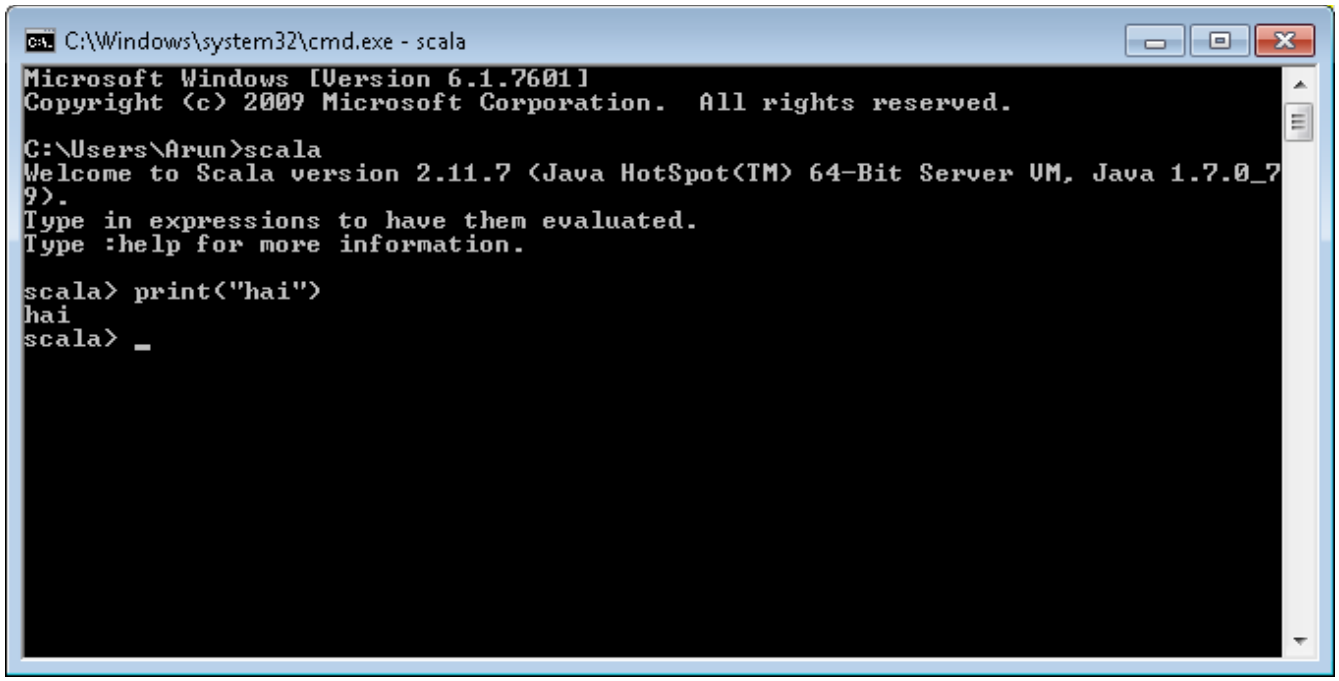
APACHE SPARK

STEP 1:

install scala



path environment variable:

C:\Users\Arun>path
PATH=C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Program Files\Intel\WiFi\bin\;C:\Program Files\Common Files\Intel\WirelessCommon\;C:\Program Files (x86)\Skype\Phone\;C:\apache-maven-3.3.9\bin;C:\protoc;C:\Program Files\Microsoft SDKs\Windows\v7.1\bin;C:\Program Files\Git\bin;C:\Java\jdk1.7.0_79\bin;C:\Anaconda2;C:\Anaconda2\Library\bin;C:\Anaconda2\Scripts;C:\Program Files\R\R-3.2.3\bin;C:\spark-1.6.0-bin-hadoop2.3\bin;C:\scala-2.11.7\bin;C:\SBT-0.13\bin;C:\hadoop-2.2.0\bin;C:\hadoop-2.2.0\sbin

STEP 2:

install spark for hadoop version if using 2.2 download pre-built 2.3 since hadop prebuilt 2.3 is lowest version for hadoop with spark spark-1.6.0-bin-hadoop2.3.tgz

STEP 3:

start spark shell

run spark in the same path where I used hadoop input files to run mapreduce program.

C:\HADOOPOUTPUT>

```
C:\Windows\system32\cmd.exe - spark-shell

C:\HADOOPOUTPUT>spark-shell
log4j:WARN No appenders could be found for logger (org.apache.hadoop.metrics2.li
b.MutableMetricsFactory).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more in
fo.
Using Spark's repl log4j profile: org/apache/spark/log4j-defaults-repl.propertie
s
To adjust logging level use sc.setLogLevel("INFO")
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_79)
Type in expressions to have them evaluated.
Type :help for more information.
Spark context available as sc.
16/03/09 22:00:04 WARN General: Plugin (Bundle) "org.datanucleus.store.rdbms" is
 already registered. Ensure you dont have multiple JAR versions of the same plug
in in the classpath. The URL "file:/C:/spark-1.6.0-bin-hadoop2.3/bin/../lib/data
nucleus-rdbms-3.2.9.jar" is already registered, and you are trying to register a
n identical plugin located at URL "file:/C:/spark-1.6.0-bin-hadoop2.3/lib/datanu
cleus-rdbms-3.2.9.jar."
16/03/09 22:00:04 WARN General: Plugin (Bundle) "org.datanucleus.api.jdo" is alr
eady registered. Ensure you dont have multiple JAR versions of the same plugin i
n the classpath. The URL "file:/C:/spark-1.6.0-bin-hadoop2.3/lib/datanucleus-api
-jdo-3.2.6.jar" is already registered, and you are trying to register an identic
al plugin located at URL "file:/C:/spark-1.6.0-bin-hadoop2.3/bin/../lib/datanucl
eus-api-jdo-3.2.6.jar."
16/03/09 22:00:04 WARN General: Plugin (Bundle) "org.datanucleus" is already reg
istered. Ensure you dont have multiple JAR versions of the same plugin in the cl
asspath. The URL "file:/C:/spark-1.6.0-bin-hadoop2.3/bin/../lib/datanucleus-core
-3.2.10.jar" is already registered, and you are trying to register an identical
plugin located at URL "file:/C:/spark-1.6.0-bin-hadoop2.3/lib/datanucleus-core-3
.2.10.jar."
16/03/09 22:00:04 WARN Connection: BoneCP specified but not present in CLASSPATH
 (or one of dependencies)
16/03/09 22:00:05 WARN Connection: BoneCP specified but not present in CLASSPATH
 (or one of dependencies)
```

SPARK run mapreduce using

1) Scala

2)Python or R.

3)Java

RDD RESILIENT DISTRIBUTED DATASET

primary data abstraction in Spark.
Resilient - fault tolerant
Distributed - across cluster
Dataset- collection of partition data

Features of RDD:

- **Immutable**, i.e. it does not change once created.
- **Lazy evaluated**, i.e. the data inside RDD is not available or transformed until an action is executed that triggers the execution.
- **Cacheable**, i.e. you can hold all the data in a persistent "storage" like memory (default and the most preferred) or disk (the least preferred due to access speed).
- **Parallel**, i.e. process data in parallel

Each RDD is characterized by five main properties:

- An array of partitions that a dataset is divided to
- A function to do a computation for a partition
- List of parent RDDs
- An optional partitioner that defines how keys are hashed, and the pairs partitioned (for key-value RDDs)
- Optional preferred locations, i.e. hosts for a partition where the data will have been loaded.

This RDD abstraction supports an expressive set of operations without having to modify scheduler for each one.

An RDD is a named (by **name**) and uniquely identified (by **id**) entity inside a SparkContext. It lives in a SparkContext and as a SparkContext creates a logical boundary, RDDs can't be shared between SparkContexts (see SparkContext and RDDs).

TRANSFORMATIONS

A **transformation** is a lazy operation on a RDD that returns another RDD, like below

- `map`,
- `flatMap`,
- `filter`,
- `reduceByKey`,
- `join`,
- `cogroup`, etc.

Spark run on top of hadoop with 3 ways

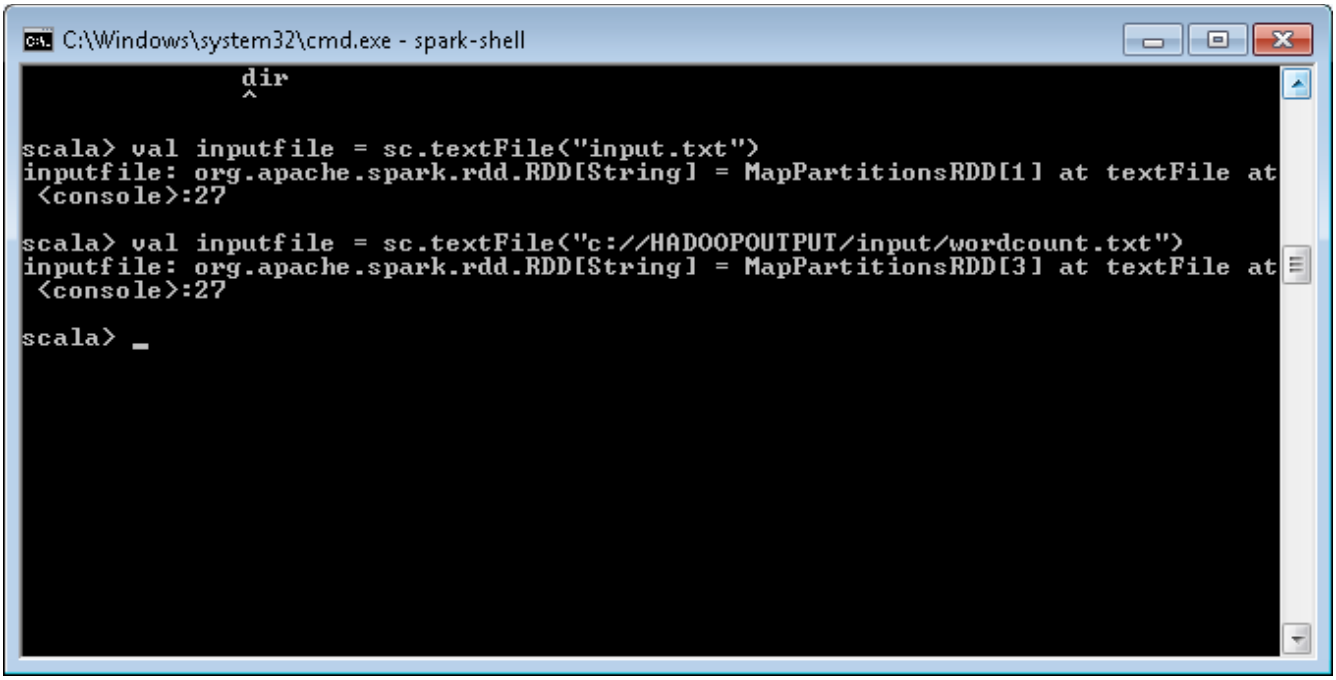
1)Spark standalone

2)Hadoop YARN

3)Spark on Mapreduce (SIMR) -plugin which allows spark to run on top of hadoop without installation of anything and without any privileges.

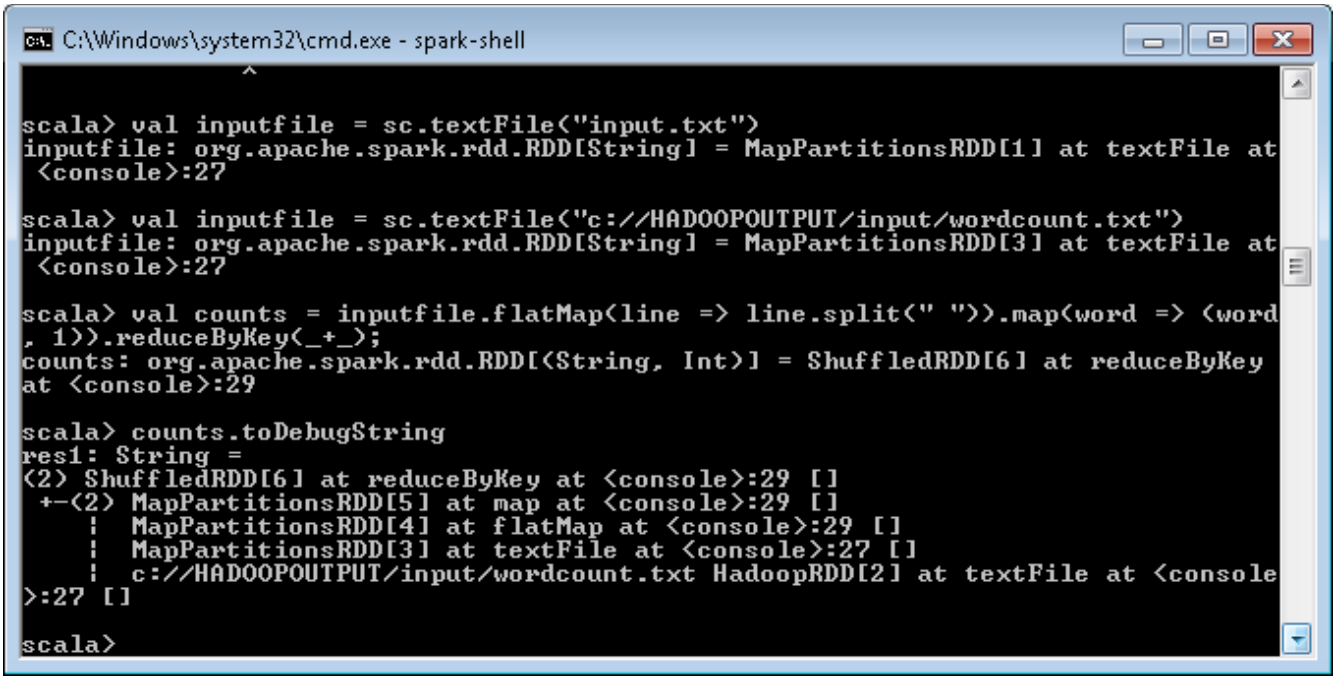
1.SPARK STANDALONE:

see below screenshot below command create RDD file with arr index 1:



Execute Mapreduce:

```
scala> val counts = inputfile.flatMap(line => line.split(" ")).map(word => (word,
1)).reduceByKey(_+_);
```

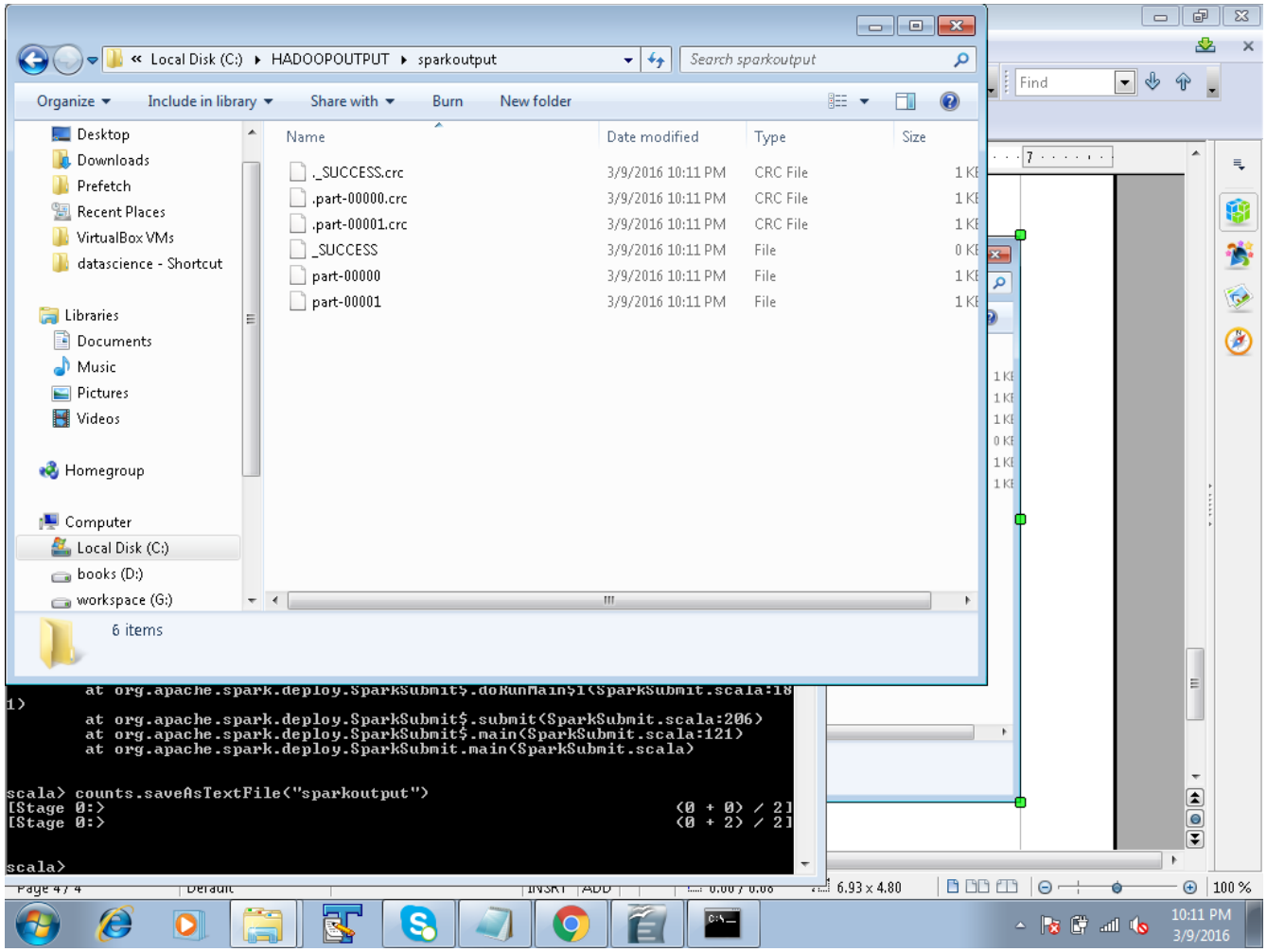


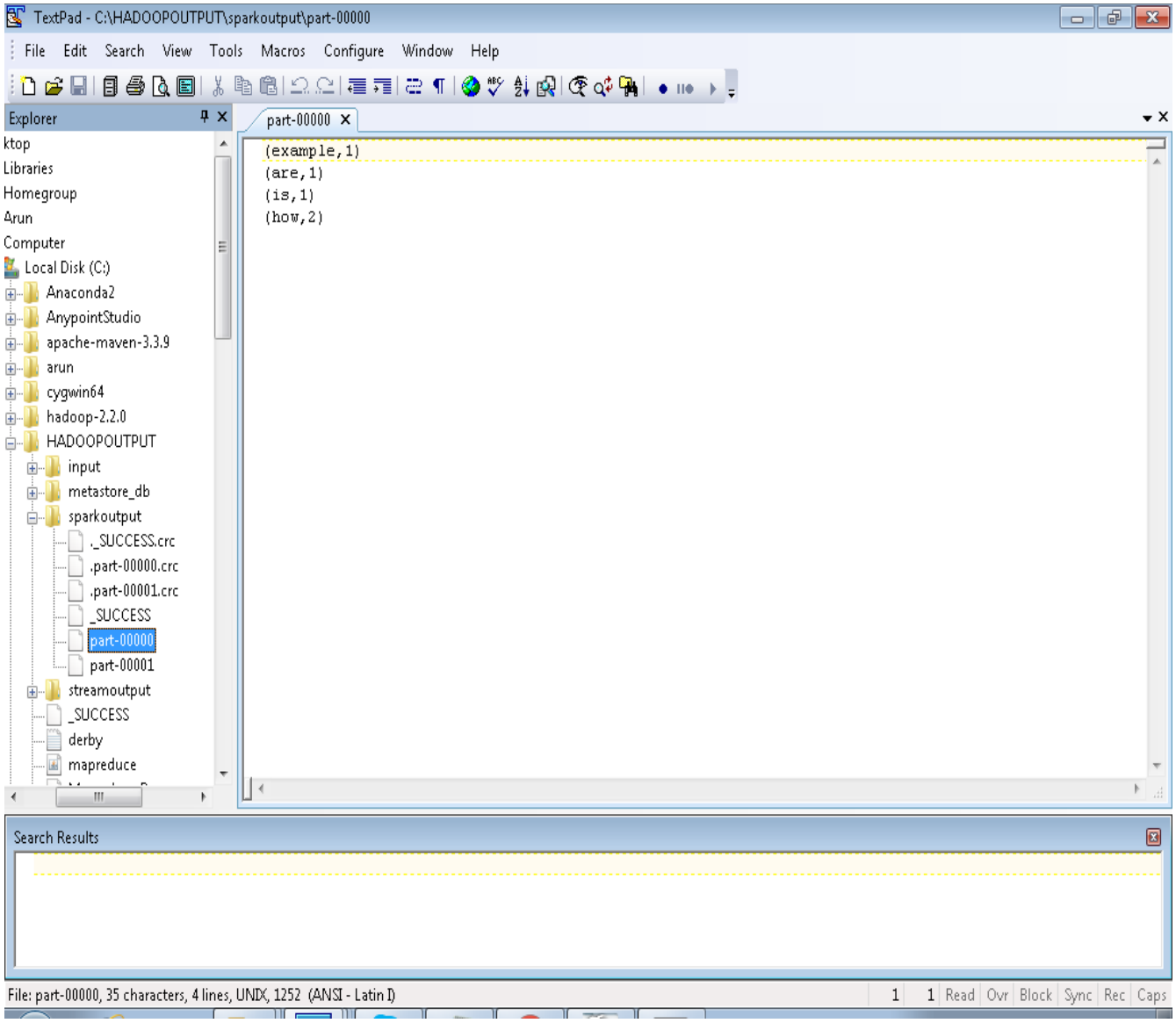
save result:

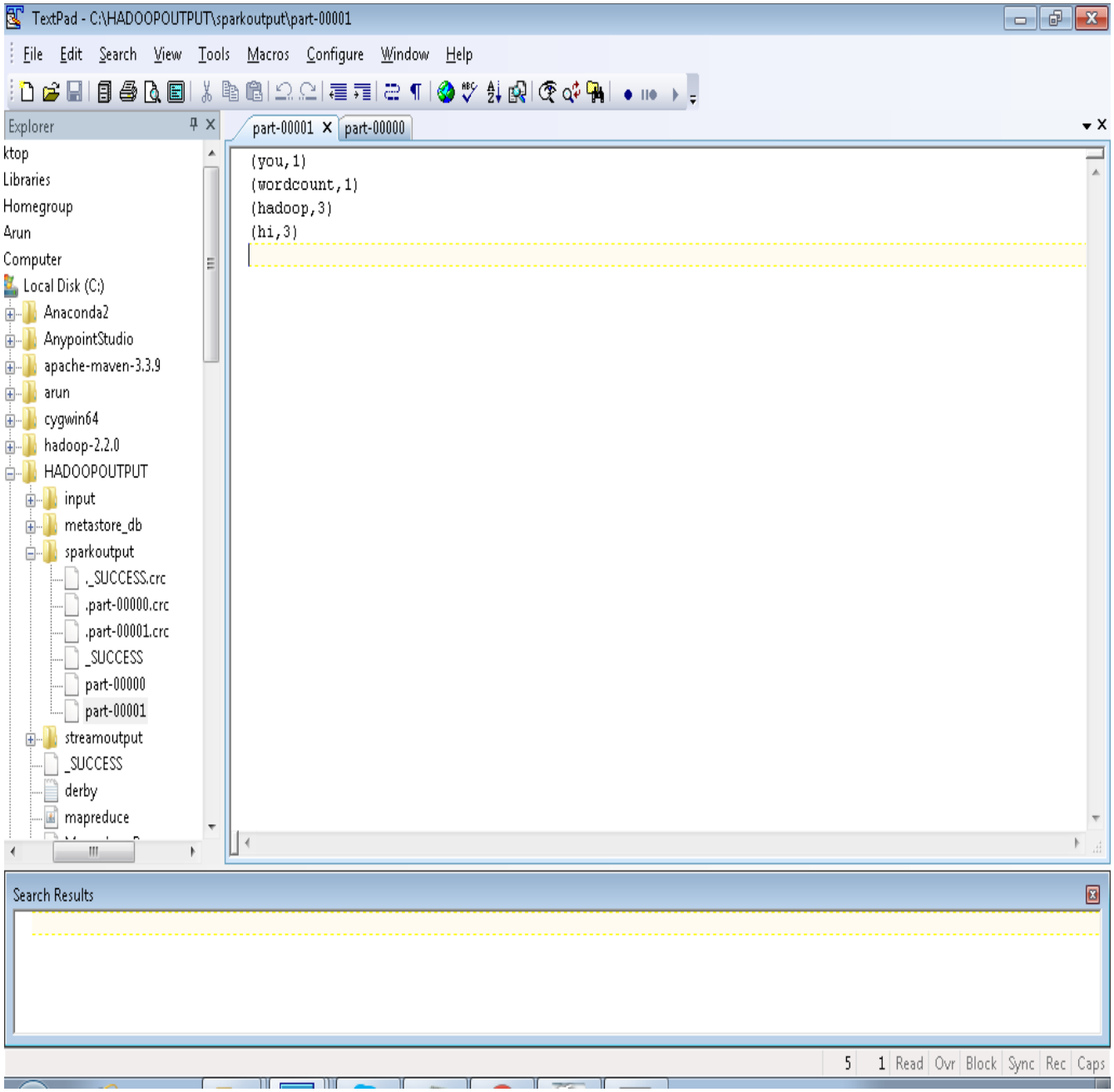
```
scala> counts.cache()
```

save result to output file:

