
Total MapReduce CPU Time Spent: 0 msec
OK
1      13346
2      14389
3      15928
4      13348
5      10495
6      12229
7      12363
Time taken: 1.762 seconds, Fetched: 7 row(s)
Hive>
```

This shows that the most ratings happened on Fridays, followed by Wednesdays. Surprisingly, weekends were the least popular, but perhaps this is when the subjects watched the content in order to give it a review later in the week. Last I decided to try out some other HQL commands to show and describe the tables that now existed.

```

[hadoop@week1 ~]$ hadoop fs -ls /user/hive/warehouse
Found 2 items
drwxrwxr-x - hadoop supergroup      0 2016-02-02 21:31 /user/hive/warehouse/u_data
drwxrwxr-x - hadoop supergroup      0 2016-02-02 21:50 /user/hive/warehouse/u_data_new
[hadoop@week1 ~]$ /home/hadoop/hive/bin/hive
Logging initialized using configuration in jar:file:/home/hadoop/apache-hive-1.2.1-bin/lib/hive-udf.jar!/hive-udf-hdfs.jar
hive> SHOW TABLES;
OK
u_data
u_data_new
Time taken: 0.949 seconds, Fetched: 2 row(s)
hive> DESCRIBE u_data
+ ;
OK
userid          int
movieid         int
rating          int
unixtime        string
Time taken: 0.571 seconds, Fetched: 4 row(s)
hive> DESCRIBE u_data_new;
OK
userid          int
movieid         int
rating          int
weekday         int
Time taken: 0.129 seconds, Fetched: 4 row(s)

```

## References

cwiki.apache.org, 2016. Retrieved from  
<https://cwiki.apache.org/confluence/display/Hive/GettingStarted#GettingStarted-InstallingHivefromaStableRelease>