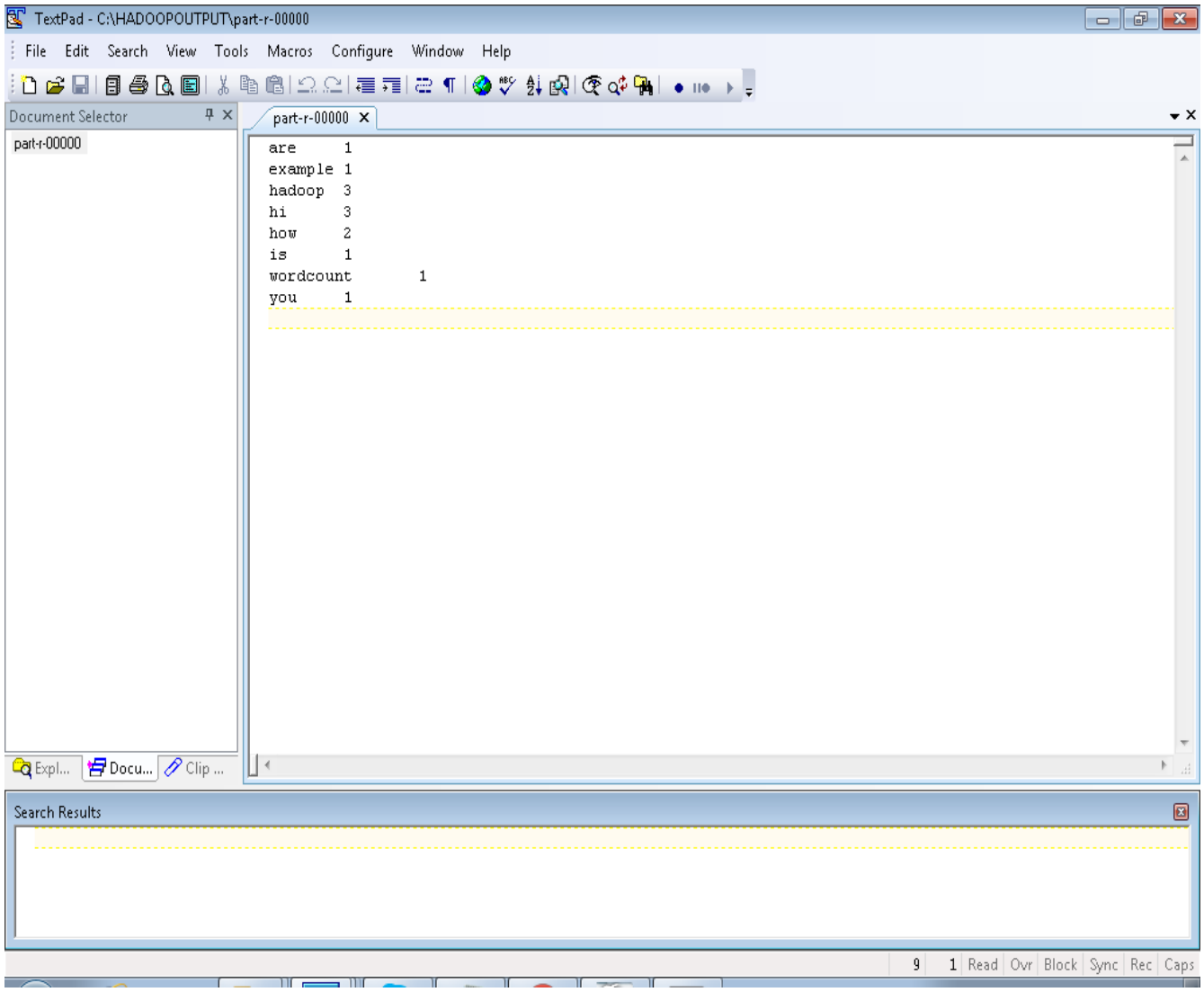
```
scala> counts.saveAsTextFile("sparkoutput")
```


« Local Disk (C:) ▸ HADOOPOUTPUT ▸ sparkoutput
Search sparkoutput
Organize
Include in library
Share with
Burn
New folder
Desktop
Downloads
Prefetch
Recent Places
VirtualBox VMs
datascience - Shortcut
Libraries
Documents
Music
Pictures
Videos
Homegroup
Computer
Local Disk (C:)
books (D:)
workspace (G:)
Name
Date modified
Type
Size
._SUCCESS.crc 3/9/2016 10:11 PM CRC File 1 KB
.part-00000.crc 3/9/2016 10:11 PM CRC File 1 KB
.part-00001.crc 3/9/2016 10:11 PM CRC File 1 KB
_SUCCESS 3/9/2016 10:11 PM File 0 KB
part-00000 3/9/2016 10:11 PM File 1 KB
part-00001 3/9/2016 10:11 PM File 1 KB
6 items
at org.apache.spark.deploy.SparkSubmit$.doRunMain$1(SparkSubmit.scala:181)
at org.apache.spark.deploy.SparkSubmit$.submit(SparkSubmit.scala:206)
at org.apache.spark.deploy.SparkSubmit$.main(SparkSubmit.scala:121)
at org.apache.spark.deploy.SparkSubmit.main(SparkSubmit.scala)
scala> counts.saveAsTextFile("sparkoutput")
[Stage 0:> (0 + 0) / 2]
[Stage 0:> (0 + 2) / 2]
scala>
Find
6.93 x 4.80
100 %
10:11 PM
3/9/2016


TextPad - C:\HADOOPOUTPUT\sparkoutput\part-00000
File Edit Search View Tools Macros Configure Window Help
Explorer
ktop
Libraries
Homegroup
Arun
Computer
Local Disk (C:)
Anaconda2
AnypointStudio
apache-maven-3.3.9
arun
cygwin64
hadoop-2.2.0
HADOOPOUTPUT
input
metastore_db
sparkoutput
._SUCCESS.crc
.part-00000.crc
.part-00001.crc
_SUCCESS
part-00000
part-00001
streamoutput
_SUCCESS
derby
mapreduce
part-00000
(example,1)
(are,1)
(is,1)
(how,2)
Search Results
File: part-00000, 35 characters, 4 lines, UNIX, 1252 (ANSI - Latin I)
1 1 Read Ovr Block Sync Rec Caps

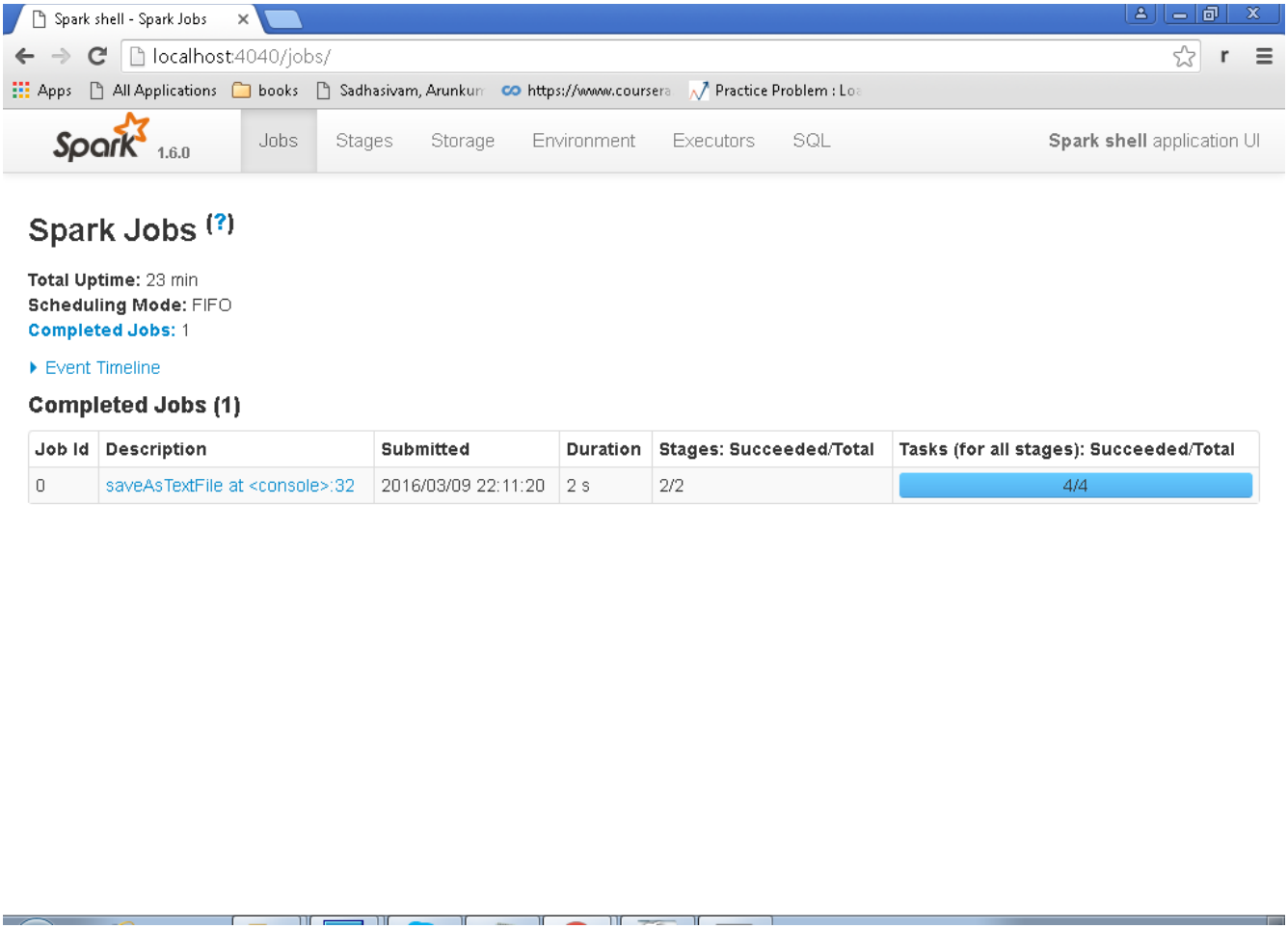

TextPad - C:\HADOOPOUTPUT\sparkoutput\part-00001
File Edit Search View Tools Macros Configure Window Help
Explorer
ktop
Libraries
Homegroup
Arun
Computer
Local Disk (C:)
Anaconda2
AnypointStudio
apache-maven-3.3.9
arun
cygwin64
hadoop-2.2.0
HADOOPOUTPUT
input
metastore_db
sparkoutput
._SUCCESS.crc
.part-00000.crc
.part-00001.crc
_SUCCESS
part-00000
part-00001
streamoutput
_SUCCESS
derby
mapreduce
part-00001
part-00000
(you,1)
(wordcount,1)
(hadoop,3)
(hi,3)
Search Results
5 1 Read Ovr Block Sync Rec Caps

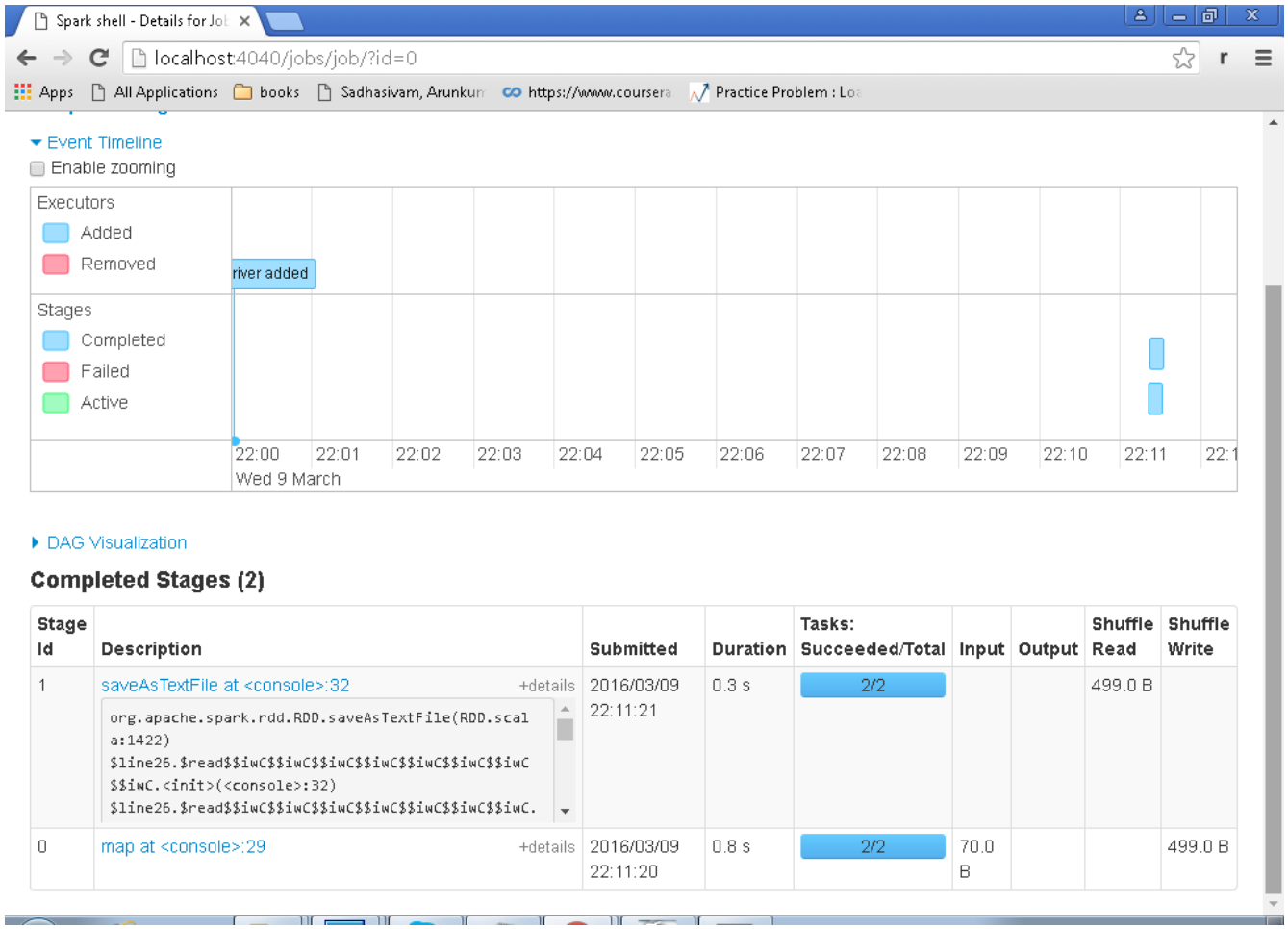
HADOOPOUTPUT



View output in web:

http://localhost:4040/jobs/





NOTE:

Above all program run inside spark-shell command. But to work in yarn command is

spark-shell --master yarn-client. For this hadoop is needed

TO INTEGRATE SPARK WITH HADOOP

To integrate spark with hadoop just need to add HADOOP_CONFIG_DIR or HADOOP_CONF_DIR environment variable in the system path.

If environment variable is not set then it works as spark standalone container

if environment variable is set it works on top on hadoop. To retrieve file inside hadoop , need to start hadoop.Although hadoop need not to be started for starting yarn-client since running

`spark-shell --master yarn-client` command or running `spark-shell` command both is same if `HADOOP_CONFIG_DIR` or `YARN_CONFIG_DIR` env variable set.

Launching Spark on YARN

Ensure that `HADOOP_CONF_DIR` or `YARN_CONF_DIR` points to the directory which contains the (client side) configuration files for the Hadoop cluster. These configs are used to write to HDFS and connect to the YARN ResourceManager. The configuration contained in this directory will be distributed to the YARN cluster so that all containers used by the application use the same configuration. If the configuration references Java system properties or environment variables not managed by YARN, they should also be set in the Spark application's configuration (driver, executors, and the AM when running in client mode).

There are two deploy modes that can be used to launch Spark applications on YARN. In `cluster` mode, the Spark driver runs inside an application master process which is managed by YARN on the cluster, and the client can go away after initiating the application. In `client` mode, the driver runs in the client process, and the application master is only used for requesting resources from YARN.

Unlike Spark standalone and Mesos modes, in which the master's address is specified in the `--master` parameter, in YARN mode the ResourceManager's address is picked up from the Hadoop configuration. Thus, the `--master` parameter is `yarn`.

To launch a Spark application in `cluster` mode:

```
$ ./bin/spark-submit --class path.to.your.Class --master yarn --deploy-mode cluster
[options] <app jar> [app options]
```

```
C:\HADOOPOUTPUT>spark-submit --class org.apache.spark.examples.SparkPi \
    --master yarn-cluster \
    --num-executors 3 \
    --driver-memory 4g \
    --executor-memory 2g \
    --executor-cores 1 \
    lib/spark-examples*.jar \
    10
```

The above starts a YARN client program which starts the default Application Master. Then SparkPi will be run as a child thread of Application Master. The client will periodically poll the Application Master for status updates and display them in the console. The client will exit once your application has finished running. Refer to the "Viewing Logs" section below for how to see driver and executor logs.

To launch a Spark application in yarn-client mode, do the same, but replace "yarn-cluster" with "yarn-client". To run spark-shell:

```
$ ./bin/spark-shell --master yarn-client
```

Above all program run inside spark-shell command. But to work in yarn command is spark-shell --master yarn-client. For this hadoop is needed

commands:

- **spark-shell:**
  It provides standalne spark scala environment cant interact with hdfs yarn

C:\HADOOPOUTPUT>spark-shell

- **spark-shell --master yarn-client:**
  It run spark on top of hdfs.

2.1 RUN SPARK ON HADOOP

since above all program run without hadoop

NOTE:

Need YARN_CONF_DIR or HADOOP_CONF_DIR need to set on System environment variables otherwise below error will occur.

C:\HADOOPOUTPUT>spark-shell --master yarn-client

```
Exception in thread "main" java.lang.Exception: When running with master 'yarn-c
lient' either HADOOP_CONF_DIR or YARN_CONF_DIR must be set in the environment.
        at org.apache.spark.deploy.SparkSubmitArguments.validateSubmitArguments(
SparkSubmitArguments.scala:251)
        at org.apache.spark.deploy.SparkSubmitArguments.validateArguments(SparkS
ubmitArguments.scala:228)
        at org.apache.spark.deploy.SparkSubmitArguments.<init>(SparkSubmitArgume
nts.scala:109)
        at org.apache.spark.deploy.SparkSubmit$.main(SparkSubmit.scala:114)
        at org.apache.spark.deploy.SparkSubmit.main(SparkSubmit.scala)
```

2.1.1 BEFORE CONFIGURE HADOOP_CONFIG_DIR(env variable):

To check where it is pointing ,wrong path given,it shows error see it is pointing towards local file system not hadoop 9000 service.

```
Administrator: Windows Command Processor - spark-shell

scala> val inputfile = sc.textFile("C://HADOOPOUTPUT//input.txt")
inputfile: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[5] at textFile at
 <console>:27

scala> val counts = inputfile.flatMap(line => line.split(" ")).map(word => (word
, 1)).reduceByKey(_+_);
org.apache.hadoop.mapred.InvalidInputException: Input path does not exist: file:
/C:/HADOOPOUTPUT/input.txt
        at org.apache.hadoop.mapred.FileInputFormat.listStatus(FileInputFormat.j
ava:251)
        at org.apache.hadoop.mapred.FileInputFormat.getSplits(FileInputFormat.ja
va:270)
        at org.apache.spark.rdd.HadoopRDD.getPartitions(HadoopRDD.scala:199)
        at org.apache.spark.rdd.RDD$$anonfun$partitions$2.apply(RDD.scala:239)
        at org.apache.spark.rdd.RDD$$anonfun$partitions$2.apply(RDD.scala:237)
        at scala.Option.getOrElse(Option.scala:120)
        at org.apache.spark.rdd.RDD.partitions(RDD.scala:237)
        at org.apache.spark.rdd.MapPartitionsRDD.getPartitions(MapPartitionsRDD.
scala:35)
        at org.apache.spark.rdd.RDD$$anonfun$partitions$2.apply(RDD.scala:239)
        at org.apache.spark.rdd.RDD$$anonfun$partitions$2.apply(RDD.scala:237)
        at scala.Option.getOrElse(Option.scala:120)
        at org.apache.spark.rdd.RDD.partitions(RDD.scala:237)
```

After giving correct local system path:

it works fine if correct path is given

```
Administrator: Windows Command Processor - spark-shell
, 1)).reduceByKey(_+_);
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[22] at reduceByKey
 at <console>:29

scala> val inputfile = sc.textFile("C://HADOOPOUTPUT//wordcount.txt")
inputfile: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[24] at textFile a
t <console>:27

scala> val counts = inputfile.flatMap(line => line.split(" ")).map(word => (word
, 1)).reduceByKey(_+_);
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[27] at reduceByKey
 at <console>:29

scala> counts.toDebugString
res1: String =
(2) ShuffledRDD[27] at reduceByKey at <console>:29 []
 +-(2) MapPartitionsRDD[26] at map at <console>:29 []
    |  MapPartitionsRDD[25] at flatMap at <console>:29 []
    |  MapPartitionsRDD[24] at textFile at <console>:27 []
    |  C://HADOOPOUTPUT//wordcount.txt HadoopRDD[23] at textFile at <console>:27
 []

scala>

scala>
```

NOTE:

All three steps works fine before I configure HADOOP_CONFIG_DIR or YARN_CONFIG_DIR

```
scala> val inputfile = sc.textFile("c://HADOOPOUTPUT/wordcount.txt")

scala> val counts = inputfile.flatMap(line => line.split(" ")).map(word => (word,
1)).reduceByKey(_+_);

scala> counts.toDebugString
```

OUTPUT:

```
(2) ShuffledRDD[27] at reduceByKey at <console>:29 []
 +-(2) MapPartitionsRDD[26] at map at <console>:29 []
    |  MapPartitionsRDD[25] at flatMap at <console>:29 []
    |  MapPartitionsRDD[24] at textFile at <console>:27 []
    |  C://HADOOPOUTPUT//wordcount.txt HadoopRDD[23] at textFile at <console>:27
 []

scala>
```

AFTER CONFIGURE HADOOP_CONFIG_DIR ENV VARIABLE:

see it looks in hadoop 9000 port since spark yarn-client is started or hadoop_conf_dir set

```
Administrator: Windows Command Processor - spark-shell
16/03/11 01:25:07 INFO storage.BlockManagerInfo: Added broadcast_1_piece0 in mem
ory on localhost:52552 (size: 15.5 KB, free: 511.5 MB)
16/03/11 01:25:07 INFO spark.SparkContext: Created broadcast 1 from textFile at
<console>:21
inputfile: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[3] at textFile at
 <console>:21

scala> val counts = inputfile.flatMap(line => line.split(" ").map(word => (word
, 1)).reduceByKey(_+_);
java.lang.IllegalArgumentException: Pathname /c:/HADOOPOUTPUT/input/wordcount.tx
t from hdfs://localhost:9000/c:/HADOOPOUTPUT/input/wordcount.txt is not a valid
DFS filename.
        at org.apache.hadoop.hdfs.DistributedFileSystem.getPathName(DistributedF
ileSystem.java:190)
        at org.apache.hadoop.hdfs.DistributedFileSystem.access$000(DistributedFi
leSystem.java:98)
        at org.apache.hadoop.hdfs.DistributedFileSystem$17.doCall(DistributedFil
eSystem.java:1112)
        at org.apache.hadoop.hdfs.DistributedFileSystem$17.doCall(DistributedFil
eSystem.java:1108)
        at org.apache.hadoop.fs.FileSystemLinkResolver.resolve(FileSystemLinkRes
olver.java:81)
        at org.apache.hadoop.hdfs.DistributedFileSystem.getFileStatus(Distribute
dFileSystem.java:1108)
        at org.apache.hadoop.fs.Globber.getFileStatus(Globber.java:57)
```

I entered spark-shell since I started spark yarn-client it looks in yarn (hadoop dir)

Above shows hadoop valid input files

now It works because since hadoop /input/wordcount.txt which is path of hadoop.it works only if hadoop is started.

RUN SPARK WITH HADOOP STARTED

```
Administrator: Windows Command Processor

C:\HADOOPOUTPUT>hdfs dfs -ls /input/
Found 8 items
-rw-r--r--   1 Arun supergroup     123637 2016-02-24 02:11 /input/sales.csv
-rw-r--r--   1 Arun supergroup    1398907 2016-02-25 00:09 /input/sales10000.csv

-rw-r--r--   1 Arun supergroup     466379 2016-02-24 22:53 /input/sales3500.csv
-rw-r--r--   1 Arun supergroup    8594762 2016-02-25 00:22 /input/sales65536.csv

-rw-r--r--   1 Arun supergroup     129745 2016-03-03 01:29 /input/salesunique.cs
v
-rw-r--r--   1 Arun supergroup     179820 2016-03-03 01:57 /input/salesunique350
0.csv
-rw-r--r--   1 Arun supergroup    1476056 2016-03-03 01:47 /input/salesunique655
36.csv
-rw-r--r--   1 Arun supergroup         70 2016-02-24 02:11 /input/wordcount.txt

C:\HADOOPOUTPUT>_
```

See it is valid hdfs input path.

```
Select Administrator: Windows Command Processor - spark-shell
ory on localhost:52552 (size: 15.5 KB, free: 511.4 MB)
16/03/11 01:29:48 INFO spark.SparkContext: Created broadcast 4 from textFile at
<console>:21
inputfile: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[15] at textFile a
t <console>:21

scala> val inputfile = sc.textFile("/input/wordcount.txt")
16/03/11 01:32:35 INFO storage.MemoryStore: Block broadcast_5 stored as values i
n memory (estimated size 162.2 KB, free 959.1 KB)
16/03/11 01:32:35 INFO storage.MemoryStore: Block broadcast_5_piece0 stored as b
ytes in memory (estimated size 15.5 KB, free 974.6 KB)
16/03/11 01:32:35 INFO storage.BlockManagerInfo: Added broadcast_5_piece0 in mem
ory on localhost:52552 (size: 15.5 KB, free: 511.4 MB)
16/03/11 01:32:35 INFO spark.SparkContext: Created broadcast 5 from textFile at
<console>:21
inputfile: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[17] at textFile a
t <console>:21

scala> val counts = inputfile.flatMap(line => line.split(" ").map(word => (word
, 1)).reduceByKey(_+_);
16/03/11 01:32:39 INFO mapred.FileInputFormat: Total input paths to process : 1
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[20] at reduceByKey
 at <console>:23

scala>
```

All the command works fine now.

See Also RDD array no keep incremented each time command is entered

Also you can see it create a folder in /tmp

you can check the Spark files created in temporary directory (C:\Users\Arun\AppData\Local\Temp)

when you manually delete these temp spark folders and try you can see

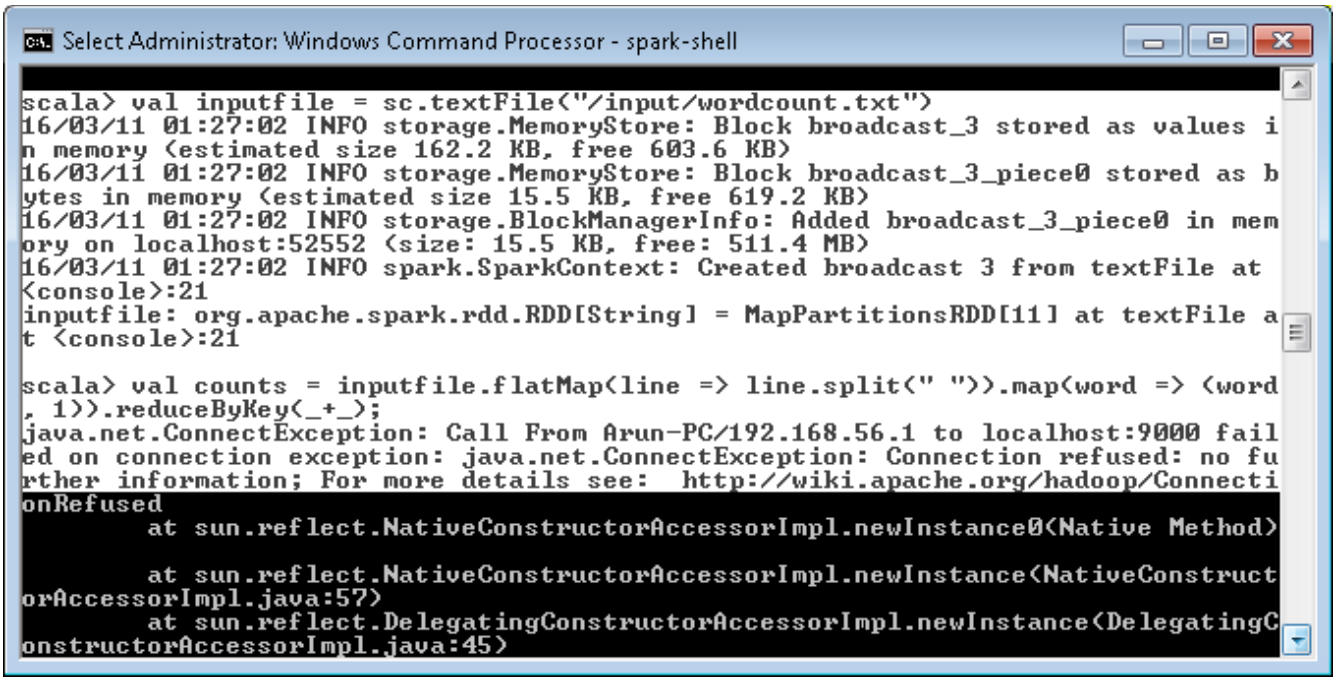
scala> val inputfile = sc.textFile("/input/wordcount.txt")

inputfile: org.apache.spark.rdd.RDD[String] = MapPartitionsRDD[1] at textFile at <console>:27

RDD index arr starts re-index from 1 again

RUN SPARK AFTER STOPPING HADOOP

I tried running the spark-shell without starting hadoop and run the same command above it shows error.



STARTING SPARK-SHELLWITH ENVIRONMENT VARIABLE:

if YARN_CONFIG_DIR or HADOOP_CONFIG_DIR env variable is set ,output it starts the web console in 1localhost:4040

NOTE:

if HADOOP CONFIG_DIR or YARN_CONFIG_DIR env variable is set then it starts using spark driver service else it starts using datanucleus api

1) if YARN_CONFIG_DIR or HADOOP_CONFIG_DIR is not set then **spark-shell --master yarn-client** wont start it shows error asking to set "HADOOP_CONFIG_DIR or YARN_CONFIG_DIR".

2)if **spark-shell** is started without hadoop config env variable, then it looks in only local dir.

3)if hadoop conf env variable is set then it points to hdfs (YARN).

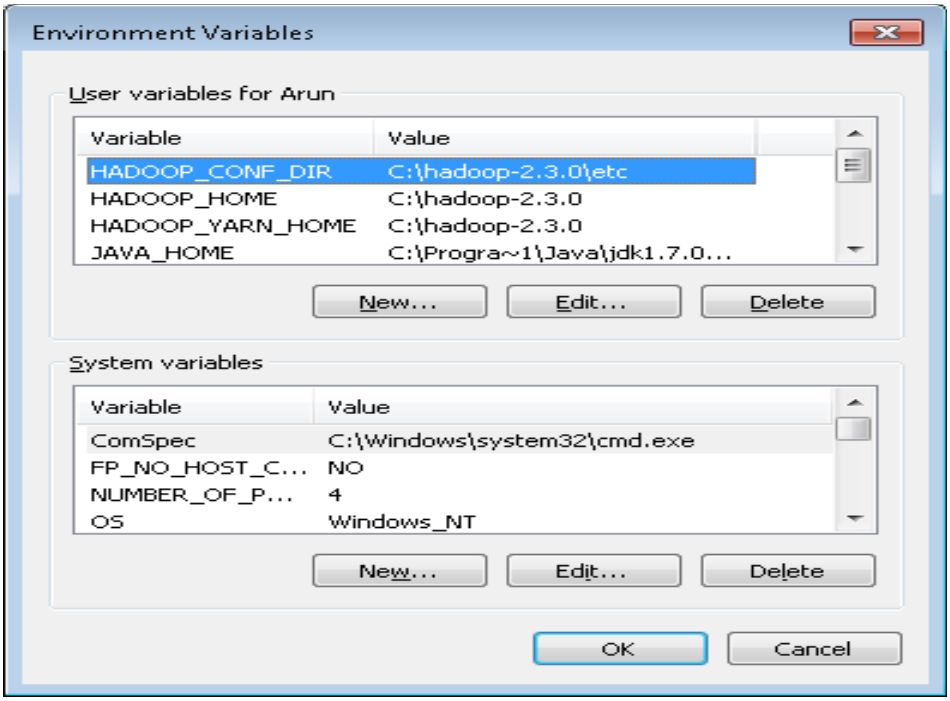
if env variable configured:

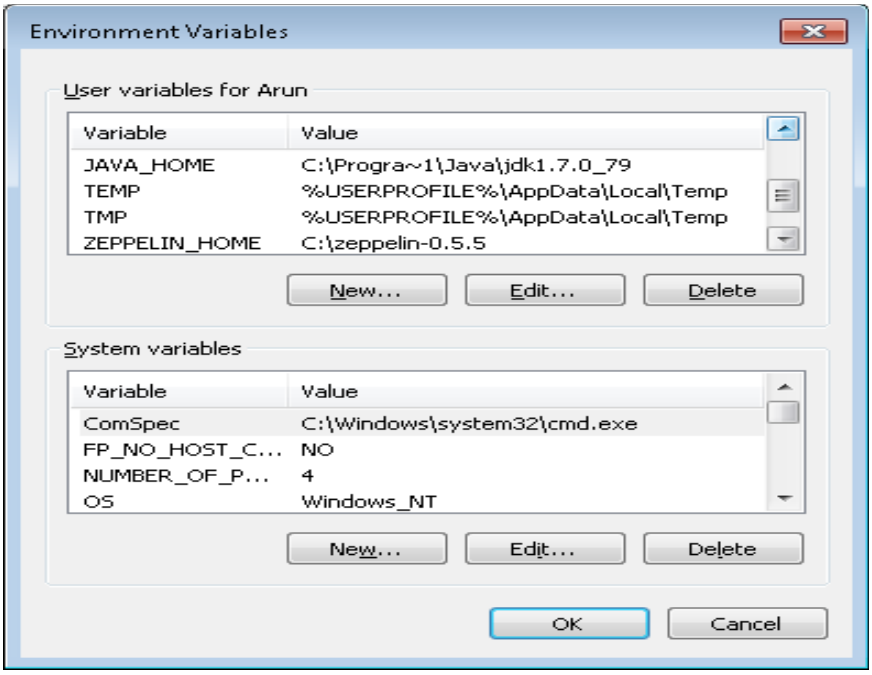
it starts withsparkdriver service

```
Administrator: Windows Command Processor - spark-shell --master yarn-client
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>spark-shell --master yarn-client
16/03/10 00:50:33 INFO spark.SecurityManager: Changing view acls to: Arun
16/03/10 00:50:33 INFO spark.SecurityManager: Changing modify acls to: Arun
16/03/10 00:50:33 INFO spark.SecurityManager: SecurityManager: authentication di
sabled; ui acls disabled; users with view permissions: Set(Arun); users with mod
ify permissions: Set(Arun)
16/03/10 00:50:33 INFO spark.HttpServer: Starting HTTP Server
16/03/10 00:50:33 INFO server.Server: jetty-8.y.z-SNAPSHOT
16/03/10 00:50:33 INFO server.AbstractConnector: Started SocketConnector@0.0.0.0
:63715
16/03/10 00:50:33 INFO util.Utils: Successfully started service 'HTTP class serv
er' on port 63715.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.6.0
      /_/

Using Scala version 2.10.5 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_79)
Type in expressions to have them evaluated.
Type :help for more information.
16/03/10 00:50:39 INFO spark.SparkContext: Running Spark version 1.6.0
16/03/10 00:50:39 INFO spark.SecurityManager: Changing view acls to: Arun
16/03/10 00:50:39 INFO spark.SecurityManager: Changing modify acls to: Arun
16/03/10 00:50:39 INFO spark.SecurityManager: SecurityManager: authentication di
sabled; ui acls disabled; users with view permissions: Set(Arun); users with mod
ify permissions: Set(Arun)
16/03/10 00:50:39 INFO util.Utils: Successfully started service 'sparkDriver' on
 port 63732.
16/03/10 00:50:40 INFO slf4j.Slf4jLogger: Slf4jLogger started
16/03/10 00:50:40 INFO Remoting: Starting remoting
16/03/10 00:50:40 INFO Remoting: Remoting started; listening on addresses :[akka
.tcp://sparkDriverActorSystem@localhost:63745]
16/03/10 00:50:40 INFO util.Utils: Successfully started service 'sparkDriverActo
rSystem' on port 63745.
16/03/10 00:50:40 INFO spark.SparkEnv: Registering MapOutputTracker
16/03/10 00:50:40 INFO spark.SparkEnv: Registering BlockManagerMaster
16/03/10 00:50:40 INFO storage.DiskBlockManager: Created local directory at C:\U
sers\Arun\AppData\Local\Temp\blockmgr-4a26d4da-2c75-4f35-981a-9afe6c4b89b2
16/03/10 00:50:40 INFO storage.MemoryStore: MemoryStore started with capacity 51
1.5 MB
16/03/10 00:50:41 INFO spark.SparkEnv: Registering OutputCommitCoordinator
16/03/10 00:50:41 INFO server.Server: jetty-8.y.z-SNAPSHOT
16/03/10 00:50:41 INFO server.AbstractConnector: Started SelectChannelConnector@
0.0.0.0:4040
16/03/10 00:50:41 INFO util.Utils: Successfully started service 'SparkUI' on por
t 4040.
16/03/10 00:50:41 INFO ui.SparkUI: Started SparkUI at http://localhost:4040
16/03/10 00:50:41 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0
:8032
16/03/10 00:50:44 INFO ipc.Client: Retrying connect to server: 0.0.0.0/0.0.0.0:8
032. Already tried 0 time(s); retry policy is RetryUpToMaximumCountWithFixedSlee
p(maxRetries=10, sleepTime=1000 MILLISECONDS)
```

SPARKR

set environment and spark variables





PATH:

```
%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\;C:\Program
Files\Intel\WiFi\bin\;C:\Program Files\Common
Files\Intel\WirelessCommon\;C:\Program Files
(x86)\Skype\Phone\;C:\apache-maven-3.3.9\bin;;C:\Program
Files\Git\bin;C:\Progra~1\Java\jdk1.7.0_79\bin;C:\Program
Files\R\R-3.2.3\bin;C:\scala-2.11.8\bin;C:\SBT-
0.13\bin;C:\protoc;C:\cygwin64\bin;C:\hadoop-
2.3.0\bin;C:\hadoop-2.3.0\sbin;C:\spark-1.6.1-bin-
hadoop2.3\bin;C:\spark-1.6.1-bin-hadoop2.3\sbin;C:\zeppelin-
0.5.5\bin;C:\pig-0.15.0\bin
```

NOTE:

if using java under c:\program files use c:\progra~1 and if use under C:\Program Files (x86) use c:\program~2\ to avoid issues because of space between program files -issues.

To make spark use data bricks csv make sure you donwload spark based on hadoop version. I.e if hadoop -2.3 download pre-build version of spark for hadoop 2.3 and aswell as download equivalent databricks jar and put in class path below

Different other approaches:

Approach 1:

read.csv(file="data/train.csv", header=TRUE)

system hangs and exist

Approach 2:

train<-read.csv.ffdf(file="data/train.csv", header=TRUE, VERBOSE=TRUE, first.rows=10000, next.rows=50000,colClasses=NA)

it takes so much time more than 2 hrs and finally show error.

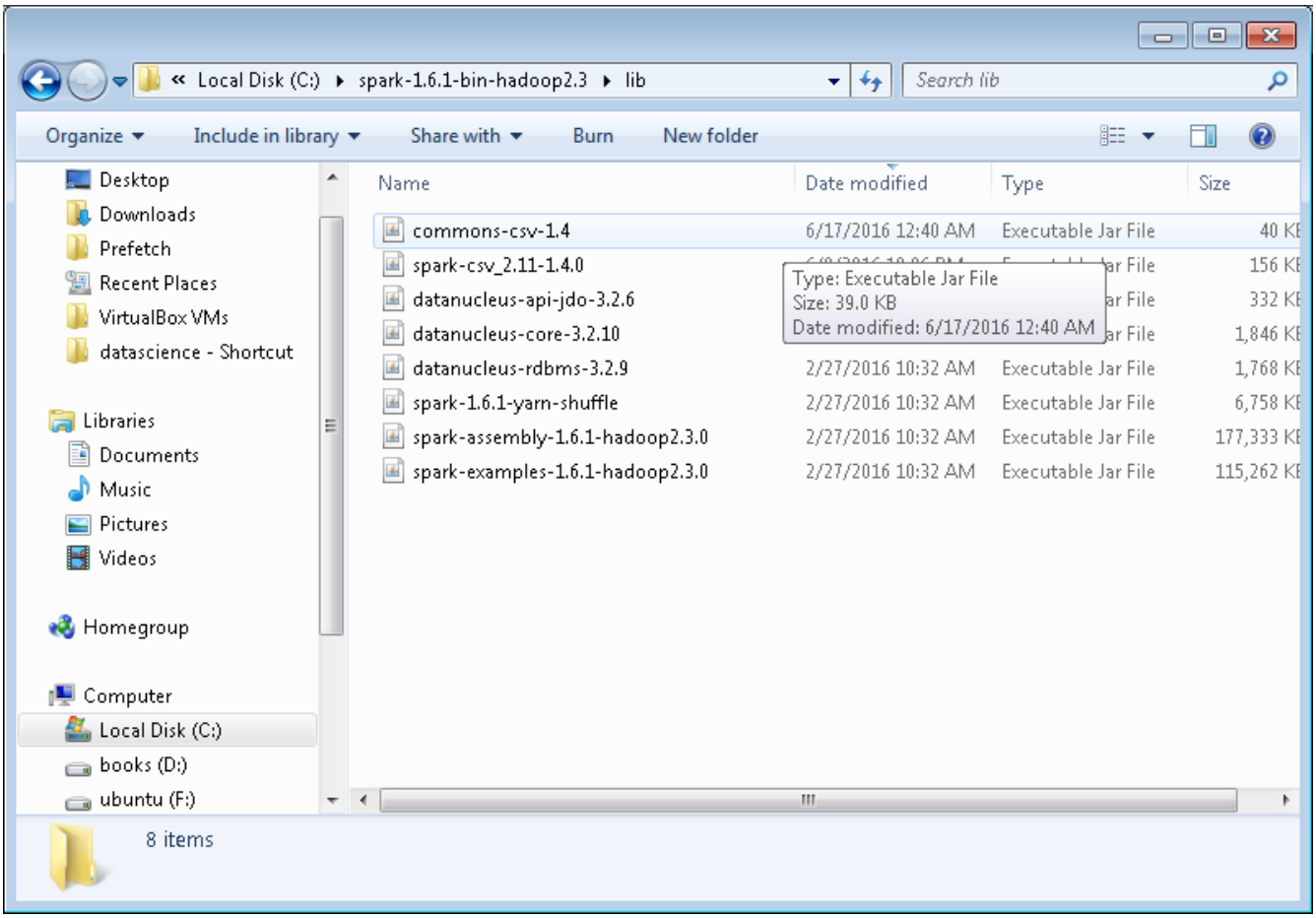
it creates a tmp file under C:\Users\Arun\AppData\Local\Temp for each load and aggregate to **dataframe.**

Sample log:

```
read.table.ffdf 8310001..8360000 (50000)  csv-read=4.63sec ffdf-write=7.91sec
read.table.ffdf 8360001..8410000 (50000)  csv-read=4.62sec ffdf-write=6.88sec
read.table.ffdf 8410001..8460000 (50000)  csv-read=4.56sec ffdf-write=6.55sec
read.table.ffdf 8460001..8510000 (50000)  csv-read=4.54sec ffdf-write=7.22sec
read.table.ffdf 8510001..8560000 (50000)  csv-read=4.68sec ffdf-write=10.42sec
read.table.ffdf 8560001..8610000 (50000)  csv-read=4.14sec ffdf-write=7.55sec
```

Approach3:

sparkR -loads faster 181-295 s on avg(3 to 5 min)



As you can see spark-csv jar is added to make use of databrick api

spark-csv_2.11-1.4.0.jar is needed to make it work

Need to give the version in the below lines to initialize spark data brick

```
sc <- sparkR.init(master = "local[*]", appName = "SparkR",
                  sparkPackages ="com.databricks:spark-csv_2.10:1.4.0")
```

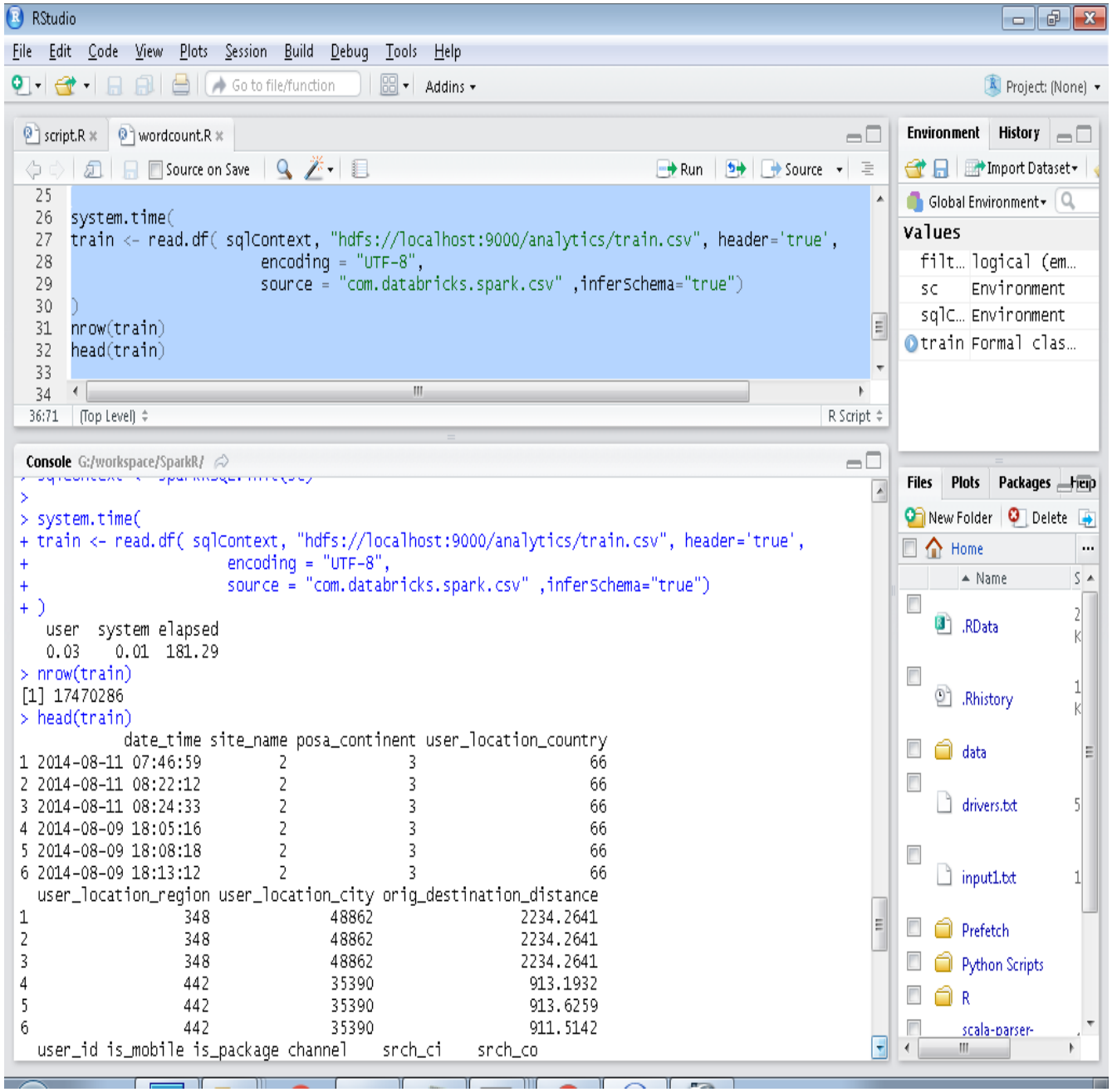
once running the above line in R program you can check which version of data frame using it shows below log when starting

```
Launching java with spark-submit command spark-submit.cmd  --packages
com.databricks:spark-csv_2.10:1.4.0 sparkr-shell
C:\Users\Arun\AppData\Local\Temp\RtmpiujqL2\backend_port17943ffb5d0f
```

```
# Sys.setenv(SPARK_HOME = "C:\\spark-1.6.1-bin-hadoop2.3")
# .libPaths(c(file.path(Sys.getenv('SPARK_HOME'), 'R', 'lib'), .libPaths()))
#
# Sys.setenv('SPARKR_SUBMIT_ARGS='"--packages" "com.databricks:spark-csv_2.10:1.4.0" "sparkr-shell"')
#
# spark_env = list('spark.executor.memory' = '4g',
#                  'spark.executor.instances' = '4',
#                  'spark.executor.cores' = '4',
#                  'spark.driver.memory' = '4g')
#

library(SparkR)
# sc <- sparkR.init(master = "local[*]", appName = "SparkR", sparkEnvir = spark_env,
#                   sparkPackages ="com.databricks:spark-csv_2.10:1.4.0",
#                   sparkJars=c("C:\\spark-1.6.1-bin-hadoop2.3\\lib\\spark-csv_2.10-1.4.0.jar",
#                               "C:\\spark-1.6.1-bin-hadoop2.3\\lib\\commons-csv-1.2.jar"))


sc <- sparkR.init(master = "local[*]", appName = "SparkR",
                  sparkPackages ="com.databricks:spark-csv_2.10:1.4.0")




sqlContext <- sparkRSQL.init(sc)


train <- read.df( sqlContext, "hdfs://localhost:9000/analytics/train.csv", header='true',
                      encoding = "UTF-8",
                      source = "com.databricks.spark.csv" ,inferSchema="true")


nrow(train)
head(train)


#sparkR.stop()
```

NOTE:

uncomment or comment no effect



Total Records:

174,70,286 -seventeen million four hundred seventy thousand two hundred eighty-six

Time taken:

```
user  system elapsed
  0.03    0.01  181.29
```

To Know the spark param to use in spark.init()

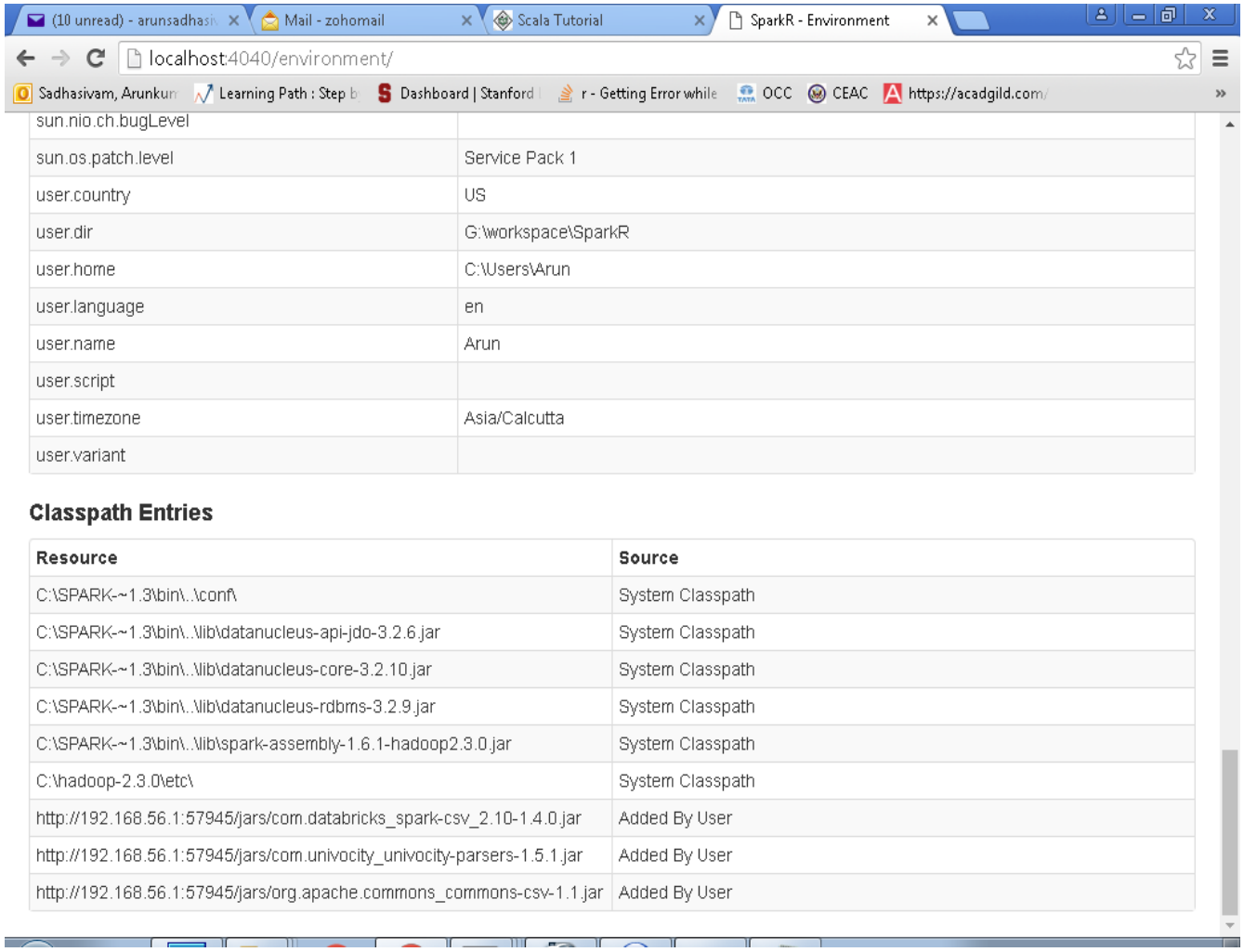
```
STEP 1:
======
start spark using
>sparkR
check the console it shows
conosle output:
================
16/06/19 19:01:21 INFO SparkUI: Started SparkUI at http://192.168.1.2:4040

STEP 2:
======
check appname, appid,master from the console
http://localhost:4040/environment/

Spark Properties
==================

spark.app.id    local-1466343081802
spark.app.name  SparkR
spark.driver.host       192.168.1.2
spark.driver.port       2863
spark.executor.id       driver
spark.executorEnv.LD_LIBRARY_PATH       $LD_LIBRARY_PATH:
spark.externalBlockStore.folderName     spark-e82d221d-7ba2-45ec-abc7-
9639ed6d1806
spark.home      C:\spark-1.6.1-bin-hadoop2.3
spark.master    local[*]
spark.jars      file:/C:/Users/Arun/.ivy2/jars/com.databricks_spark-csv_2.10-
                1.4.0.jar,file:/C:/Users/Arun/.ivy2/jars/org.apache.commons_commons-csv-
                1.1.jar,file:/C:/Users/Arun/.ivy2/jars/com.univocity_univocity-parsers-1.5.1.jar
spark.scheduler.mode    FIFO
spark.submit.deployMode client
```

The above driver host,master and post will be use below:

```
sc <- sparkR.init(master = "local[*]", appName = "SparkR",
                  sparkPackages ="com.databricks:spark-csv_2.10:1.4.0")
```

It uses jars from C:/Users/Arun/.ivy2/jars/ **to run R program with initializing spark.init() and try to delete the jars ,you can't delete unless you give sparkR.stop()**

sparkR.stop()



Once extra are added check whether it is referenced correctly it should show like below Added by user.

http://192.168.56.1:57945/jars/com.databricks_spark-csv_2.10-1.4.0.jar	Added By User
http://192.168.56.1:57945/jars/com.univocity_univocity-parsers-1.5.1.jar	Added By User
http://192.168.56.1:57945/jars/org.apache.commons_commons-csv-1.1.jar	Added By User

If the expected jar is not in the above **Added by user list** then try add the below lines to add to class path.

```
Sys.setenv(SPARK_HOME = "C:\\spark-1.6.1-bin-hadoop2.3")
 .libPaths(c(sc <- sparkR.init(master = "local[*]", appName = "SparkR",
              sparkPackages ="com.databricks:spark-csv_2.10:1.4.0",
              sparkJars=c("C:\\spark-1.6.1-bin-hadoop2.3\\lib\\spark-csv_2.10-1.4.0.jar",
                     "C:\\spark-1.6.1-bin-hadoop2.3\\lib\\commons-csv-
1.2.jar"))file.path(Sys.getenv('SPARK_HOME'), 'R', 'lib'), .libPaths()))
```

```
Error in invokeJava(isStatic = TRUE, className, methodName, ...) :
  No connection to backend found. Please re-run sparkR.init
>
```

NOTE:

Make sure init() parameters is given correctly as in spark envv by checking spark env site.http://localhost:4040/environment/ **other wise above error No connection to backend will be thrown.**

SparkR - Environment

localhost:4040/environment/

sun.os.patch.level	Service Pack 1
user.country	US
user.dir	G:\workspace\SparkR
user.home	C:\Users\Arun
user.language	en
user.name	Arun
user.script	
user.timezone	Asia/Calcutta
user.variant	

Classpath Entries

Resource	Source
C:\SPARK-~1.3\bin\..\conf\	System Classpath
C:\SPARK-~1.3\bin\..\lib\datanucleus-api-jdo-3.2.6.jar	System Classpath
C:\SPARK-~1.3\bin\..\lib\datanucleus-core-3.2.10.jar	System Classpath
C:\SPARK-~1.3\bin\..\lib\datanucleus-rdbms-3.2.9.jar	System Classpath
C:\SPARK-~1.3\bin\..\lib\spark-assembly-1.6.1-hadoop2.3.0.jar	System Classpath
C:\hadoop-2.3.0\etc\	System Classpath
http://192.168.56.1:59807/jars/avro-1.7.7.jar	Added By User
http://192.168.56.1:59807/jars/com.databricks_spark-csv_2.10-1.4.0.jar	Added By User
http://192.168.56.1:59807/jars/com.univocity_univocity-parsers-1.5.1.jar	Added By User
http://192.168.56.1:59807/jars/org.apache.commons_commons-csv-1.1.jar	Added By User

See now Avro added to classpath.

APACHE HIVE

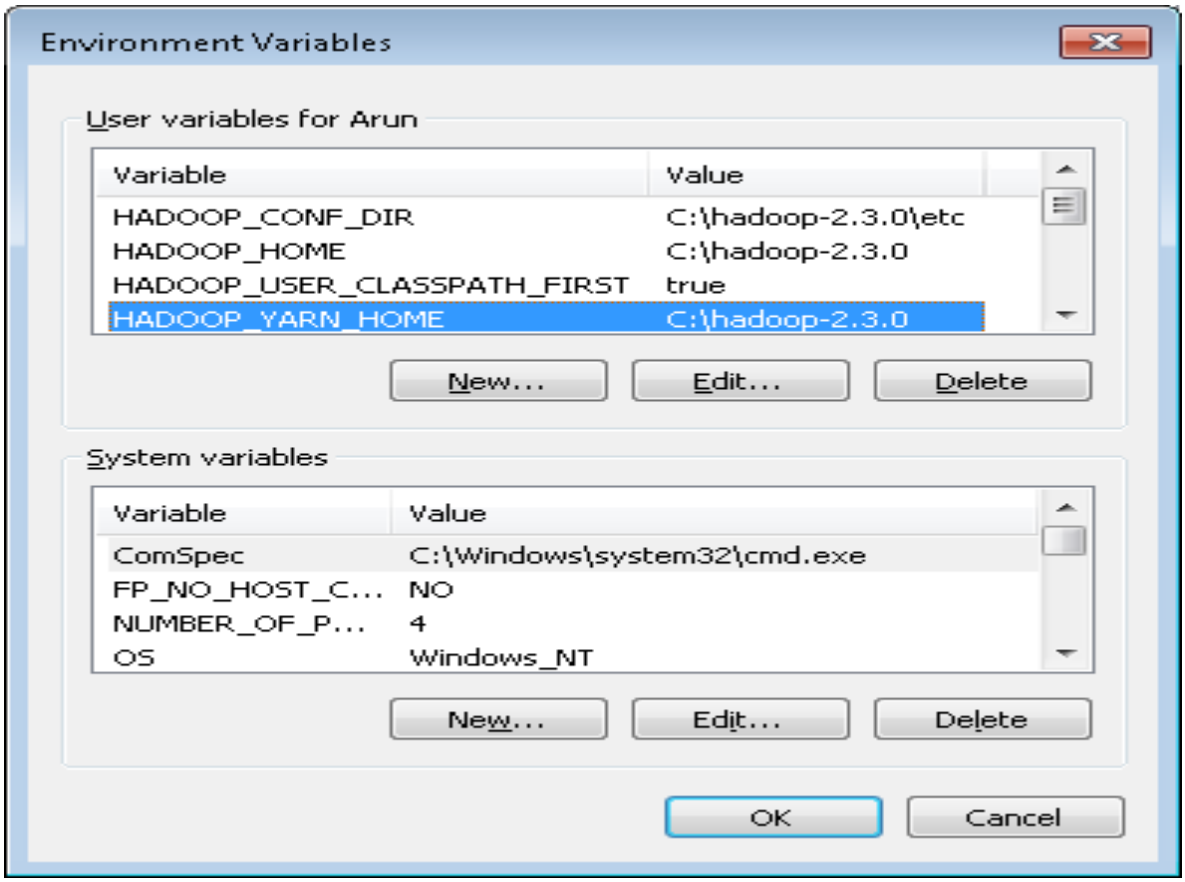
Download hive add hive to path and also since hive need some db

I have added mongodb to PATH variable on top of exising values

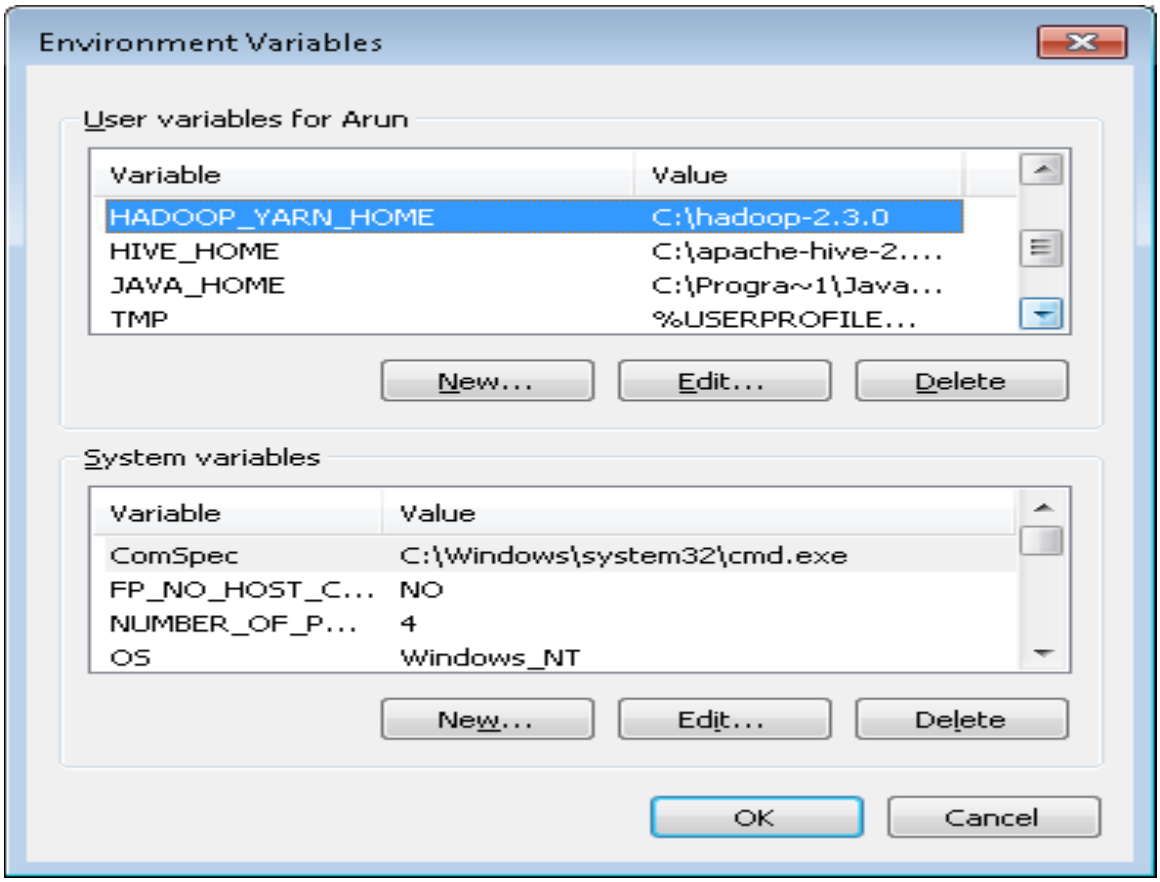
%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\;C:\Program Files\Intel\WiFi\bin\;C:\Program Files\Common Files\Intel\WirelessCommon\;C:\Program Files (x86)\Skype\Phone\;C:\apache-maven-3.3.9\bin;;C:\Program Files\Git\bin;C:\Progra~1\Java\jdk1.7.0_79\bin;C:\Program Files\R\R-3.2.3\bin;C:\scala-2.11.8\bin;C:\SBT-0.13\bin;C:\protoc;C:\cygwin64\bin;C:\hadoop-2.3.0\bin;C:\hadoop-2.3.0\sbin;C:\spark-1.6.1-bin-hadoop2.3\bin;C:\spark-1.6.1-bin-hadoop2.3\sbin;C:\zeppelin-0.5.5\bin;C:\pig-0.15.0\bin;**C:\apache-hive-2.1.0-bin\bin;C:\Program Files\MongoDB\Server\3.2\bin**

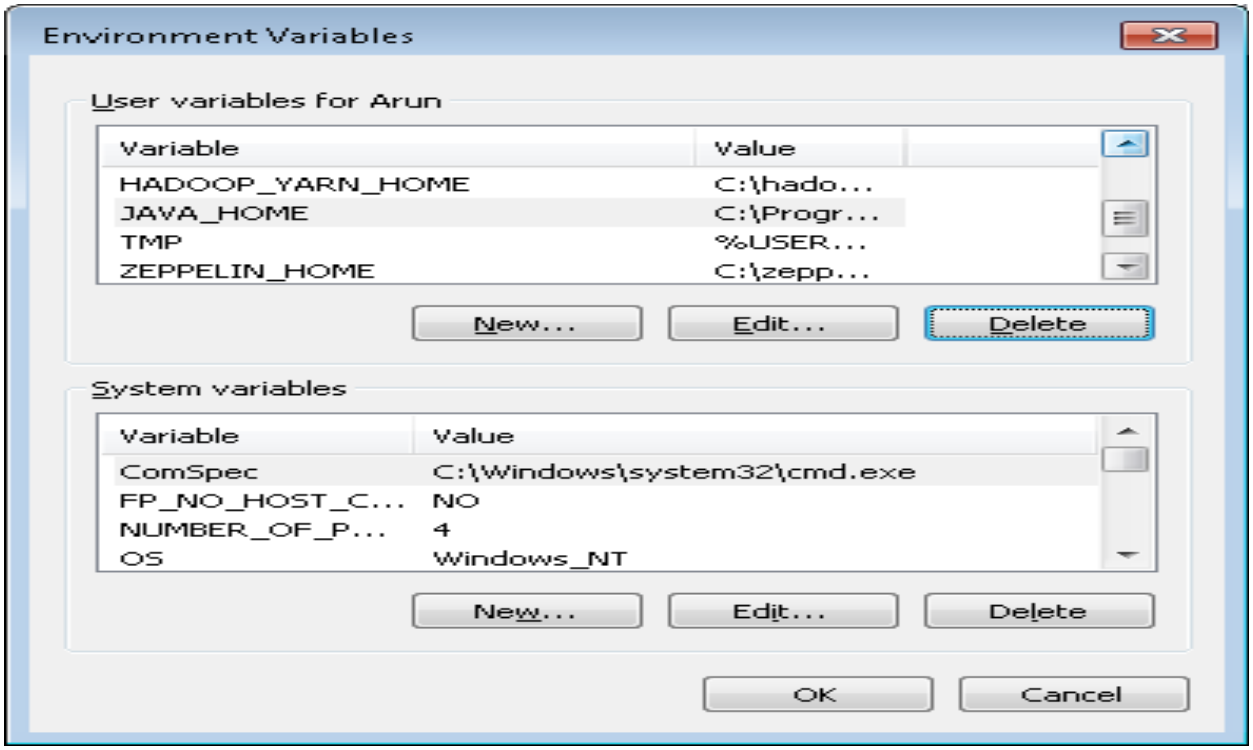
Dont change anything I tried mongo but hive does not support mongo directly. It only works with derby or mysql.

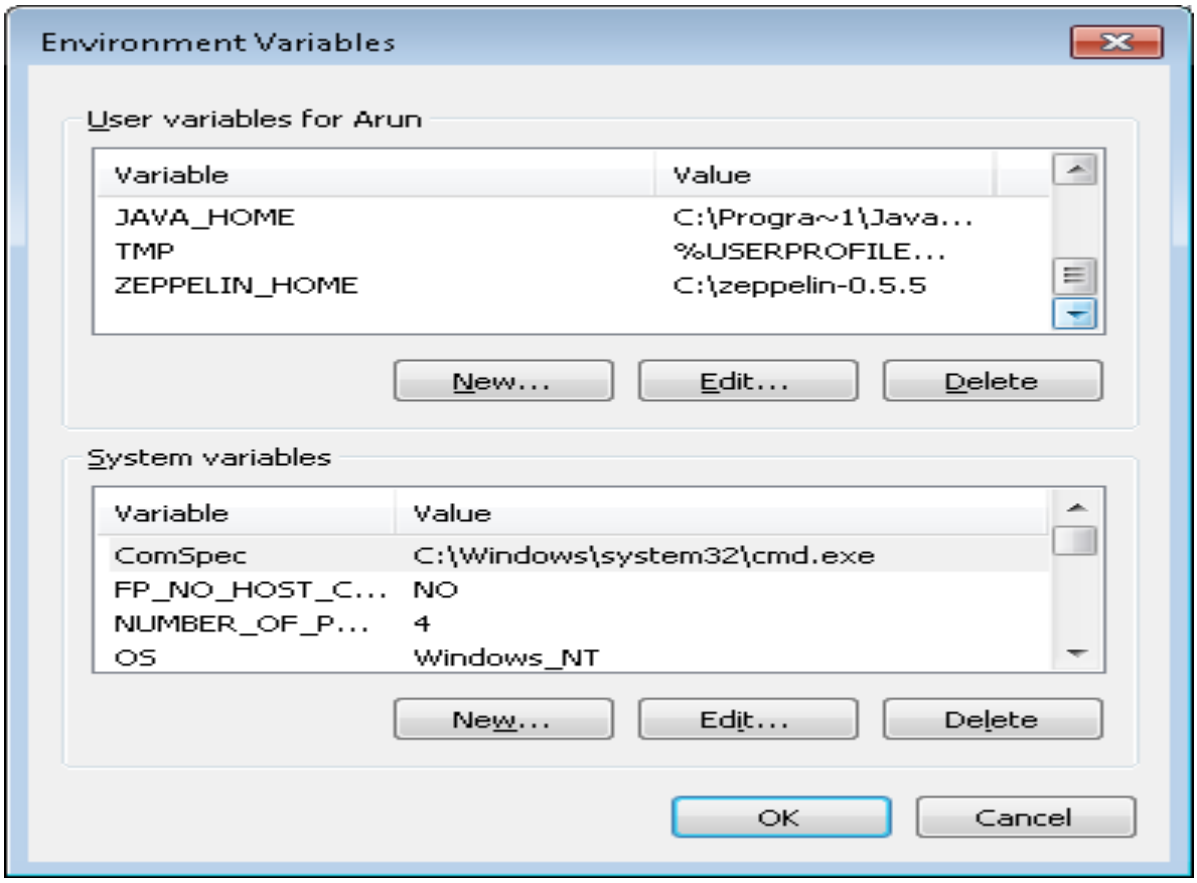
Rename C:\apache-hive-2.1.0-bin\conf\hive-default.xml.template to hive-site.xml



Add HADOOP_USER_CLASSPATH_FIRST variable




Environment Variables
User variables for Arun
Variable
Value
HADOOP_YARN_HOME
C:\hado...
JAVA_HOME
C:\Progr...
TMP
%USER...
ZEPPELIN_HOME
C:\zepp...
New...
Edit...
Delete
System variables
Variable
Value
ComSpec
C:\Windows\system32\cmd.exe
FP_NO_HOST_C...
NO
NUMBER_OF_P...
4
OS
Windows_NT
New...
Edit...
Delete
OK
Cancel


Environment Variables
User variables for Arun
Variable
Value
JAVA_HOME
C:\Progra~1\Java...
TMP
%USERPROFILE...
ZEPPELIN_HOME
C:\zeppelin-0.5.5
New...
Edit...
Delete
System variables
Variable
Value
ComSpec
C:\Windows\system32\cmd.exe
FP_NO_HOST_C...
NO
NUMBER_OF_P...
4
OS
Windows_NT
New...
Edit...
Delete
OK
Cancel

ERROR: `IncompatibleClassChangeError:`

if `HADOOP_USER_CLASSPATH_FIRST` is not set below error will be thrown.

```
java.lang.IncompatibleClassChangeError: Found class jline.Terminal, but interface was expected
         at jline.TerminalFactory.create(TerminalFactory.java:101)
         at jline.TerminalFactory.get(TerminalFactory.java:158)
         at jline.console.ConsoleReader.<init>(ConsoleReader.java:229)
```

if below error occurs

```
Error applying authorization policy on hive configuration: Couldn't create direc
tory ${system:java.io.tmpdir}\${hive.session.id}_resources
Connection is already closed.
```

Only change to do in hive installation is change hive-default.xml.template to hive-site.xml

After Change hive-default.xml.template in C:\apache-hive-2.1.0-bin\conf to normal file ,do changes below.

```
<property>
  <name>hive.exec.local.scratchdir</name>
    <value>${HIVE_HOME}/iotmp</value>
    <description>Local scratch space for Hive jobs</description>
</property>



 <property>
   <name>hive.downloaded.resources.dir</name>
    <value>${HIVE_HOME}/iotmp</value>
    <description>Temporary local directory for added resources in the remote file
system.</description>
 </property>
```

You could try adding both the "mongo-hadoop-hive.jar" and "mongo-hadoop-core.jar" to the hive.aux.jars.path setting in your hive-site.xml.

STEPS:

1)just rename C:\apache-hive-2.1.0-bin\conf\hive-default.xml.template to hive-site.xml .

2)Change all ${system:java.io.tmpdir}/${system:user.name} to some valid path like c://hive_resources

3)if needed add jar to C:\apache-hive-2.1.0-bin\lib directory.

`Hive ADD JAR` C:\apache-hive-2.1.0-bin\lib\mongo-hadoop-hive-1.5.2.jar

`hive ADD JAR` C:\apache-hive-2.1.0-bin\lib\mongo-hadoop-core-1.5.2.jar.

`Hive ADD JAR` C:\apache-hive-2.1.0-bin\lib\mongodb-driver-3.2.2`.jar`

Derby

```
C:\db-derby-10.12.1.1-bin\bin>ij
ij version 10.12
ij> connect 'jdbc:derby:hl7;create=true';
ij> connect 'jdbc:derby:analytics;create=true';
ij(CONNECTION1)>
```

```
C:\Users\Arun>hive
ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
Connecting to jdbc:hive2://
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]
Connected to: Apache Hive (version 2.1.0)
Driver: Hive JDBC (version 2.1.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 2.1.0 by Apache Hive
hive>
```

This shows hive is started successfully!!!

hive commands

NOTE:

if you give (;) at the end of the statement it gets executed else it takes to next line only after (;) it executes.

```
Microsoft Windows [Version 6.1.7601]

Copyright (c) 2009 Microsoft Corporation.  All rights reserved.


C:\Users\Arun>hive

ERROR StatusLogger No log4j2 configuration file found. Using default configurati

on: logging only errors to the console.

Connecting to jdbc:hive2://

SLF4J: Class path contains multiple SLF4J bindings.

SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl

-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j

-log4j12-1.7.5.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]

Connected to: Apache Hive (version 2.1.0)

Driver: Hive JDBC (version 2.1.0)

Transaction isolation: TRANSACTION_REPEATABLE_READ

Beeline version 2.1.0 by Apache Hive

hive> CREATE DATABASE  hl7

. . > ;

OK

No rows affected (4.593 seconds)
```

```
hive> CREATE SCHEMA hl7details;

OK

No rows affected (0.204 seconds)

hive> SHOW DATABASES;

OK

default

hl7

hl7details

medical

userdb

5 rows selected (1.217 seconds)

hive>
```

if trying to run hive without starting hadoop below error will be thrown:

Hive depends on Hadoop

```
Administrator: Windows Command Processor
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>hive
ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
Connecting to jdbc:hive2://
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]
06:57:22.377 [main] ERROR org.apache.hadoop.hive.metastore.ObjectStore - Version
 information found in metastore differs 1.2.0 from expected schema version 2.1.0
. Schema verififcation is disabled hive.metastore.schema.verification so setting
 version.
Error applying authorization policy on hive configuration: java.net.ConnectExcep
tion: Call From Arun-PC/192.168.0.105 to localhost:9000 failed on connection exc
eption: java.net.ConnectException: Connection refused: no further information; F
or more details see:  http://wiki.apache.org/hadoop/ConnectionRefused
Beeline version 2.1.0 by Apache Hive
Error applying authorization policy on hive configuration: java.net.ConnectExcep
tion: Call From Arun-PC/192.168.0.105 to localhost:9000 failed on connection exc
eption: java.net.ConnectException: Connection refused: no further information; F
or more details see:  http://wiki.apache.org/hadoop/ConnectionRefused
Connection is already closed.

C:\Windows\System32>_
```

ERROR :java.lang.VerifyError: class

java.lang.VerifyError: class org.apache.hadoop.hdfs.protocol.proto.ClientNamenodeProtocolProtos$AppendRequestProto overrides final method getUnknownFields.()Lcom/google/protobuf/UnknownFieldSet;

INSTALLATION STEPS:

To start hive type hive in command prompt.

```
C:\hive_warehouse>hive
ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
Connecting to jdbc:hive2://
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]
```

Run in Admin mode.

use point to any dir like C:\hive_warehouse otherwise it creates block and other thing c:\windows\system 32.

when run from c:\windows\system32
log created in c:\windows\system32\derby.log shows user.dir = c:\windows\system32

```
----------------------------------------------------------------
Sat Jul 16 15:39:49 IST 2016:
Booting Derby version The Apache Software Foundation - Apache Derby - 10.10.2.0 -
(1582446): instance a816c00e-0155-f32e-f5bb-0000031ee388
on database directory C:\Windows\System32\metastore_db with class loader
sun.misc.Launcher$AppClassLoader@30a4effe
Loaded from file:/C:/apache-hive-2.1.0-bin/lib/derby-10.10.2.0.jar
java.vendor=Oracle Corporation
java.runtime.version=1.7.0_80-b15
user.dir=C:\Windows\System32
os.name=Windows 7
os.arch=amd64
os.version=6.1
derby.system.home=null
Database Class Loader started - derby.database.classpath=''
```

when run from c:\hive_warehouse

```
Sat Jul 16 15:32:00 IST 2016:
Booting Derby version The Apache Software Foundation - Apache Derby - 10.10.2.0 -
(1582446): instance a816c00e-0155-f327-ce55-000003270550
on database directory C:\hive_warehouse\metastore_db with class loader
sun.misc.Launcher$AppClassLoader@30a4effe
Loaded from file:/C:/apache-hive-2.1.0-bin/lib/derby-10.10.2.0.jar
java.vendor=Oracle Corporation
java.runtime.version=1.7.0_80-b15
user.dir=C:\hive_warehouse
os.name=Windows 7
os.arch=amd64
os.version=6.1
derby.system.home=null
Database Class Loader started - derby.database.classpath=''
```

but when run from c:/hive_warehouse below error shows:
ERROR
======
Error applying authorization policy on hive configuration: org.apache.hadoop.hiv
e.ql.metadata.HiveException: org.apache.hadoop.hive.ql.metadata.HiveException: M
etaException(message:Hive metastore database is not initialized. Please use sche
matool (e.g. ./schematool -initSchema -dbType ...) to create the schema. If need
ed, don't forget to include the option to auto-create the underlying database in
 your JDBC connection string (e.g. ?createDatabaseIfNotExist=true for mysql))
Connection is already closed.

To make particular folder as root for hive give admin rights first

```
C:\hive_warehouse>TAKEOWN /A /R /F c:\hive_warehouse
SUCCESS: The file (or folder): "c:\hive_warehouse" now owned by the administrato
rs group.

SUCCESS: The file (or folder): "c:\hive_warehouse\allocator_mmap" now owned by t
he administrators group.
SUCCESS: The file (or folder): "c:\hive_warehouse\downloaded" now owned by the a
dministrators group.

SUCCESS: The file (or folder): "c:\hive_warehouse\local_scratchdir" now owned by
 the administrators group.

SUCCESS: The file (or folder): "c:\hive_warehouse\metastore_db" now owned by the
 administrators group.
SUCCESS: The file (or folder): "c:\hive_warehouse\derby.log" now owned by the ad
ministrators group.
```

above shows sucess

FOLDER SETUP

To delete any directory

Hdfs have 2 types of delete policy trash
1) skip trash - cannot recover like windows trash
2)if no skipTrash added deleted files saved in trash.
By default trash feature is disabled.

NOTE: give rm -r for both skipTrash and ordinary delete otherwise 'hl7_details':is a directory error will be thrown.

```
C:\Windows\system32>hdfs dfs -rm -r /hl7_details
C:\Windows\system32>hdfs dfs -rm -r -skipTrash /hl7_details
16/07/16 14:55:05 INFO fs.TrashPolicyDefault: Namenode trash configuration: Dele
tion interval = 0 minutes, Emptier interval = 0 minutes.
Deleted /hl7_details
```

STEP 1:

set below environment variables
HIVE_HOME -C:\apache-hive-2.1.0-bin
HADOOP_USER_CLASSPATH_FIRST - TRUE
to make sure that hadoop components loads first

STEP 2:

only changes needed is below 4 things:

default values before changing:

```
 <property>
    <name>hive.exec.scratchdir</name>
    <value>/tmp/hive</value>
    <description>HDFS root scratch dir for Hive jobs which gets created with write
all (733) permission. For each connecting user, an HDFS scratch dir: $
{hive.exec.scratchdir}/&lt;username&gt; is created, with $
{hive.scratch.dir.permission}.</description>
  </property>

  <property>
    <name>hive.exec.local.scratchdir</name>
    <value>${system:java.io.tmpdir}/${system:user.name}</value>
    <description>Local scratch space for Hive jobs</description>
  </property>

  <property >
    <name>hive.downloaded.resources.dir</name>
    <value>${system:java.io.tmpdir}/${hive.session.id}_resources</value>
    <description>Temporary local directory for added resources in the remote file
system.</description>
  </property>
```

autocreate

```
  <property>
    <name>datanucleus.schema.autoCreateAll</name>
    <value>false</value>
    <description>creates necessary schema on a startup if one doesn't exist. set
this to false, after creating it once</description>
  </property>
```

After changing 4 values:

```
 <property>
    <name>hive.exec.scratchdir</name>
    <value>\hive</value>
    <description>HDFS root scratch dir for Hive jobs which gets created with write
all (733) permission. For each connecting user, an HDFS scratch dir: $
{hive.exec.scratchdir}/&lt;username&gt; is created, with $
{hive.scratch.dir.permission}.</description>
  </property>
  <property>
    <name>hive.exec.local.scratchdir</name>
```

```
    <value>C:\hive_warehouse\scratchdir\</value>
    <description>Local scratch space for Hive jobs</description>
  </property>



  <property>
    <name>hive.downloaded.resources.dir</name>
    <value>C:\hive_warehouse\downloaded\</value>
    <description>Temporary local directory for added resources in the remote file
system.</description>
  </property>



  <property>
    <name>datanucleus.schema.autoCreateAll</name>
    <value>true</value>
    <description>creates necessary schema on a startup if one doesn't exist. set
this to false, after creating it once</description>
  </property>
```

Mainly datanucleus.schema.autoCreateAll needed if mounting on different directory I.e if mount on default admin dir c:/windows/system32 it works fine , but mount on different dir as hive warehouse like c:/hive_warehouse needs to set it as true.

```
e.g
===
  if given any path like c:/hl7 it shows error like below.
Error: Error while processing statement: FAILED: Execution Error, return code 1
from org.apache.hadoop.hive.ql.exec.DDLTask. MetaException(message:java.lang.Ill
egalArgumentException: Pathname /c:/hl7/hl71.db from hdfs://localhost:9000/c:/hl
7/hl71.db is not a valid DFS filename.) (state=08S01,code=1).It should be like
/hive or /hl7 as given in hive-site.xml.
```

STEP 3:

```
check whether hive started sucessfully.if started sucessfully it should prompt with
hive> prompt as like below

C:\hive_warehouse>hive
ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
Connecting to jdbc:hive2://
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]
Connected to: Apache Hive (version 2.1.0)
Driver: Hive JDBC (version 2.1.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 2.1.0 by Apache Hive
hive>
```

STEP 4:

check whether able to create database,schema,table.

```
hive> show databases
. . > ;
OK
default
1 row selected (2.24 seconds)
hive> CREATE DATABASE hl7;
OK
No rows affected (0.29 seconds)
hive> CREATE SCHEMA hl7details;
OK
No rows affected (0.12 seconds)
hive> SHOW DATABASES;
OK
default
hl7
hl7details
3 rows selected (0.104 seconds)
hive>
hive> CREATE TABLE employee ( eid int, name String,
. . > salary String, destination String)
. . > COMMENT "Employee details"
. . > ;
OK
No rows affected (1.187 seconds)
hive>
```

STEP 5

check name given in scratch dir reflects in the hdfs

```
<property>
    <name>hive.exec.scratchdir</name>
    <value>\hive</value>
  </property>
```

```
C:\Windows\system32>hdfs dfs -ls /hive/*
Found 2 items
drwx------   - admin supergroup          0 2016-07-16 21:59 /hive/admin/7de88bd0
-98a4-4afe-b973-a154d70f0f56
drwx------   - admin supergroup          0 2016-07-16 21:59 /hive/admin/7e0b4141
-c2ff-4a61-a0a5-671fbd5bca23
```

STEP 6

create a table in hdfs and load in to hive and confirm

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.
```

```
C:\Windows\system32>hdfs dfs -put C:\HADOOPOUTPUT\hive.txt /hive/employee1.txt
C:\Windows\system32>hdfs dfs -ls /hive/*
Found 2 items
drwx------   - admin supergroup          0 2016-07-16 21:59 /hive/admin/7de88bd0
-98a4-4afe-b973-a154d70f0f56
drwx------   - admin supergroup          0 2016-07-16 21:59 /hive/admin/7e0b4141
```

```
-c2ff-4a61-a0a5-671fbd5bca23
Found 1 items
-rw-r--r--   1 admin supergroup        206 2016-07-16 22:24 /hive/employee.txt
Found 1 items
-rw-r--r--   1 admin supergroup        206 2016-07-16 22:25 /hive/employee1.txt

C:\Windows\system32>
```

STEP 7:

check whether hive able to insert from local

```
hive> LOAD DATA LOCAL INPATH 'c:/HADOOPOUTPUT/hive.txt' OVERWRITE INTO TABLE emp
loyee;
Loading data to table default.employee
OK
No rows affected (1.287 seconds)
hive>
```

it should trigger hdfs

```
16/07/16 22:29:50 INFO hdfs.StateChange: DIR* completeFile: /user/hive/warehouse
/employee/hive.txt is closed by DFSClient_NONMAPREDUCE_-444183833_1
```

CHECK BY SELECTING TABLES:

```
hive> select eid,name from employee;
OK
.
.
.
5 rows selected (0.29 seconds)
hive>
hive>
```

STEP8:LOAD TABLES FROM HDFS

```
hive> load data inpath '/hive/employee.txt' into table employee;
Loading data to table default.employee
OK
No rows affected (0.924 seconds)
```

see when tried for 2 time you can see /hime/employee.txt is no more in hdfs since it is copied to hive warehouse table.so it throws error.

```
hive> load data inpath '/hive/employee.txt' into table employee;


FAILED: SemanticException Line 1:17 Invalid path ''/hive/employee.txt'': No file
s matching path hdfs://localhost:9000/hive/employee.txt
22:48:04.879 [9901c7c1-1e66-4395-a6fd-993ab58f09ac main] ERROR org.apache.hadoop
.hive.ql.Driver - FAILED: SemanticException Line 1:17 Invalid path ''/hive/emplo
yee.txt'': No files matching path hdfs://localhost:9000/hive/employee.txt
org.apache.hadoop.hive.ql.parse.SemanticException: Line 1:17 Invalid path ''/hiv
e/employee.txt'': No files matching path hdfs://localhost:9000/hive/employee.txt

        at org.apache.hadoop.hive.ql.parse.LoadSemanticAnalyzer.applyConstraints
```

check the hdfs

see the /hive/employee.txt exists previously copied to hive from hdfs.
see it is now removed in hdfs only employee1.txt remains.

```
C:\Windows\system32>hdfs dfs -ls /hive/*
Found 4 items
drwx------   - admin supergroup          0 2016-07-16 21:59 /hive/admin/7de88bd0
-98a4-4afe-b973-a154d70f0f56
drwx------   - admin supergroup          0 2016-07-16 22:36 /hive/admin/7e0b4141
-c2ff-4a61-a0a5-671fbd5bca23
drwx------   - admin supergroup          0 2016-07-16 22:37 /hive/admin/9901c7c1
-1e66-4395-a6fd-993ab58f09ac
drwx------   - admin supergroup          0 2016-07-16 22:37 /hive/admin/9d5841d1
-510b-481d-9d7d-6dd9a8ef8ff7
Found 1 items
-rw-r--r--   1 admin supergroup        206 2016-07-16 22:25 /hive/employee1.txt

C:\Windows\system32>
```

NOTES:

For loading from local file in windows to hive warehouse

1.For loading from HDFS to hive warehouse:

```
hive> LOAD DATA LOCAL INPATH 'c:/HADOOPOUTPUT/hive.txt' OVERWRITE INTO TABLE
employee;
```

2.For loading from hdfs to hive:

```
 hive> load data inpath '/hive/employee.txt' into table employee;
```

Access Hive data from R

library(SparkR)

library(sparkRHive)

sc <- sparkR.init(master = "local[*]", appName = "SparkR")

```
library(SparkR)

library(sparkRHive)

sc <- sparkR.init(master = "local[*]", appName = "SparkR")

hiveContext <- sparkRHive.init(sc)

sql(hiveContext, "CREATE TABLE  HL7_Details (key INT, value STRING)")

sql(hiveContext, "LOAD DATA LOCAL INPATH 'G:/hl7/uploads/sample.txt' INTO TABLE HL7_Details")

results <- sql(hiveContext, "FROM HL7_Details SELECT key, value")

head(results)

results <- sql(hiveContext, "FROM HL7 SELECT key, value")

head(results)
```

output:

```
 key value
1   NA    NA
2   NA    NA
3   NA    NA
4   NA    NA
5   NA    NA
6   NA    NA
```

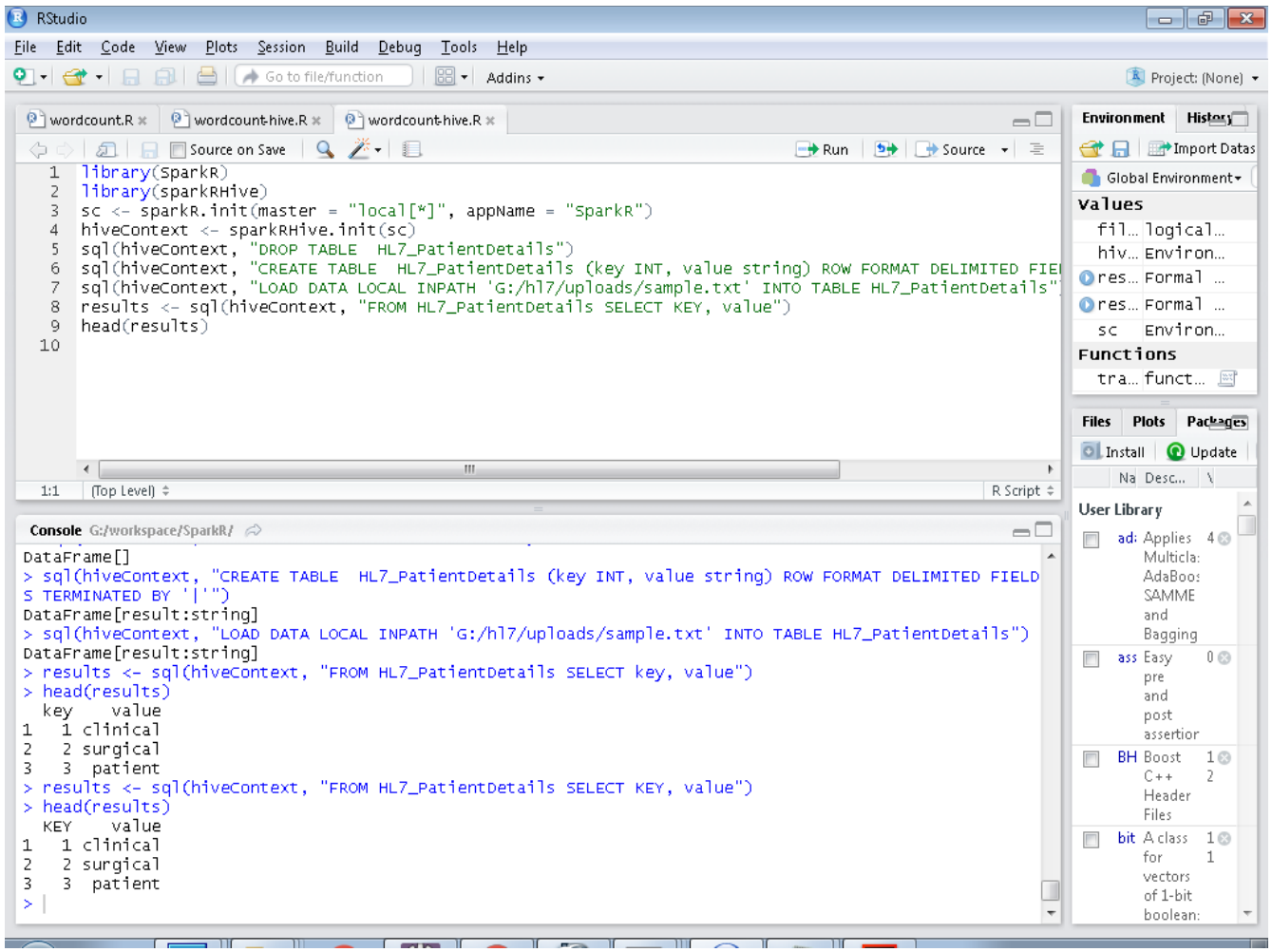
ERROR:

```
Error in above is that it create table command does not specify how to delimit
between 2 fields.
```

```
library(SparkR)
library(sparkRHive)
sc <- sparkR.init(master = "local[*]", appName = "SparkR")
hiveContext <- sparkRHive.init(sc)
sql(hiveContext, "DROP TABLE  HL7_PatientDetails")
sql(hiveContext, "CREATE TABLE  HL7_PatientDetails (key INT, value string) ROW
FORMAT DELIMITED FIELDS TERMINATED BY '|'")
sql(hiveContext, "LOAD DATA LOCAL INPATH 'G:/hl7/uploads/sample.txt' INTO TABLE
HL7_PatientDetails")
results <- sql(hiveContext, "FROM HL7_PatientDetails SELECT key, value")
head(results)
```

NOTE:
see highlighted row is very important , now it shows result.
Input file format:

1|clinical
2|surgical
3|patient



Thrift Hive Server

HiveServer is an optional service that allows a remote client to submit requests to Hive, using a variety of programming languages, and retrieve results. HiveServer is built on Apache Thrift™ (http://thrift.apache.org/), therefore it is sometimes called the Thrift server although this can lead to confusion because a newer service named HiveServer2 is also built on Thrift. Since the introduction of HiveServer2, HiveServer has also been called HiveServer1.

To start Hive server 2

To start Hive server 2
==========

command:
========
C:\Users\Arun>hive --service hiveserver2

ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]

Default port and Usage
==========
0.8 and Later
$ build/dist/bin/hive --service hiveserver –help
$ build/dist/bin/hive --service hiveserver2 –help # If hiveserver.cmd is unrecognized error thrown.
usage: hiveserver
 -h,--help Print help information
   --hiveconf <property=value> Use value for given property
   --maxWorkerThreads <arg> maximum number of worker threads,
                default:2147483647

```
  --minWorkerThreads <arg>      minimum number of worker threads,
                                default:100
 -p <port>                      Hive Server port number, default:10000
 -v,--verbose                   Verbose mode
```

$ bin/hive --service hiveserver

To view which port in use:
==========
C:\Users\Arun>netstat -a

Active Connections

Proto	Local Address	Foreign Address	State
TCP	0.0.0.0:22	Arun-PC:0	LISTENING
TCP	0.0.0.0:135	Arun-PC:0	LISTENING
TCP	0.0.0.0:445	Arun-PC:0	LISTENING
TCP	0.0.0.0:554	Arun-PC:0	LISTENING
TCP	0.0.0.0:2869	Arun-PC:0	LISTENING
TCP	0.0.0.0:8030	Arun-PC:0	LISTENING
TCP	0.0.0.0:8031	Arun-PC:0	LISTENING
TCP	0.0.0.0:8032	Arun-PC:0	LISTENING
TCP	0.0.0.0:8033	Arun-PC:0	LISTENING
TCP	0.0.0.0:8088	Arun-PC:0	LISTENING
TCP	0.0.0.0:10000	Arun-PC:0	LISTENING
TCP	0.0.0.0:10002	Arun-PC:0	LISTENING
TCP	0.0.0.0:10243	Arun-PC:0	LISTENING
TCP	0.0.0.0:49152	Arun-PC:0	LISTENING

Configuration Properties in the `hive-site.xml` File

hive.server2.thrift.min.worker.threads – Minimum number of worker threads, default 5.

hive.server2.thrift.max.worker.threads – Maximum number of worker threads, default 500.

hive.server2.thrift.port – TCP port number to listen on, default 10000.

hive.server2.thrift.bind.host – TCP interface to bind to.

Using the BeeLine CLI

BeeLine is a new CLI (command-line interface) for HiveServer2. It is based on the **SQLLine CLI** written by Marc Prud'hommeaux.

> You cannot use BeeLine to communicate with the original HiveServer (HiveServer1).

Use the following commands to start `beeline` and connect to a running HiveServer2 process. In this example the HiveServer2 process is running on `localhost` at port `10000`:

```
$ /usr/lib/hive/bin/beeline
beeline> !connect jdbc:hive2://localhost:10000 username password
org.apache.hive.jdbc.HiveDriver
0: jdbc:hive2://localhost:10000>
```

> **Note**:
>
> If you using HiveServer2 on a cluster that does *not* have Kerberos security enabled, then the password is arbitrary in the command for starting BeeLine.

Beeline – Command Line Shell

HiveServer2 supports a command shell Beeline that works with HiveServer2. It's a JDBC client that is based on the SQLLine CLI (http://sqlline.sourceforge.net/). There's detailed documentation of SQLLine which is applicable to Beeline as well.

Replacing the Implementation of Hive CLI Using Beeline

The Beeline shell works in both embedded mode as well as remote mode. In the embedded mode, it runs an embedded Hive (similar to Hive CLI) whereas remote mode is for connecting to a separate HiveServer2 process over Thrift. Starting in Hive 0.14, when Beeline is used with HiveServer2, it also prints the log messages from HiveServer2 for queries it executes to STDERR. Remote HiveServer2 mode is recommended for production use, as it is more secure and doesn't require direct HDFS/metastore access to be granted for users.
In remote mode HiveServer2 only accepts valid Thrift calls – even in HTTP mode, the message body contains Thrift payloads.

Beeline Example

```
% bin/beeline
Hive version 0.11.0-SNAPSHOT by Apache
beeline> !connect jdbc:hive2://localhost:10000 scott tiger
!connect jdbc:hive2://localhost:10000 scott tiger
```

```
Connecting to jdbc:hive2://localhost:10000
Connected to: Hive (version 0.10.0)
Driver: Hive (version 0.10.0-SNAPSHOT)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://localhost:10000> show tables;
show tables;
```

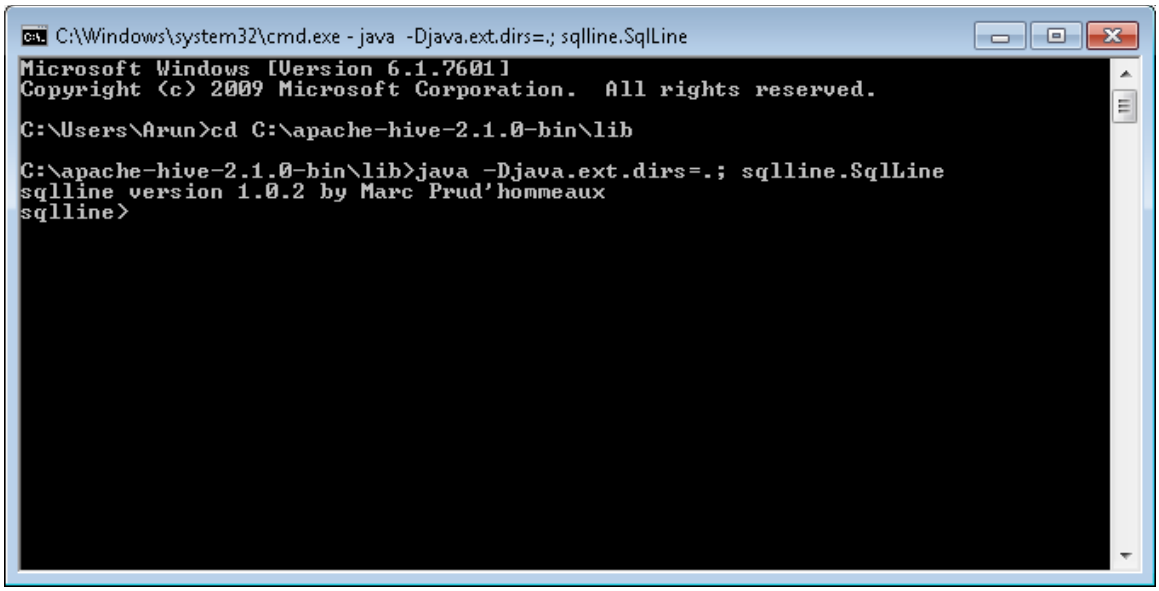
SQLLine:

Create a new directory/folder where you like. This will be referred to as sqllinedir.

1. Download sqlline.jar into sqllinedir.
2. Download the latest jline.jar from http://jline.sf.net into sqllinedir.
3. Download your database's JDBC driver files into sqllinedir. Note that some JDBC drivers require some installation, such as uncompressing or unzipping.

To confirm that HiveServer2 is working, start the beeline CLI and use it to execute a SHOW TABLES query on the HiveServer2 process:

downloaded sqlline and jline to hive path C:\apache-hive-2.1.0-bin\lib



Hive Installation Details

hive.exec.mode.local.auto:

When working with small data sets, using local mode execution will make Hive queries much faster. Setting the property set **hive.exec.mode.local.auto=true;** will cause Hive to use this mode more aggressively, even when you are running Hadoop in distributed or pseudodistributed mode.

Hive also has other components. A Thrift service provides remote access from other processes. Access using JDBC and ODBC are provided, too. They are implemented on op of the Thrift service.

All Hive installations require a metastore service, which Hive uses to store table schemas and other metadata.ressively, even when you are running Hadoop in distributed or pseudodistributed

Hive uses a built-in Derby SQL server, which provides limited, singleprocess storage. For example, when using Derby, you can't run two simultaneous instances of the Hive CLI. Setting the property set.

If you are running with the default Derby database for the metastore, you'll notice that your current working directory now contains a new subdirectory called metastore_db that was created by Derby during the short hive session you just executed. If you are running one of the VMs, it's possible it has configured different behavior.

Creating a **metastore_db subdirectory** under whatever working directory you happen to be in is not convenient, as Derby forgets" about previous metastores when you change to a new working directory! In the next section, we'll see how to configure a permanent location for the metastore database, as well as make other changes.

hive.metastore.warehouse.dir:

It indicate, the hive.metastore.warehouse.dir tells Hive where in your local filesystem to keep the data contents for Hive's tables.

hive.metastore.local :property defaults to true, so we don't really need to show

This property controls whether to connect to a remote metastore server or open a new metastore server as part *of the Hive Client JVM.*

Example 2-1. Local-mode hive-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
<property>
<name>hive.metastore.warehouse.dir</name>
<value>/home/me/hive/warehouse</value>
<description>
Local or HDFS directory where Hive keeps table contents.
</description>
</property>

<property>
<name>hive.metastore.local</name>
Configuring Your Hadoop Environment | 25
<value>true</value>
<description>
Use false if a production metastore server is used.
</description>

</property>
<property>
<name>javax.jdo.option.ConnectionURL</name>
<value>jdbc:derby:;databaseName=/home/me/hive/ metastore_db;create=true</value>
<description>
The JDBC connection URL.
</description>
</property>
</configuration>
```

in above xml <description> tags indicate, the hive.metastore.warehouse.dir tells Hive where in your local filesystem to keep the data contents for Hive's tables. (This value is appended to the value of fs.default.name defined in the Hadoop configuration and defaults to *file:///*.) You can use any directory path you want for the value. Note that this directory will not be used to store the table metadata, which goes in the separate *metastore*.

The hive.metastore.local property defaults to true, so we don't really need to show `. It's there more for documentation purposes. This property controls whether to connect to a remote metastore server or open a new metastore server as part

of the Hive Client JVM. This setting is almost always set to true and JDBC is used to communicate directly to a relational database. When it is set to false, Hive will communicate through a **metastore Methods below**

The value for the javax.jdo.option.ConnectionURL property makes one small but convenient change to the default value for this property. This property tells Hive how to connect to the *metastore* server. By default, it uses the current working directory for the databaseName part of the value string. As shown in above xml, we use database Name=**/home/me/hive/metastore_db** as the absolute path instead, which is the location where the *metastore_db* directory will always be located. This change eliminates the problem of Hive dropping the *metastore_db* directory in the current working directory every time we start a new Hive session. Now, we'll always have access to all our metadata, no matter what directory we are working in.Distributed

Metastore Methods

The Hive service also connects to the Hive metastore via Thrift. Generally, users should not call metastore methods that modify directly and should only interact with Hive via the HiveQL language. Users should utilize the read-only methods that provide meta-information about tables. For example, the get_partition_names (String,String,short) method can be used to determine which partitions are available to a query:

```
groovy:000> client.get_partition_names("default", "fracture_act", (short)0)
[ hit_date=20120218/mid=001839,hit_date=20120218/mid=001842,
hit_date=20120218/mid=001846 ]
```

It is important to remember that while the metastore API is relatively stable in terms of changes, the methods inside, including their signatures and purpose, can changebetween releases. Hive tries to maintain compatibility in the HiveQL language, which masks changes at these levels.

3 Ways to Access data in HDFS

1)RHadoop
2)SparkR
3)H20

1)R Hadoop
Sys.setenv(HADOOP_CMD="/bin/hadoop")

```
library(rhdfs)
hdfs.init()

f = hdfs.file("fulldata.csv","r",buffersize=104857600)
m = hdfs.read(f)
c = rawToChar(m)

data = read.table(textConnection(c), sep = ",")



reader = hdfs.line.reader("fulldata.csv")

x = reader$read()
typeof(x)
```

ISSUE 1:
JVM is not ready after 10 seconds

solution:
restart R session

C:\apache-hive-2.1.0-bin\conf\hive-site.xml
NOTE:
hdfs - stores table in /user/hive/warehouse based on hive.metastore.warehouse.dir value
hive -stores table based on hive.exec.local.scratchdir value e.g stores metastore in local directory - C:\hive_warehouse\iotmp

```
 <property>
   <name>hive.metastore.warehouse.dir</name>
   <value>/user/hive/warehouse</value>
   <description>location of default database for the warehouse</description>
  </property>

<property>
   <name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:derby:;databaseName=/user/hive/warehouse/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>
```

```
C:\Users\Arun>hdfs dfs -ls /user/hive/warehouse/
Found 6 items
drwxr-xr-x   - Arun supergroup          0 2016-08-03 21:57 /user/hive/warehouse/healthcare.db
drwxr-xr-x   - Arun supergroup          0 2016-08-03 09:24 /user/hive/warehouse/hl7.db
drwxr-xr-x   - Arun supergroup          0 2016-08-03 02:29 /user/hive/warehouse/hl71.db
drwxr-xr-x   - Arun supergroup          0 2016-08-09 00:11 /user/hive/warehouse/hl7_patientdetails
drwxr-xr-x   - Arun supergroup          0 2016-08-09 01:22 /user/hive/warehouse/hl7_patientdetails1
drwxr-xr-x   - Arun supergroup          0 2016-08-09 00:23 /user/hive/warehouse/
```

hl7_patientdetails3

C:\Users\Arun>

C:\Windows\system32>hive
ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
Connecting to jdbc:hive2://
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]
Connected to: Apache Hive (version 2.1.0)
Driver: Hive JDBC (version 2.1.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 2.1.0 by Apache Hive
hive> show tables;
OK
hl7_patientdetails
hl7_patientdetails1
hl7_patientdetails3
3 rows selected (4.802 seconds)
hive>

hit Tab to show the commands exists:

hive> D
DATA DATE
DATETIME_INTERVAL_CODE DATETIME_INTERVAL_PRECISION
DAY DEALLOCATE
DEC DECIMAL
DECLARE DEFAULT
DEFERRABLE DEFERRED
DELETE DESC
DESCRIBE DESCRIPTOR
DIAGNOSTICS DISCONNECT
DISTINCT DOMAIN
DOUBLE DROP

hive> drop table if exists hl7_patientdetails1;

```
OK
No rows affected (12.785 seconds)
hive>
```

Once drop table in hive hdfs also gets Updated:

```
hive> show tables;
OK
hl7_patientdetails
hl7_patientdetails3
2 rows selected (0.197 seconds)
hive>
```

INTERNAL AND EXTERNAL TABLES WITH POPULATE DATA
Create External Tables:

```
hive> CREATE EXTERNAL TABLE EXT_HL7_PatientDetails (key INT, value string) ROW F
ORMAT DELIMITED FIELDS TERMINATED BY '|';
OK
No rows affected (1.057 seconds)
hive>
```

we have 1 ext_hl7_patientdetails (external table) hl7_

```
hive> show tables;
OK
ext_hl7_patientdetails
hl7_patientdetails
hl7_patientdetails3
3 rows selected (0.135 seconds)
hive> drop table hl7_patientdetails3;
OK
No rows affected (1.122 seconds)
hive>
```

Create Internal Tables:

```
hive> CREATE  TABLE INT_HL7_PatientDetails (key INT, value string) ROW FORMAT D
LIMITED FIELDS TERMINATED BY '|';
OK
No rows affected (0.336 seconds)
hive>
```

```
hive> LOAD DATA LOCAL INPATH 'c:/Test/data.txt' OVERWRITE INTO TABLE
INT_HL7_PatientDetails
ientDetails ;
Loading data to table default.int_hl7_patientdetails
OK
No rows affected (0.831 seconds)
hive>
```

Internal and external tables:

```
C:\Users\Arun>hdfs dfs -ls /user/hive/warehouse/*_patientdetails
Found 1 items
-rwxr-xr-x   1 Arun supergroup        31 2016-08-10 23:32 /user/hive/warehouse/
ext_hl7_patientdetails/data.txt
Found 1 items
-rwxr-xr-x   1 Arun supergroup        31 2016-08-10 23:38 /user/hive/warehouse/
int_hl7_patientdetails/data.txt

C:\Users\Arun>
```

HIVE tables:

```
hive> show tables;
OK
ext_hl7_patientdetails
hl7_patientdetails
int_hl7_patientdetails
3 rows selected (0.156 seconds)
hive>
```

INTERNAL & EXTERNAL VIEW IN HDFS :

```
C:\Users\Arun>hdfs dfs -cat /user/hive/warehouse/int_hl7_patientdetails/data.txt

1|Test1
2|Test2
3|Test3


C:\Users\Arun>hdfs dfs -cat /user/hive/warehouse/ext_hl7_patientdetails/data.txt

1|Test1
2|Test2
3|Test3

C:\Users\Arun>
```

INTERNAL & EXTERNAL VIEW IN HIVE:

```
hive> select * from ext_hl7_patientdetails;
OK
1 Test1
2 Test2
3 Test3

3 rows selected (3.464 seconds)
hive> select * from int_hl7_patientdetails;
OK
1 Test1
2 Test2
3 Test3

3 rows selected (0.466 seconds)
hive>
```

UPDATE INTERNAL & EXTERNAL VIEW IN HIVE:

```
hive> LOAD DATA LOCAL INPATH 'c:/Test/data1.txt' OVERWRITE INTO TABLE
INT_HL7_PatientDetails ;
Loading data to table default.int_hl7_patientdetails
OK
No rows affected (0.782 seconds)

hive> LOAD DATA LOCAL INPATH 'c:/Test/data1.txt' OVERWRITE INTO TABLE
EXT_HL7_PatientDetails ;
Loading data to table default.ext_hl7_patientdetails
OK
No rows affected (0.928 seconds)
hive>
```

SELECT INT & EXT TABLES IN HIVE:

```
hive> select * from int_hl7_patientdetails;
OK
4 Test4
5 Test5
6 Test6


5 rows selected (0.436 seconds)
hive> select * from ext_hl7_patientdetails;
OK
4 Test4
5 Test5
6 Test6
```

5 rows selected (0.461 seconds)
hive>

UPDATED INT & EXTERNAL TABLE VIEW IN HDFS

```
C:\Users\Arun>hdfs dfs -ls /user/hive/warehouse/ext_hl7_patientdetails/
Found 1 items
-rwxr-xr-x   1 Arun supergroup        31 2016-08-10 23:50 /user/hive/warehouse/
ext_hl7_patientdetails/data1.txt

C:\Users\Arun>hdfs dfs -ls /user/hive/warehouse/int_hl7_patientdetails/
Found 1 items
-rwxr-xr-x   1 Arun supergroup        31 2016-08-10 23:50 /user/hive/warehouse/
int_hl7_patientdetails/data1.txt
```

HDFS VIEW AFTER UPDATED TABLES

```
C:\Users\Arun>hdfs dfs -cat /user/hive/warehouse/int_hl7_patientdetails/data1.txt

4|Test4
5|Test5
6|Test6

C:\Users\Arun>hdfs dfs -cat /user/hive/warehouse/ext_hl7_patientdetails/data1.txt

4|Test4
5|Test5
6|Test6
C:\Users\Arun>
```

DROP EXT & INT TABLES:

```
hive> drop table ext_hl7_patientdetails;
OK
No rows affected (0.312 seconds)
hive>
```

After drop in hive still hdfs shows records.

```
C:\Users\Arun>hdfs dfs -cat /user/hive/warehouse/ext_hl7_patientdetails/data1.txt
4|Test4
5|Test5
6|Test6

hive> drop table int_hl7_patientdetails;
OK
No rows affected (0.363 seconds)
hive>
```

INTERNAL VS EXT DROP:

```
C:\Users\Arun>hdfs dfs -cat /user/hive/warehouse/int_hl7_patientdetails/data1.txt
cat: `/user/hive/warehouse/int_hl7_patientdetails/data1.txt': No such file or directory

C:\Users\Arun>
```

NOTE:

see when external table dropped – Table not gets dropped in hdfs
when internal table dropped – Table gets dropped in hdfs

But both tables see below is not in meta store I.e hive.

```
Error: Error while compiling statement: FAILED: SemanticException [Error 10001]:
 Line 1:14 Table not found 'int_hl7_patientdetails' (state=42S02,code=10001)
hive> select * from int_hl7_patientdetails;

Error: Error while compiling statement: FAILED: SemanticException [Error 10001]:
 Line 1:14 Table not found 'ext_hl7_patientdetails' (state=42S02,code=10001)
hive> select * from ext_hl7_patientdetails;
```

see only ext_hl7_patientdetail only exist

```
C:\Windows\system32>hdfs dfs -ls /user/hive/warehouse/
Found 6 items
drwxr-xr-x   - Arun supergroup          0 2016-08-10 23:50 /user/hive/warehouse/
ext_hl7_patientdetails
drwxr-xr-x   - Arun supergroup          0 2016-08-03 21:57 /user/hive/warehouse/
healthcare.db
drwxr-xr-x   - Arun supergroup          0 2016-08-03 09:24 /user/hive/warehouse/
hl7.db
drwxr-xr-x   - Arun supergroup          0 2016-08-03 02:29 /user/hive/warehouse/
hl71.db
drwxr-xr-x   - Arun supergroup          0 2016-08-09 00:11 /user/hive/warehouse/
hl7_patientdetails
drwxr-xr-x   - Arun supergroup          0 2016-08-09 01:22 /user/hive/warehouse/
hl7_patientdetails1

C:\Windows\system32>
```

Moving Data from HDFS to Hive Using an External Table

This is the most common way to move data into Hive when the ORC file format is required as the target data format. Then Hive can be used to perform a fast parallel and distributed conversion of your data into ORC.

NOTE:

Tried deleting hive.exec.local.scratchdir (c:\hive_warehouse\iotmp still hdfs showing values

```
<property>
   <name>hive.exec.local.scratchdir</name>
   <value>C:\hive_warehouse\iotmp</value>
   <description>Local scratch space for Hive jobs</description>
  </property>
```

```
C:\Windows\system32>hdfs dfs -cat /user/hive/warehouse/ext_hl7_patientdetails/data1.txt
4|Test4
5|Test5
6|Test6
```

C:\hadoop-2.3.0\etc\hadoop\hdfs-site.xml

Deleted data node values in **C:\yarn_data2.3\dfs\datanode\current** and restart hdfs, hivea

```
<property>
      <name>dfs.replication</name>
      <value>1</value>
   </property>
   <property>
      <name>dfs.namenode.name.dir</name>
      <value>/yarn_data2.3/dfs/namenode</value>
   </property>
   <property>
     <name>dfs.datanode.data.dir</name>
     <value>/yarn_data2.3/dfs/datanode</value>
   </property>
</configuration>
```

After deleting data node configured_C:\yarn_data2.3\dfs\datanode\current

C:\Windows\system32>hdfs dfs -cat /user/hive/warehouse/ext_hl7_patientdetails/da

ta1.txt

cat: Zero blocklocations for /user/hive/warehouse/ext_hl7_patientdetails/data1.t

xt. Name node is in safe mode.

C:\Windows\system32>

showing same message after deleting lock file alone in **C:\yarn_data2.3\dfs\datanode\in_use.lock.**

After formatting Namenode:

C:\Windows\system32>hdfs dfs -cat /user/hive/warehouse/ext_hl7_patientdetails/da

ta1.txt

cat: `/user/hive/warehouse/ext_hl7_patientdetails/data1.txt': No such file or di

rectory

NOTE:

After formatting name node no metadata exists in namenode hence err msg changed from "Zero blocklocations for /user/hive/warehouse/ext_hl7_patientdetails/data1.t

xt. Name node is in safe mode."

to `/user/hive/warehouse/ext_hl7_patientdetails/data1.txt': No such file or di

rectory

EXTERNAL TABLE CREATION WITH FILE LOCATION:

hive> CREATE EXTERNAL TABLE EXT_WITH_LOC_HL7_PatientDetails (key INT, value string) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' LOCATION '/hive_warehouse/tables';

OK

No rows affected (1.243 seconds)

BEFORE TABLE CREATION WITH LOCATION

C:\Users\Arun>hdfs dfs -ls /tables ls: `/tables': No such file or directory
C:\Users\Arun>

C:\Windows\system32>hdfs dfs -ls /

Found 2 items

drwxr-xr-x - Arun supergroup 0 2016-08-11 01:53 /hive

drwx-wx-wx - Arun supergroup 0 2016-08-11 01:44 /tmp

C:\Windows\system32>

hive> CREATE TABLE INT_WITH_LOC_HL7_PatientDetails3 (key INT, value string) ROW

FORMAT DELIMITED FIELDS TERMINATED BY '|' LOCATION '/hive/table/int';

OK

No rows affected (0.423 seconds)

C:\Windows\system32>hdfs dfs -ls /hive/table/

Found 1 items

drwxr-xr-x - Arun supergroup 0 2016-08-11 01:55 /hive/table/int

C:\Windows\system32>

when giving location it creates a directory inside hdfs automatically without hdfs dfs -mkdir command.

Connecting to HiveServer2

The method that HiveServer2 clients use to connect to HiveServer2 is based on the HiveServer2 Authentication method and the type of client:

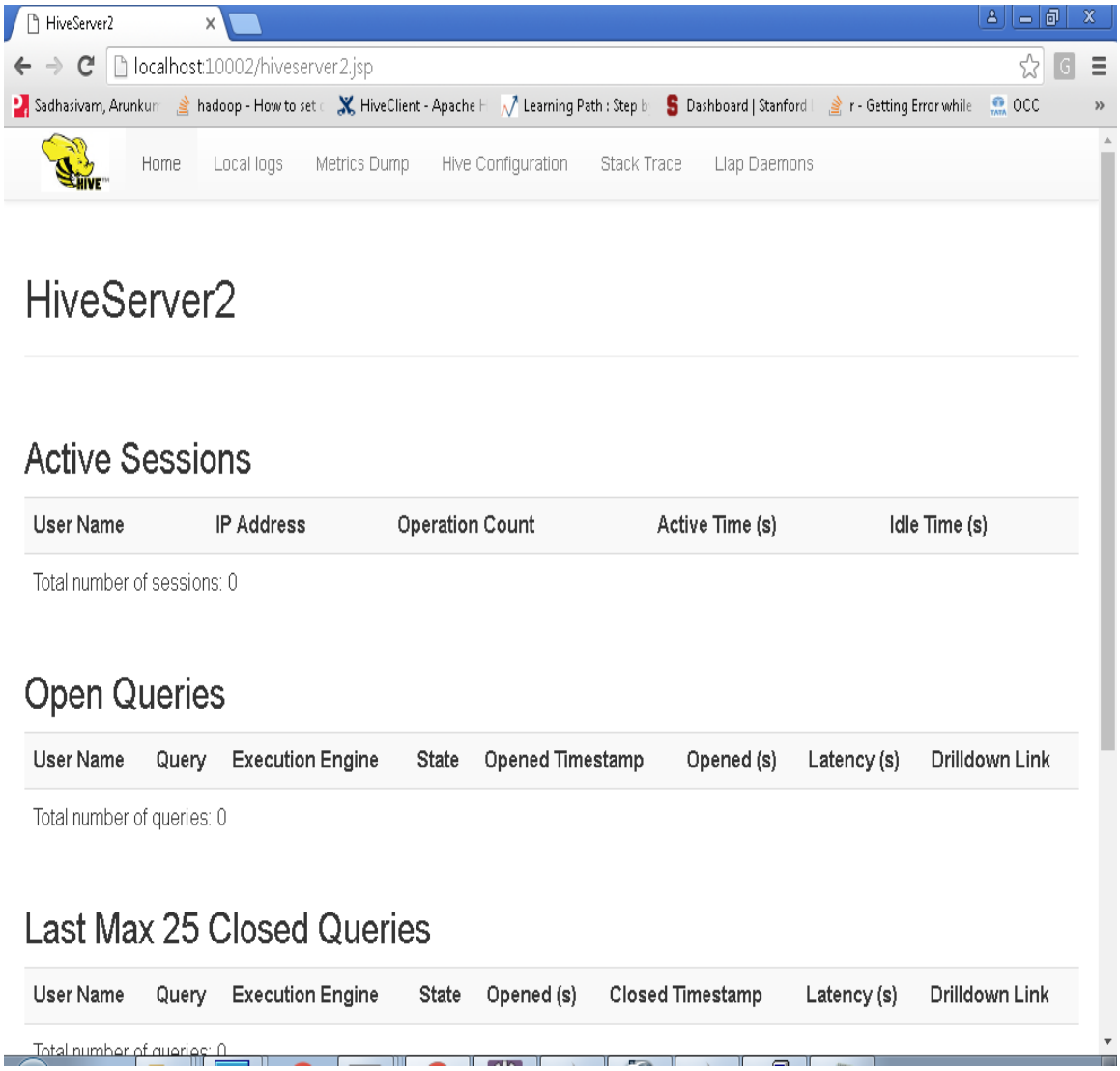
- Using ODBC to Connect to HiveServer2
- Using JDBC or Beeline to Connect to HiveServer2

Using JDBC or Beeline to Connect to HiveServer2

HiveServer2 Authentication	Connection Requirements
No Authentication	**Connection String:** jdbc:hive2://<hostname>:10000/default **For encryption, JDBC requires a truststore and an optional truststore password.** **Connection String with Encryption:** jdbc:hive2://<host>:<port>/<database>;ssl=true;sslTrustStore=<path-to-truststore>;sslTrustStorePassword=<password> **Connection String with Encryption (truststore passed in JVM arguments):** jdbc:hive2://<host>:<port>/<database>;ssl=true Prior to connecting to an application that uses JDBC,such as Beeline, you can run the following command to pass the truststore parameters as java arguments: export HADOOP_OPTS="-Djavax.net.ssl.trustStore=<path-to-trust-store-file> -Djavax.net.ssl.trustStorePassword=<password>"
MapR-SASL	**Connection String:** jdbc:hive2://<hostname>:10000/default;auth=maprsasl **Connection String with Encryption (Hive 0.13 version):** jdbc:hive2://<hostname>:10000/default;auth=maprsasl;sasl.qop=auth-conf **Connection String with Encryption (Hive 1.0 version and above):** jdbc:hive2://<hostname>:10000/default;auth=maprsasl;saslQop=auth-conf **Connection for Java Application:** Use the -D flag to append the JVM argument: `-Dhadoop.login=maprsasl`
PAM	**Connection String:** jdbc:hive2://hs2node:10000/default;user=<userid>;password=<password>
Kerberos	**Connection String:** jdbc:hive2://<hostname>:10000/default;principal=mapr/<FQDN@REALM> **Connection String with Encryption (Hive 0.13 version):** jdbc:hive2://<hostname>:10000/default;principal=mapr/<FQDN@REALM>;sasl.qop=auth-conf **Connection String with Encryption (Hive 1.0 version and above):** jdbc:hive2://<hostname>:10000/default;principal=mapr/<FQDN@REALM>;saslQop=auth-conf

HiveServer2 Authentication	Connection Requirements
	Connection for Java Application: Use the -D flag to append the JVM argument: `-Dhadoop.login=hybrid` The client nodes must also have a Kerberos ticket and be configured to connect to HiveServer to using Kerberos. See Example: Generating a Kerberos Ticket and Authentication for HiveServer2.
LDAP	**Connection String:** jdbc:hive2://hs2node:10000/default;user=<userid>;password=<password>

Run hiveserver 2 using Http://localhost:1002/



HiveServer2 Beeline Introduction

2

This entry was posted in Hive *on March 14, 2015 by Siva*

In this post we will discuss about HiveServer2 Beeline Introduction. As of hive-0.11.0, Apache Hive started decoupling HiveServer2 from Hive. It is because of overcoming the existing Hive Thrift Server.

Table of Contents [hide]

Below are the Limitations of Hive Thrift Server 1

- No Sessions/Concurrency
- Essentially need 1 server per client
- Security
- Client Interface
- Stability

Sessions/Currency

- Old Thrift API and server implementation didn't support concurrency.

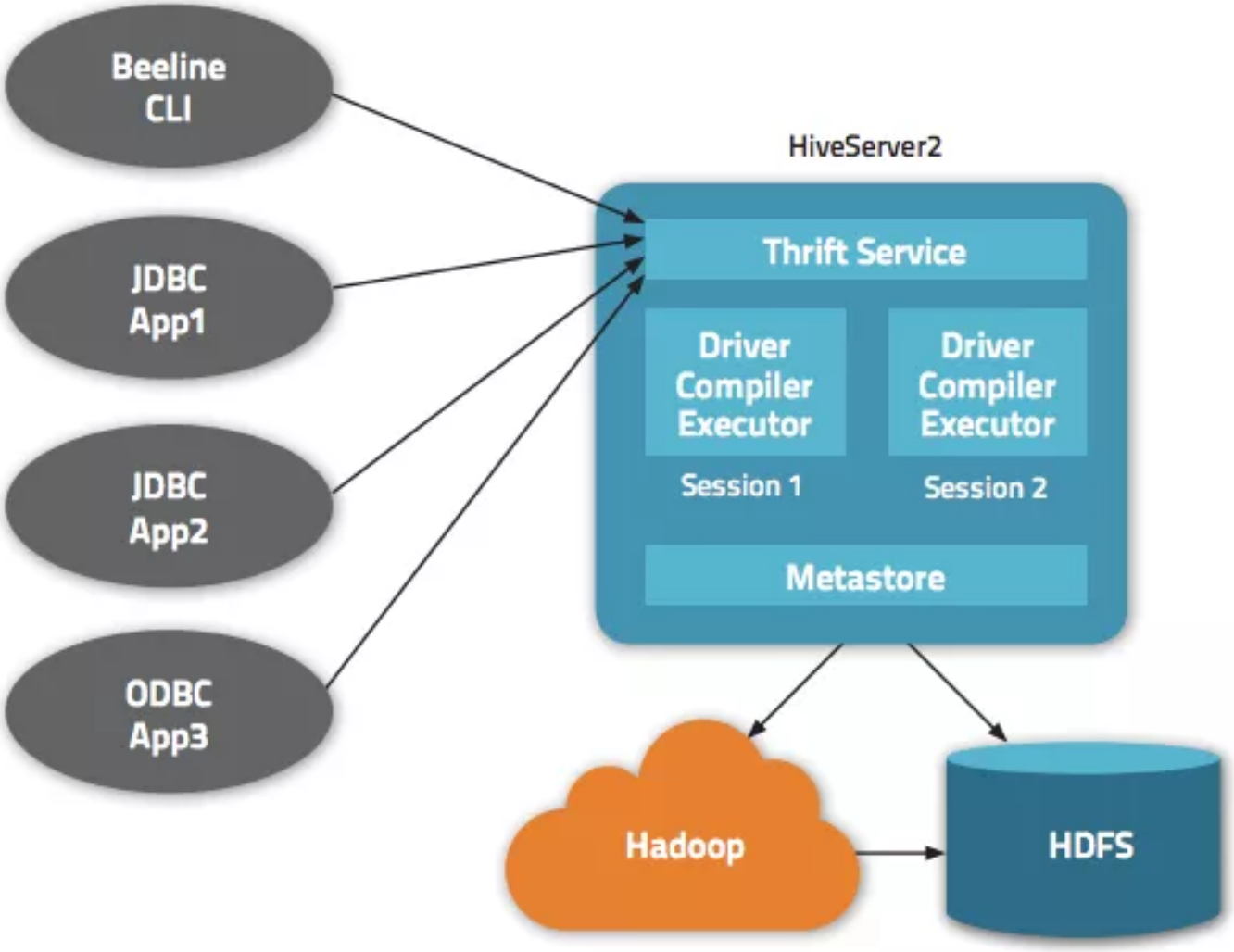
Authentication/Authorization

- Incomplete implementations of Authentication (verifying the identity of the user) and Authorizations (Verifying if user has permissions to perform this action).

HiveServer2

HiveServer2 is a container for the Hive execution engine (Driver). For each client connection, it creates a new execution context (Connection and Session) that serves Hive SQL requests from the client. The new RPC interface enables the server to associate this Hive execution context with the thread serving the client's request.

Below is the high level architecture of HiveServer2.



Sourced from cloudera.com

Run HiveServer2:

We can start Thrift HiveServer2 service with the below command if hive-0.11.0 or above is installed in our machine.

```
hadoop1@ubuntu-1:~$ hive --service hiveserver2
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/hadoop-2.3.0/share/hadoop/common/lib/slf4
SLF4J: Found binding in [jar:file:/usr/lib/hive/apache-hive-0.14.0-bin/lib/hive-jdbc-0.14.0
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
```

If we need to customize HiveServer2, we can set below properties in hive-site.xml file.

```
hive.server2.transport.mode = binary | http | https
```

```
hive.server2.thrift.port = 10000
hive.server2.thrift.bind.host
hive.server2.thrift.min.worker.threads = 5
hive.server2.thrift.max.worker.threads = 500
hive.server2.async.exec.threads = 50
hive.server2.async.exec.shutdown.timeout = 10 (seconds)
hive.support.concurrency = true
```

Start Beeline Client for HiveServer2:

We can start the client service for HiveServer2 from various clients like SQLine, Beeline or Squirrel or Web Interface. But in this we will see how to connect to HiveServer2 via Beeline client.

Below command can be used to connect to HiveServer2.

```
Beeline Client Interface                                                    MySQL
hadoop1@ubuntu-1:~$ beeline
Beeline version 0.14.0 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000
scan complete in 32ms
Connecting to jdbc:hive2://localhost:10000
Enter username for jdbc:hive2://localhost:10000: hadoop1
Enter password for jdbc:hive2://localhost:10000: ********
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/lib/hadoop/hadoop-2.3.0/share/hadoop/common/lib/slf
SLF4J: Found binding in [jar:file:/usr/lib/hive/apache-hive-0.14.0-bin/lib/hive-jdbc-0.14.
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Connected to: Apache Hive (version 0.14.0)
Driver: Hive JDBC (version 0.14.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://localhost:10000> show tables;
+-----------------------+--+
|       tab_name        |
+-----------------------+--+
| apache_combined_log   |
| apache_common_log     |
| docs                  |
| docs2                 |
| docs3                 |
| hadoop_log            |
| hbase_table_emp       |
| hbase_table_user      |
| hcat_table1           |
| increment_table1      |
| partitioned_user      |
| posts                 |
| posts2                |
| posts3                |
+-----------------------+--+
12 rows selected (3.353 seconds)
0: jdbc:hive2://localhost:10000> select * from posts3;
+--------------+-------------------+----------------+--+
| posts3.user  |    posts3.post    |  posts3.time   |
```

About Siva

Senior Hadoop developer with 4 years of experience in designing and architecture solutions for the Big Data domain and has been involved with several complex engagements. Technical strengths include Hadoop, YARN, Mapreduce, Hive, Sqoop, Flume, Pig, HBase, Phoenix, Oozie, Falcon, Kafka, Storm, Spark, MySQL and Java.

View all posts by Siva →

Leave a comment

Your email address will not be published. Required fields are marked *

Visual Text

File Edit Insert View Format Table Tools

p

Name *

Email *

Website

Post Comment

2 thoughts on “HiveServer2 Beeline Introduction”

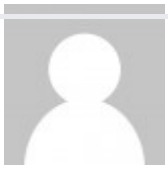
Lekan

Reply ↓

April 13, 2015 at 1:18 pm

Hi, Any time I run the bin/hiveserver2 command, I get this response
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hadoop/hive/lib/hive-jdbc-1.0.0-standalone.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.slf4j.impl.Log4jLoggerFactory]

And it never loads further. I have tried connecting via PHP client but it does not return anything. Why does it hang there?



Eswar

Reply ↓

June 21, 2016 at 4:29 am

Hi Siva,

Thank you for the beeline introduction.

Currently we are using Hive CLI on my cluster. we are planning to disable Hive CLI by bringing up beeline for the security purpose.

Could you please guide me the process to disable the current Hive CLI version from the existing cluster.

-> What is the process to bring up beeline with out any impact of the current databases , tables and data.

-> what are the properties needs to be change in hive-site.xml file. Please specify the properties .

Please suggest me the process on this scenario.

Thank you in advance !!!

Post navigation

← Mapreduce Use Case for N-Gram Statistics

Hive JDBC Client Example →

What Experts say ?

[testimonials_cycle show_thumbs="1" pager="1" easy_testimonial_title="1" itemprop="name"]

Search

Search

Core Hadoop

Big Data

Hadoop

Map Reduce

EcoSystem Tools

Hive

Pig

Hbase

Impala

Next Batch on Hadoop from 5th August

Next Batch on Hadoop Developer Online Training starts around 5th of August 2016. If any one interested to attend this batch please register by sending email to me on siv535@gmail.com.

Training Course Includes below topics:

For Complete details - Refer Link

1) Big Data & Hadoop Introduction
2) Linux Basics
3) Core Java Essentials
4) HDFS
5) Map Reduce & YARN
6) Pig
7) Hive
8) Impala
9) HBase
10) Sqoop
11) Flume
12) Oozie

13) Hue
14) Cloudera Manager
15) Real Time projects

if there are any doubts or questions call on +91-9704231873.

Next Batch on Spark from 29th July

Next Batch on Hadoop Developer Online Training starts around 29th of July. If any one interested to attend this batch please register by sending email to me on siv535@gmail.com.

For Complete details - Refer Link

Training Course Includes below topics:

1) Scala
2) Spark
3) Kafka
4) Real Time projects

if there are any doubts or questions call
704231873.

.

Recent Comments

Contat Us

Call Us On : +91-9704231873
Mail Us On : siv535@gmail.com

Email ID

Your email address

Send

Let’s get Social :

Back to top

ISSUE:

```
C:\apache-hive-2.1.0-bin\lib>beeline
Beeline version 1.6.1 by Apache Hive
Exception in thread "main" java.lang.NoSuchMethodError:
org.fusesource.jansi.int
ernal.Kernel32.GetConsoleOutputCP()I
	at
jline.WindowsTerminal.getConsoleOutputCodepage(WindowsTerminal.java:2
93)
	at jline.WindowsTerminal.getOutputEncoding(WindowsTerminal.java:186)
	at jline.console.ConsoleReader.<init>(ConsoleReader.java:230)
	at jline.console.ConsoleReader.<init>(ConsoleReader.java:221)
	at jline.console.ConsoleReader.<init>(ConsoleReader.java:209)
	at org.apache.hive.beeline.BeeLine.getConsoleReader(BeeLine.java:834)
	at org.apache.hive.beeline.BeeLine.begin(BeeLine.java:770)
	at org.apache.hive.beeline.BeeLine.mainWithInputRedirection(BeeLine.java
:484)
	at org.apache.hive.beeline.BeeLine.main(BeeLine.java:467)

C:\apache-hive-2.1.0-bin\lib>
```

SOLUTION:
Debugging steps
Step 1:
go to C:\apache-hive-2.1.0-bin\bin\beeline.cmd
Turn on debug by changing first line in beeline.cmd to **@echo on** by default
@echo will be off

step 2:
go to bin path of beeline.cmd then only will show error correctly
C:\>cd C:\apache-hive-2.1.0-bin
C:\apache-hive-2.1.0-bin>cd bin
step 3:

```
C:\apache-hive-2.1.0-bin\bin>beeline>c:/arun.txt
File Not Found
Exception in thread "main" java.lang.NoClassDefFoundError:
org/apache/hadoop/hiv
e/conf/HiveConf
	at java.lang.Class.getDeclaredMethods0(Native Method)
	at java.lang.Class.privateGetDeclaredMethods(Class.java:2615)
	at java.lang.Class.getMethod0(Class.java:2856)
	at java.lang.Class.getMethod(Class.java:1668)
```

```
        at sun.launcher.LauncherHelper.getMainMethod(LauncherHelper.java:494)
        at
sun.launcher.LauncherHelper.checkAndLoadMain(LauncherHelper.java:486)

Caused by: java.lang.ClassNotFoundException:
org.apache.hadoop.hive.conf.HiveCon
f
        at java.net.URLClassLoader$1.run(URLClassLoader.java:366)
        at java.net.URLClassLoader$1.run(URLClassLoader.java:355)
        at java.security.AccessController.doPrivileged(Native Method)
        at java.net.URLClassLoader.findClass(URLClassLoader.java:354)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:425)
        at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:308)
        at java.lang.ClassLoader.loadClass(ClassLoader.java:358)
        ... 6 more

C:\apache-hive-2.1.0-bin\bin>
```

step 4:

checking the generated output

```
"started....."
C:\apache-hive-2.1.0-bin\bin>SetLocal EnableDelayedExpansion
C:\apache-hive-2.1.0-bin\bin>pushd C:\apache-hive-2.1.0-bin\bin\..
C:\apache-hive-2.1.0-bin>if not defined HIVE_HOME (set
HIVE_HOME=C:\apache-hive-2.1.0-bin )
C:\apache-hive-2.1.0-bin>popd
C:\apache-hive-2.1.0-bin\bin>if "~-1" == "\" (set HADOOP_BIN_PATH=~0,-1 )
C:\apache-hive-2.1.0-bin\bin>if not defined JAVA_HOME (
echo Error: JAVA_HOME is not set.
 goto :eof
)
C:\apache-hive-2.1.0-bin\bin>if not exist C:\hadoop-2.3.0\libexec\hadoop-
config.cmd (
 exit /b 1
)
```

hive-beeline-2.1.0.jar

```
C:\apache-hive-2.1.0-bin\bin>set HADOOP_HOME_WARN_SUPPRESS=true
C:\apache-hive-2.1.0-bin\bin>pushd C:\apache-hive-2.1.0-bin\lib
C:\apache-hive-2.1.0-bin\lib>for /F %a IN ('dir /b hive-beeline-**.jar') do (set
HADOOP_CLASSPATH=;C:\apache-hive-2.1.0-bin\lib\%a )
```

C:\apache-hive-2.1.0-bin\lib>(**set HADOOP_CLASSPATH=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-2.1.0.jar**)

super-csv-2.2.0.jar

C:\apache-hive-2.1.0-bin\lib>for /F %a IN ('dir /b **super-csv-**.jar**') do (set HADOOP_CLASSPATH=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-2.1.0.jar;C:\apache-hive-2.1.0-bin\lib\%a)

C:\apache-hive-2.1.0-bin\lib>(**set HADOOP_CLASSPATH**=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-2.1.0.jar;C:\apache-hive-2.1.0-bin\lib\super-csv-2.2.0.jar)

jline-2.14.2.jar

C:\apache-hive-2.1.0-bin\lib>for /F %a IN ('dir /b **jline-**.jar**') do (set HADOOP_CLASSPATH=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-2.1.0.jar;C:\apache-hive-2.1.0-bin\lib\super-csv-2.2.0.jar;C:\apache-hive-2.1.0-bin\lib\%a)

C:\apache-hive-2.1.0-bin\lib>(**set HADOOP_CLASSPATH**=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-2.1.0.jar;C:\apache-hive-2.1.0-bin\lib\super-csv-2.2.0.jar;C:\apache-hive-2.1.0-bin\lib\jline-2.14.2.jar)

hive-jdbc<<version>>-standalone.jar

C:\apache-hive-2.1.0-bin\lib>for /F %a IN ('dir /b **hive-jdbc-**-standalone.jar**') do (**set HADOOP_CLASSPATH**=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-2.1.0.jar;C:\apache-hive-2.1.0-bin\lib\super-csv-2.2.0.jar;C:\apache-hive-2.1.0-bin\lib\jline-2.14.2.jar;**C:\apache-hive-2.1.0-bin\lib\%a**)

C:\apache-hive-2.1.0-bin\lib>popd
C:\apache-hive-2.1.0-bin\bin>set HADOOP_USER_CLASSPATH_FIRST=true
C:\apache-hive-2.1.0-bin\bin>call C:\hadoop-2.3.0\libexec\hadoop-config.cmd

Reason for issue :

for /F %a IN ('dir /b hive-jdbc-**-standalone.jar')do (**set HADOOP_CLASSPATH**=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-2.1.0.jar;C:\apache-hive-2.1.0-bin\lib\super-csv-2.2.0.jar;C:\apache-hive-2.1.0-bin\lib\jline-2.14.2.jar;**C:\apache-hive-2.1.0-bin\lib\%a**)

see from above for %a is not able to find any jar hence it append %a in path.
Hence only **hive-jdbc<<any version>>standalone.jar missing** need to be in classpath I.e inside C:\apache-hive-2.1.0-bin\lib
push takes to one dir before like cd ..

Explanation:

```
C:\apache-hive-2.1.0-bin\bin>pushd C:\apache-hive-2.1.0-bin\bin\..

C:\apache-hive-2.1.0-bin>
```

you can run each line one by one also

```
C:\apache-hive-2.1.0-bin\lib>for /F %a IN ('dir /b hive-beeline-**.jar') do (set
 HADOOP_CLASSPATH=;C:\apache-hive-2.1.0-bin\lib\%a )
C:\apache-hive-2.1.0-bin\lib>(set HADOOP_CLASSPATH=;C:\apache-hive-
2.1.0-bin\lib\hive-beeline-2.1.0.jar  )
C:\apache-hive-2.1.0-bin\lib>
```

To check each line just echo instead of set classpath

```
C:\apache-hive-2.1.0-bin\lib>for /F %a IN ('dir /b hive-beeline-**.jar') do (ech
o %a%)
```

output:

```
C:\apache-hive-2.1.0-bin\lib>(echo hive-beeline-2.1.0.jar% )
hive-beeline-2.1.0.jar%

C:\apache-hive-2.1.0-bin\lib>
```

above **command iterates over lib** dir and find if any lib inside dir having format **hive-beeline-**jar** and set it to class path .
HADOOP_CLASSPATH=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-2.1.0.jar

```
C:\apache-hive-2.1.0-bin\lib>for /F %a IN ('dir /b hive-jdbc-**-standalone.jar')
 do (set HADOOP_CLASSPATH=;C:\apache-hive-2.1.0-bin\lib\hive-beeline-
2.1.0.jar;C
:\apache-hive-2.1.0-bin\lib\super-csv-2.2.0.jar;C:\apache-hive-2.1.0-bin\lib\jli
ne-2.14.2.jar;C:\apache-hive-2.1.0-bin\lib\%a )

C:\apache-hive-2.1.0-bin\lib>(set HADOOP_CLASSPATH=;C:\apache-hive-
2.1.0-bin\lib
\hive-beeline-2.1.0.jar;C:\apache-hive-2.1.0-bin\lib\super-csv-2.2.0.jar;C:\apac
he-hive-2.1.0-bin\lib\jline-2.14.2.jar;C:\apache-hive-2.1.0-bin\lib\hive-jdbc-1.
2.1-standalone.jar  )
```

see now C:\apache-hive-2.1.0-bin\lib\hive-jdbc-1.2.1-standalone.jar now gets generated.

C:\apache-hive-2.1.0-bin\bin>beeline>c:/arun.txt

NOTE:

when pipe output to file only it shows error in terminal from which you can identify which is error and logs will be written to txt .when trying again it shows below error.

```
C:\apache-hive-2.1.0-bin\bin>beeline>c:/arun.txt
Error: Could not find or load main class ;C:\apache-hive-2.1.0-bin\lib\hive-beel
ine-2.1.0.jar;C:\apache-hive-2.1.0-bin\lib\super-csv-2.2.0.jar;C:\apache-hive-2.
1.0-bin\lib\jline-2.14.2.jar;C:\apache-hive-2.1.0-bin\lib\hive-jdbc-1.2.1-standa
lone.jar;C:\hadoop-2.3.0\etc\hadoop;C:\hadoop-
2.3.0\share\hadoop\common\lib\*;C:
\hadoop-2.3.0\share\hadoop\common\*;C:\hadoop-
2.3.0\share\hadoop\hdfs;C:\hadoop-
2.3.0\share\hadoop\hdfs\lib\*;C:\hadoop-2.3.0\share\hadoop\hdfs\*;C:\hadoop-2.3.
0\share\hadoop\yarn\lib\*;C:\hadoop-2.3.0\share\hadoop\yarn\*;C:\hadoop-
2.3.0\sh
are\hadoop\mapreduce\lib\*;C:\hadoop-2.3.0\share\hadoop\mapreduce\*;
```

STEP 5:

check all dependent cmd in beeline.cmd . It has hadoop-config.cmd after adding **hive-jdbc-1.2.1-standalone.jar to classpath .** All Error related to beeline.cmd fixed only issue in dependent cmd **(hadoop-config.cmd,hadoop-env.cmd)** . check associated dependent cmd files while in this case hadoop-config.cmd.

```
call %HADOOP_HOME%\libexec\hadoop-config.cmd
call %JAVA_HOME%\bin\java %JAVA_HEAP_MAX% %HADOOP_OPTS%
-classpath %CLASSPATH% org.apache.hive.beeline.BeeLine %*
```

Turn on **hadoop-config.cmd**

```
C:\hadoop-2.3.0\libexec>if exist C:\hadoop-2.3.0\etc\hadoop\hadoop-env.cmd
(call
 C:\hadoop-2.3.0\etc\hadoop\hadoop-env.cmd )
Error: Could not find or load main class org.apache.hadoop.util.PlatformName
C:\hadoop-2.3.0\libexec>hadoop-config>c:/arun.txt
Error: Could not find or load main class org.apache.hadoop.util.PlatformName
C:\hadoop-2.3.0\libexec>
```

when pipe output to file last for beeline start up and dependencies,only actual root cause of error is beeline.cmd dependency file hadoop-config.cmd

```
C:\hadoop-2.3.0\libexec>hadoop-config.cmd>c:/arun1.txt
Error: Could not find or load main class org.apache.hadoop.util.PlatformName
C:\hadoop-2.3.0\libexec>
```

line which has PlatformName below

```
for /f "delims=" %%A in ('%JAVA% -Xmx32m
%HADOOP_JAVA_PLATFORM_OPTS% -classpath "%CLASSPATH%"
org.apache.hadoop.util.PlatformName') do set JAVA_PLATFORM=%%A
```

beeline.cmd – sets hadoop and hive in classpath
hadoo-config.cmd – sets all hadoop files in classpath
hadoop-env.cmd – sets java and other in classpath. Sets heap size

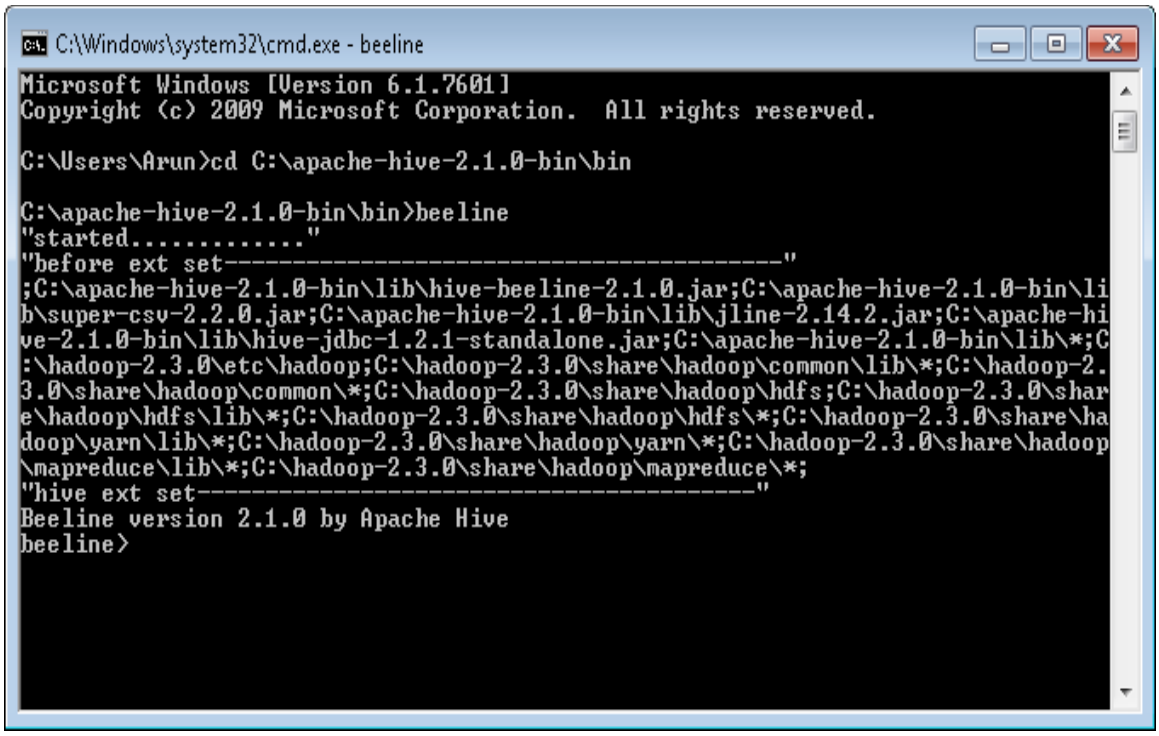
C:\hadoop-2.3.0\libexec**hadoop-config.cmd**
C:\hadoop-2.3.0\etc\hadoop**hadoop-env.cmd**
C:\apache-hive-2.1.0-bin\bin**beeline.cmd**

beeline.cmd when running shows hive error **"Exception in thread "main" java.lang.NoSuchMethodError: org.fusesource.jansi.internal.Kernel32.GetConsoleOutputCP()I"**
Reason is hive is not able to load all jars and set in classpath since below script sets only jdbc, beeline,supercsv,jline. Hence below line set all jars in hive to classpath to make it work!!!.

```
 set HADOOP_CLASSPATH=%HADOOP_CLASSPATH%;
%HIVE_HOME%\lib\*
```

Beeline started by adding below line in C:\apache-hive-2.1.0-bin\bin\beeline.cmd

```
pushd %HIVE_HOME%\jdbc –newly added change dir to jbdc

for /f %%a IN ('dir /b hive-jdbc-**-standalone.jar') do (
 set HADOOP_CLASSPATH=%HADOOP_CLASSPATH%;
%HIVE_HOME%\jdbc\%%a
)
popd

pushd %HIVE_HOME%\lib  –newly added

–newly added to add all jars inside lib to classpath.
set HADOOP_CLASSPATH=%HADOOP_CLASSPATH%;
%HIVE_HOME%\lib\*

set HADOOP_USER_CLASSPATH_FIRST=true
call %HADOOP_HOME%\libexec\hadoop-config.cmd
```

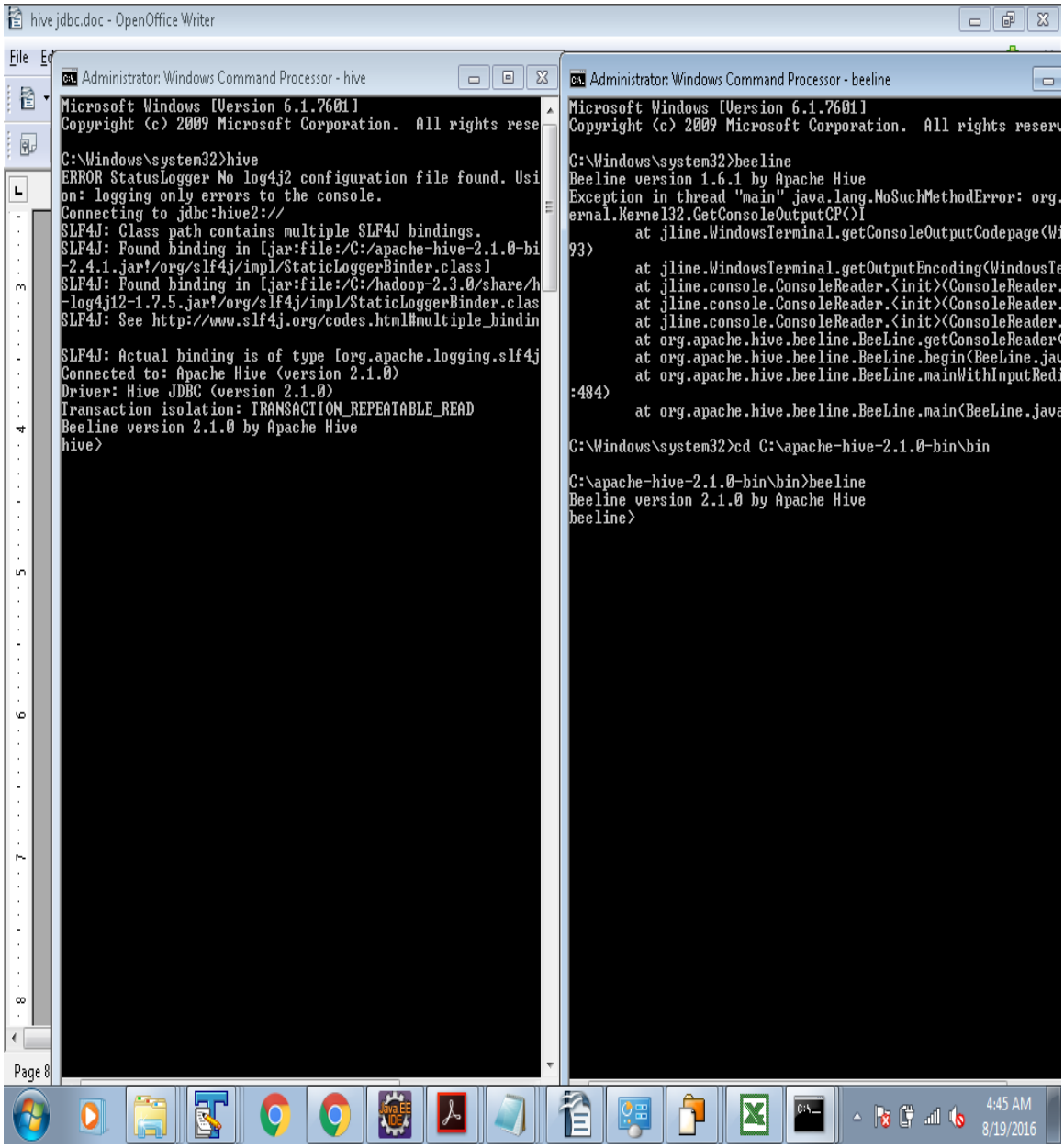


beeline only works when run from bin folder C:\apache-hive-2.1.0-bin\bin

PATH:

%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\;C:\Program Files\Intel\WiFi\bin\;C:\Program Files\Common Files\Intel\WirelessCommon\;C:\Program Files (x86)\Skype\Phone\;C:\apache-maven-3.3.9\bin;;C:\Program Files\Git\bin;C:\Progra~1\Java\jdk1.7.0_79\bin;C:\Program Files\R\R-3.2.3\bin;C:\scala-2.11.8\bin;C:\SBT-0.13\bin;C:\protoc;C:\cygwin64\bin;C:\hadoop-2.3.0\bin;C:\hadoop-2.3.0\sbin;C:\spark-1.6.1-bin-hadoop2.3\bin;C:\spark-1.6.1-bin-hadoop2.3\sbin;C:\zeppelin-0.5.5\bin;C:\pig-0.15.0\bin;C:\apache-hive-2.1.0-bin\bin;C:\Program Files\MongoDB\Server\3.2\bin;C:\db-derby-10.12.1.1-bin\bin;C:\zeppelin-0.6.0-bin-all\bin;

Env	value
HADOOP_CONF_DIR	C:\hadoop-2.3.0\etc
HADOOP_HOME	C:\hadoop-2.3.0
HADOOP_USER_CLASSPATH_FIRST	TRUE
HADOOP_YARN_HOME	C:\hadoop-2.3.0
HIVE_CONF_DIR	C:\apache-hive-2.1.0-bin\conf
HIVE_HOME	C:\apache-hive-2.1.0-bin
java.io.tmpdir	c:/tmp
JAVA_HOME	C:\Progra~1\Java\jdk1.7.0_79
TMP	%USERPROFILE%\AppData\Local\Temp
ZEPPELIN_HOME	C:\zeppelin-0.5.5



Administrator: Windows Command Processor - beeline

```
        at jline.WindowsTerminal.getOutputEncoding(WindowsTerminal.java:186)
        at jline.console.ConsoleReader.<init>(ConsoleReader.java:230)
        at jline.console.ConsoleReader.<init>(ConsoleReader.java:221)
        at jline.console.ConsoleReader.<init>(ConsoleReader.java:209)
        at org.apache.hive.beeline.BeeLine.getConsoleReader(BeeLine.java:834)
        at org.apache.hive.beeline.BeeLine.begin(BeeLine.java:770)
        at org.apache.hive.beeline.BeeLine.mainWithInputRedirection(BeeLine.java
:484)
        at org.apache.hive.beeline.BeeLine.main(BeeLine.java:467)

C:\Windows\system32>cd C:\apache-hive-2.1.0-bin\bin

C:\apache-hive-2.1.0-bin\bin>beeline
Beeline version 2.1.0 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000
Connecting to jdbc:hive2://localhost:10000
Enter username for jdbc:hive2://localhost:10000: hadoop1
Enter password for jdbc:hive2://localhost:10000: *******
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/jdbc/hive-jdbc-2.1.0
-standalone.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]
ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
Error: Could not open client transport with JDBC Uri: jdbc:hive2://localhost:100
00: java.net.ConnectException: Connection refused: connect (state=08S01,code=0)
beeline> CREATE USER 'hiveuser'@'%' IDENTIFIED BY 'hivepassword';
No current connection
beeline> mysql
. . . .> ;
No current connection
beeline> !connect jdbc:hive2://localhost:10000
Connecting to jdbc:hive2://localhost:10000
Enter username for jdbc:hive2://localhost:10000: APP
Enter password for jdbc:hive2://localhost:10000: ****
Error: Could not open client transport with JDBC Uri: jdbc:hive2://localhost:100
00: java.net.ConnectException: Connection refused: connect (state=08S01,code=0)
beeline> !connect jdbc:hive2://localhost:10000/healthcare
Connecting to jdbc:hive2://localhost:10000/healthcare
Enter username for jdbc:hive2://localhost:10000/healthcare: APP
Enter password for jdbc:hive2://localhost:10000/healthcare: ****
Error: Could not open client transport with JDBC Uri: jdbc:hive2://localhost:100
00/healthcare: java.net.ConnectException: Connection refused: connect (state=08S
01,code=0)
beeline> !connect jdbc:hive2://localhost:10000/healthcare
Connecting to jdbc:hive2://localhost:10000/healthcare
Enter username for jdbc:hive2://localhost:10000/healthcare:
Enter password for jdbc:hive2://localhost:10000/healthcare:
Error: Could not open client transport with JDBC Uri: jdbc:hive2://localhost:100
00/healthcare: java.net.ConnectException: Connection refused: connect (state=08S
01,code=0)
beeline>
```

Configuring the Hive Metastore

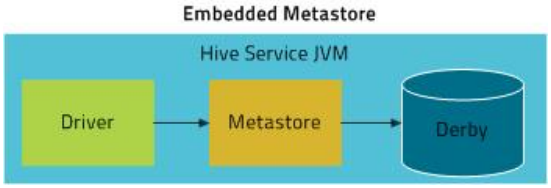
The Hive metastore service stores the metadata for Hive tables and partitions in a relational database, and provides clients (including Hive) access to this information via the metastore service API. The subsections that follow discuss the deployment options and provide instructions for setting up a database in a recommended configuration.

Metastore Deployment Modes

> **Note**:
> **HiveServer** in the discussion that follows refers to HiveServer1 or HiveServer2, whichever you are using.

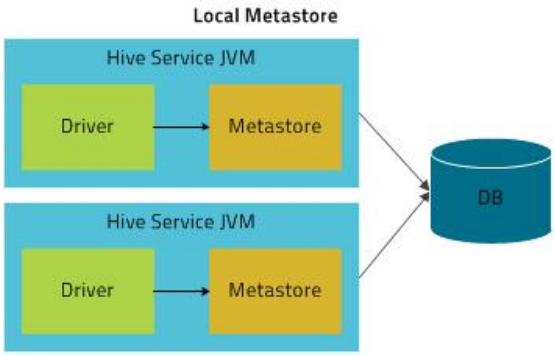
Embedded Mode

Cloudera recommends using this mode for experimental purposes only.



This is the default metastore deployment mode for CDH. In this mode the metastore uses a Derby database, and both the database and the metastore service run embedded in the main HiveServer process. Both are started for you when you start the HiveServer process. This mode requires the least amount of effort to configure, but it can support only one active user at a time and is not certified for production use.

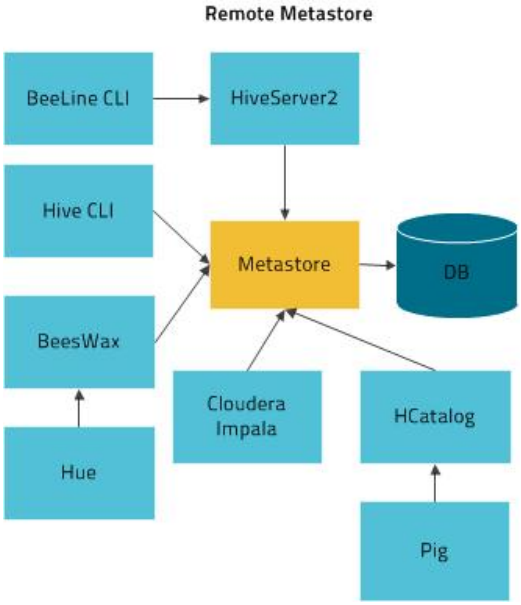
Local Mode



In this mode the Hive metastore service runs in the same process as the main HiveServer process, but the metastore database runs in a separate process, and can be on a separate host. The embedded metastore service communicates with the metastore database over JDBC.

Remote Mode

Cloudera recommends that you use this mode.



In this mode the Hive metastore service runs in its own JVM process; HiveServer2, HCatalog, Cloudera Impala™, and other processes communicate with it via the Thrift network API (configured via the `hive.metastore.uris` property). The metastore service communicates with the metastore database over JDBC (configured via the `javax.jdo.option.ConnectionURL` property). The database, the HiveServer process, and the metastore service can all be on the same host, but running the HiveServer process on a separate host provides better availability and scalability.

The main advantage of Remote mode over Local mode is that Remote mode does not require the administrator to share JDBC login information for the metastore database with each Hive user. **HCatalog** requires this mode.

Supported Metastore Databases

See the **CDH4 Requirements and Supported Versions page** for up-to-date information on supported databases. Cloudera strongly encourages you to use MySQL because it is the most popular with the rest of the Hive user community, and so receives more testing than the other options.

Configuring the Metastore Database

This section describes how to configure Hive to use a remote database, with examples for **MySQL** and **PostgreSQL**.

The configuration properties for the Hive metastore are documented on the **Hive Metastore documentation** page, which also includes a pointer to the E/R diagram for the Hive metastore.

> **Note**:
> For information about additional configuration that may be needed in a secure cluster, see **Hive Security Configuration**.

Configuring a remote MySQL database for the Hive Metastore

Cloudera recommends you configure a database for the metastore on one or more remote servers (that is, on a host or hosts separate from the HiveServer1 or HiveServer2 process). MySQL is the most popular database to use. Proceed as follows.

Step 1: Install and start MySQL if you have not already done so

To install MySQL on a Red Hat system:

```
$ sudo yum install mysql-server
```

To install MySQL on a SLES system:

```
$ sudo zypper install mysql
$ sudo zypper install libmysqlclient_r15
```

To install MySQL on an Debian/Ubuntu system:

```
$ sudo apt-get install mysql-server
```

After using the command to install MySQL, you may need to respond to prompts to confirm that you do want to complete the installation. After installation completes, start the `mysql` daemon.

On Red Hat systems

```
$ sudo service mysqld start
```

On SLES and Debian/Ubuntu systems

```
$ sudo service mysql start
```

Step 2: Configure the MySQL Service and Connector

Before you can run the Hive metastore with a remote MySQL database, you must configure a connector to the remote MySQL database, set up the initial database schema, and configure the MySQL user account for the Hive user.

To install the MySQL connector on a Red Hat 6 system:

Install mysql-connector-java and symbolically link the file into the `/usr/lib/hive/lib/` directory.

```
$ sudo yum install mysql-connector-java
$ ln -s /usr/share/java/mysql-connector-java.jar /usr/lib/hive/lib/mysql-connector-java.jar
```

To install the MySQL connector on a Red Hat 5 system:

Download the MySQL JDBC connector from **http://www.mysql.com/downloads/connector/j/5.1.html** and copy it to the `/usr/lib/hive/lib/` directory. For example:

```
$ curl -L 'http://www.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.22.tar.gz/from/http://mysql.he.net/' | tar xz
$ sudo cp mysql-connector-java-5.1.22/mysql-connector-java-5.1.22-bin.jar /usr/lib/hive/lib/
```

To install the MySQL connector on a SLES system:

Install `mysql-connector-java` and symbolically link the file into the `/usr/lib/hive/lib/` directory.

```
$ sudo zypper install mysql-connector-java
$ ln -s /usr/share/java/mysql-connector-java.jar /usr/lib/hive/lib/mysql-connector-java.jar
```

To install the MySQL connector on a Debian/Ubuntu system:

Install `mysql-connector-java` and symbolically link the file into the `/usr/lib/hive/lib/` directory.

```
$ sudo apt-get install libmysql-java
$ ln -s /usr/share/java/libmysql-java.jar /usr/lib/hive/lib/libmysql-java.jar
```

Configure MySQL to use a strong password and to start at boot. Note that in the following procedure, your current `root` password is blank. Press the Enter key when you're prompted for the root password.

To set the MySQL root password:

```
$ sudo /usr/bin/mysql_secure_installation
[...]
Enter current password for root (enter for none):
OK, successfully used password, moving on...
[...]
Set root password? [Y/n] y
New password:
Re-enter new password:
Remove anonymous users? [Y/n] Y
[...]
Disallow root login remotely? [Y/n] N
[...]
Remove test database and access to it [Y/n] Y
[...]
Reload privilege tables now? [Y/n] Y
All done!
```

To make sure the MySQL server starts at boot:

- On Red Hat systems:

```
$ sudo /sbin/chkconfig mysqld on
$ sudo /sbin/chkconfig --list mysqld
mysqld          0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

- On SLES systems:

```
$ sudo chkconfig --add mysql
```

- On Debian/Ubuntu systems:

```
$ sudo chkconfig mysql on
```

Step 3. Create the Database and User

The instructions in this section assume you are using **Remote mode**, and that the MySQL database is installed on a separate host from the metastore service, which is running on a host named `metastorehost` in the example.

> **Note**:
> If the metastore service will run on the host where the database is installed, replace `'metastorehost'` in the `CREATE USER` example with `'localhost'`. Similarly, the value of `javax.jdo.option.ConnectionURL` in `/etc/hive/conf/hive-site.xml` (discussed in the **next step**) must be `jdbc:mysql://localhost/metastore`. For more information on adding MySQL users, see **http://dev.mysql.com/doc/refman/5.5/en/adding-users.html**.

Create the initial database schema using the `hive-schema-0.10.0.mysql.sql` file located in the `/usr/lib/hive/scripts/metastore/upgrade/mysql` directory.

Example

```
$ mysql -u root -p
Enter password:
mysql> CREATE DATABASE metastore;
mysql> USE metastore;
mysql> SOURCE /usr/lib/hive/scripts/metastore/upgrade/mysql/hive-schema-0.10.0.mysql.sql;
```

You also need a MySQL user account for Hive to use to access the metastore. It is very important to prevent this user account from creating or altering tables in the metastore database schema.

> **Important**:
> If you fail to restrict the ability of the metastore MySQL user account to create and alter tables, it is possible that users will inadvertently corrupt the metastore schema when they use older or newer versions of Hive.

Example

```
mysql> CREATE USER 'hive'@'metastorehost' IDENTIFIED BY 'mypassword';
...
mysql> REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'hive'@'metastorehost';
mysql> GRANT SELECT,INSERT,UPDATE,DELETE,LOCK TABLES,EXECUTE ON metastore.* TO 'hive'@'metastorehost';
mysql> FLUSH PRIVILEGES;
mysql> quit;
```

Step 4: Configure the Metastore Service to Communicate with the MySQL Database

This step shows the configuration properties you need to set in `hive-site.xml` to configure the metastore service to communicate with the MySQL database, and provides sample settings. Though you can use the same `hive-site.xml` on all hosts (client, metastore, HiveServer), `hive.metastore.uris` is the only property that **must** be configured on all of them; the others are used only on the metastore host.

Given a MySQL database running on `myhost` and the user account `hive` with the password `mypassword`, set the configuration as follows (overwriting any existing values).

> **Note**:
> The `hive.metastore.local` property is no longer supported as of Hive 0.10; setting `hive.metastore.uris` is sufficient to indicate that you are using a remote metastore.

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://myhost/metastore</value>
  <description>the URL of the MySQL database</description>
</property>

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

<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hive</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>mypassword</value>
</property>
```

```
<property>
  <name>datanucleus.autoCreateSchema</name>
  <value>false</value>
</property>

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

<property>
  <name>hive.metastore.uris</name>
  <value>thrift://<n.n.n.n>:9083</value>
  <description>IP address (or fully-qualified domain name) and port of the metastore host</description>
</property>
```

Configuring a remote PostgreSQL database for the Hive Metastore

Before you can run the Hive metastore with a remote PostgreSQL database, you must configure a connector to the remote PostgreSQL database, set up the initial database schema, and configure the PostgreSQL user account for the Hive user.

Step 1: Install and start PostgreSQL if you have not already done so

To install PostgreSQL on a Red Hat system:

```
$ sudo yum install postgresql-server
```

To install PostgreSQL on a SLES system:

```
$ sudo zypper install postgresql-server
```

To install PostgreSQL on an Debian/Ubuntu system:

```
$ sudo apt-get install postgresql
```

After using the command to install PostgreSQL, you may need to respond to prompts to confirm that you do want to complete the installation. In order to finish installation on Red Hat compatible systems, you need to initialize the database. Please note that this operation is not needed on Ubuntu and SLES systems as it's done automatically on first start:

To initialize database files on Red Hat compatible systems

```
$ sudo service postgresql initdb
```

To ensure that your PostgreSQL server will be accessible over the network, you need to do some additional configuration.

First you need to edit the `postgresql.conf` file. Set the `listen` property to `*` to make sure that the PostgreSQL server starts listening on all your network interfaces. Also make sure that the `standard_conforming_strings` property is set to `off`.

You can check that you have the correct values as follows:

On Red-Hat-compatible systems:

```
$ sudo cat /var/lib/pgsql/data/postgresql.conf  | grep -e listen -e standard_conforming_strings
listen_addresses = '*'
standard_conforming_strings = off
```

On SLES systems:

```
$ sudo cat /var/lib/pgsql/data/postgresql.conf  | grep -e listen -e standard_conforming_strings
listen_addresses = '*'
standard_conforming_strings = off
```

On Ubuntu and Debian systems:

```
$ cat /etc/postgresql/9.1/main/postgresql.conf | grep -e listen -e standard_conforming_strings
listen_addresses = '*'
standard_conforming_strings = off
```

You also need to configure authentication for your network in `pg_hba.conf`. You need to make sure that the PostgreSQL user that you will create in the **next step** will have access to the server from a remote host. To do this, add a new line into `pg_hba.con` that has the following information:

```
host    <database>         <user>         <network address>         <mask>               password
```

The following example to allows all users connect from all hosts to all your databases:

```
host    all         all         0.0.0.0         0.0.0.0               password
```

> **Note**:
> This configuration is applicable only for a network listener. Using this configuration won't open all your databases to the entire world; the user must still supply a password to authenticate himself, and privilege restrictions configured in PostgreSQL will still be applied.

After completing the installation and configuration, you can start the database server:

Start PostgreSQL Server

```
$ sudo service postgresql start
```

Use `chkconfig` utility to ensure that your PostgreSQL server will start at a boot time. For example:

```
chkconfig postgresql on
```

You can use the `chkconfig` utility to verify that PostgreSQL server will be started at boot time, for example:

```
chkconfig --list postgresql
```

Step 2: Install the Postgres JDBC Driver

Before you can run the Hive metastore with a remote PostgreSQL database, you must configure a JDBC driver to the remote PostgreSQL database, set up the initial database schema, and configure the PostgreSQL user account for the Hive user.

To install the PostgreSQL JDBC Driver on a Red Hat 6 system:

Install `postgresql-jdbc` package and create symbolic link to the `/usr/lib/hive/lib/` directory. For example:

```
$ sudo yum install postgresql-jdbc
$ ln -s /usr/share/java/postgresql-jdbc.jar /usr/lib/hive/lib/postgresql-jdbc.jar
```

To install the PostgreSQL connector on a Red Hat 5 system:

You need to manually download the PostgreSQL connector from **http://jdbc.postgresql.org/download.html** and move it to the `/usr/lib/hive/lib/` directory. For example:

```
$ wget http://jdbc.postgresql.org/download/postgresql-9.2-1002.jdbc4.jar
$ mv postgresql-9.2-1002.jdbc4.jar /usr/lib/hive/lib/
```

> **Note**:
> You may need to use a different version if you have a different version of Postgres. You can check the version as follows:
>
> ```
> $ sudo rpm -qa | grep postgres
> ```

To install the PostgreSQL JDBC Driver on a SLES system:

Install `postgresql-jdbc` and symbolically link the file into the `/usr/lib/hive/lib/` directory.

```
$ sudo zypper install postgresql-jdbc
$ ln -s /usr/share/java/postgresql-jdbc.jar /usr/lib/hive/lib/postgresql-jdbc.jar
```

To install the PostgreSQL JDBC Driver on a Debian/Ubuntu system:

Install `libpostgresql-jdbc-java` and symbolically link the file into the `/usr/lib/hive/lib/` directory.

```
$ sudo apt-get install libpostgresql-jdbc-java
$ ln -s /usr/share/java/postgresql-jdbc4.jar /usr/lib/hive/lib/postgresql-jdbc4.jar
```

Step 3: Create the metastore database and user account

Proceed as in the following example:

```
bash# sudo –u postgres psql
bash$ psql
postgres=# CREATE USER hiveuser WITH PASSWORD 'mypassword';
postgres=# CREATE DATABASE metastore;
postgres=# \c metastore;
You are now connected to database 'metastore'.
postgres=# \i /usr/lib/hive/scripts/metastore/upgrade/postgres/hive-schema-0.10.0.postgres.sql
SET
```

```
SET
...
```

Now you need to grant permission for all metastore tables to user `hiveuser`. PostgreSQL does not have statements to grant the permissions for all tables at once; you'll need to grant the permissions one table at a time. You could automate the task with the following SQL script:

```
bash# sudo -u postgres psql
metastore=#         \o /tmp/grant-privs
metastore=#         SELECT 'GRANT SELECT,INSERT,UPDATE,DELETE ON "'  || schemaname || '"."' || tablename || '" TO hiveuser ;'
metastore-#           FROM pg_tables
metastore-#           WHERE tableowner = CURRENT_USER and schemaname = 'public';
metastore=#         \o
metastore=#         \i /tmp/grant-privs
```

You can verify the connection from the machine where you'll be running the metastore service as follows:

```
psql -h myhost -U hiveuser -d metastore
metastore=#
```

Step 4: Configure the Metastore Service to Communicate with the PostgreSQL Database

This step shows the configuration properties you need to set in `hive-site.xml` to configure the metastore service to communicate with the PostgreSQL database. Though you can use the same `hive-site.xml` on all hosts (client, metastore, HiveServer), `hive.metastore.uris` is the only property that **must** be configured on all of them; the others are used only on the metastore host.

Given a PostgreSQL database running on host `myhost` under the user account `hive` with the password `mypassword`, you would set configuration properties as follows.

> **Note**:
> - The instructions in this section assume you are using **Remote mode**, and that the PostgreSQL database is installed on a separate host from the metastore server.
> - The `hive.metastore.local` property is no longer supported as of Hive 0.10; setting `hive.metastore.uris` is sufficient to indicate that you are using a remote metastore.

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:postgresql://myhost/metastore</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>org.postgresql.Driver</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hiveuser</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>mypassword</value>
</property>

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

<property>
  <name>hive.metastore.uris</name>
  <value>thrift://<n.n.n.n>:9083</value>
  <description>IP address (or fully-qualified domain name) and port of the metastore host</description>
</property>
```

Step 6: Test connectivity to the metastore:

```
$ hive -e “show tables;”
```

> **Note**:

This will take a while the first time.

Configuring a remote Oracle database for the Hive Metastore

Before you can run the Hive metastore with a remote Oracle database, you must configure a connector to the remote Oracle database, set up the initial database schema, and configure the Oracle user account for the Hive user.

Step 1: Install and start Oracle

The Oracle database is not part of any Linux distribution and must be purchased, downloaded and installed separately. You can use **Express edition** that can be downloaded for free from Oracle website.

Step 2: Install the Oracle JDBC Driver

You must download the Oracle JDBC Driver from the Oracle website and put the file `ojdbc6.jar` into `/usr/lib/hive/lib/` directory. The driver is available for download **here**.

```
$ sudo mv ojdbc6.jar /usr/lib/hive/lib/
```

Step 3: Create the Metastore database and user account

Connect to your Oracle database as an administrator and create the user that will use the Hive metastore.

```
$ sqlplus "sys as sysdba"
SQL> create user hiveuser identified by mypassword;
SQL> grant connect to hiveuser;
SQL> grant all privileges to hiveuser;
```

Connect as the newly created `hiveuser` user and load the initial schema:

```
$ sqlplus hiveuser
SQL> @/usr/lib/hive/scripts/metastore/upgrade/oracle/hive-schema-0.10.0.oracle.sql
```

Connect back as an administrator and remove the power privileges from user `hiveuser`. Then grant limited access to all the tables:

```
$ sqlplus "sys as sysdba"
SQL> revoke all privileges from hiveuser;
SQL> BEGIN
  2     FOR R IN (SELECT owner, table_name FROM all_tables WHERE owner='HIVEUSER') LOOP
  3        EXECUTE IMMEDIATE 'grant  SELECT,INSERT,UPDATE,DELETE on '||R.owner||'.'||R.table_name||' to hiveuser';
  4     END LOOP;
  5  END;
  6
  7  /
```

Step 4: Configure the Metastore Service to Communicate with the Oracle Database

This step shows the configuration properties you need to set in `hive-site.xml` to configure the metastore service to communicate with the Oracle database, and provides sample settings. Though you can use the same `hive-site.xml` on all hosts (client, metastore, HiveServer), `hive.metastore.uris` is the only property that must be configured on all of them; the others are used only on the metastore host.

Example

Given an Oracle database running on `myhost` and the user account `hiveuser` with the password `mypassword`, set the configuration as follows (overwriting any existing values):

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:oracle:thin:@//myhost/xe</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>oracle.jdbc.OracleDriver</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>hiveuser</value>
</property>

<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>mypassword</value>
</property>
```

```
<property>
  <name>datanucleus.autoCreateSchema</name>
  <value>false</value>
</property>

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

<property>
  <name>hive.metastore.uris</name>
  <value>thrift://<n.n.n.n>:9083</value>
  <description>IP address (or fully-qualified domain name) and port of the metastore host</description>
</property>
```

Configuring HiveServer2

You must make the following configuration changes before using HiveServer2. Failure to do so may result in unpredictable behavior.

Table Lock Manager (Required)

You must properly configure and enable Hive's Table Lock Manager. This requires installing ZooKeeper and setting up a ZooKeeper ensemble; see **ZooKeeper Installation**.

> **Important**:
> Failure to do this will prevent HiveServer2 from handling concurrent query requests and may result in data corruption.

Enable the lock manager by setting properties in `/etc/hive/conf/hive-site.xml` as follows (substitute your actual ZooKeeper node names for those in the example):

```
<property>
  <name>hive.support.concurrency</name>
  <description>Enable Hive's Table Lock Manager Service</description>
  <value>true</value>
</property>

<property>
  <name>hive.zookeeper.quorum</name>
  <description>Zookeeper quorum used by Hive's Table Lock Manager</description>
  <value>zk1.myco.com,zk2.myco.com,zk3.myco.com</value>
</property>
```

> **Important**:
> Enabling the Table Lock Manager without specifying a list of valid Zookeeper quorum nodes will result in unpredictable behavior. Make sure that both properties are properly configured.

JDBC driver

The connection URL format and the driver class are different for HiveServer2 and HiveServer1:

HiveServer version	Connection URL	Driver Class
HiveServer2	`jdbc:hive2://<host>:<port>`	`org.apache.hive.jdbc.HiveDriver`
HiveServer1	`jdbc:hive://<host>:<port>`	`org.apache.hadoop.hive.jdbc.HiveDriver`

Authentication

HiveServer2 can be **configured** to authenticate all connections; by default, it allows any client to connect. HiveServer2 supports either **Kerberos** or **LDAP** authentication; configure this in the `hive.server2.authentication` property in the `hive-site.xml` file. You can also configure **pluggable authentication**, which allows you to use a custom authentication provider for HiveServer2; and **impersonation**, which allows users to execute queries and access HDFS files as the connected user rather than the super user who started the HiveServer2 daemon. For more information, see **Hive Security Configuration**.

Configuring HiveServer2 for YARN

To use HiveServer2 with YARN, you must set the `HADOOP_MAPRED_HOME` environment variable: add the following line to `/etc/default/hive-server2`:

```
export HADOOP_MAPRED_HOME=/usr/lib/hadoop-mapreduce
```

Running HiveServer2 and HiveServer Concurrently

Cloudera recommends running HiveServer2 instead of the original HiveServer (HiveServer1) package in most cases; HiveServer1 is included for backward compatibility. Both HiveServer2 and HiveServer1 can be run concurrently on the same system, sharing the same data sets. This allows you to run HiveServer1 to support, for example, Perl or Python scripts that use the native HiveServer1 Thrift bindings.

Both HiveServer2 and HiveServer1 bind to port 10000 by default, so at least one of them must be configured to use a different port. The environment variables used are:

HiveServer version	Specify Port	Specify Bind Address
HiveServer2	HIVE_SERVER2_THRIFT_PORT	HIVE_SERVER2_THRIFT_BIND_HOST
HiveServer1	HIVE_PORT	*<Host bindings cannot be specified>*

Mysql Integration with Hive Installation:

Need 2 things:
1)mysql-installer-community-5.5.52.0.msi
2)mysql-workbench-community-6.3.7-winx64
3)server.
below are url links to download.
Mysql-installer is bundled with all above server,client,workbench.
http://dev.mysql.com/doc/refman/5.6/en/mysql-installer-gui.html

http://dev.mysql.com/downloads/windows/

http://www.mysql.com/products/workbench/
Need

Environment variable:
export USE_DEPRECATED_CLI=false

Create and set 777(RWX) permission for user /usr/hive:
C:\Windows\system32>hdfs dfs -mkdir /user/hive

C:\Windows\system32>hdfs dfs -chmod 777 /user/hive

Create and set 777(RWX) permission for /usr/hive/warehouse:

C:\Windows\system32>hdfs dfs -mkdir /user/hive/warehouse

C:\Windows\system32>hdfs dfs -chmod 777 /user/hive/warehouse

Create and set 777(RWX) permission for /tmp:

C:\Windows\system32>hdfs dfs -mkdir /tmp

C:\Windows\system32>hdfs dfs -chmod 777 /tmp

Create and set 777(RWX) permission for /tmp/hive:

C:\Windows\system32>hdfs dfs -mkdir /tmp/hive

C:\Windows\system32>hdfs dfs -chmod 777 /tmp/hive

Create and set 777(RWX) permission for warehouse:

C:\Windows\system32>hdfs dfs -ls /user/*

Found 1 items

drwxrwxrwx - admin supergroup 0 2016-08-28 11:13 /user/hive/warehouse

C:\Users\admin>hdfs dfs -chmod 777 /user/admin/

Microsoft Windows [Version 6.1.7601]

Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\System32>cd C:\Program Files\MySQL\MySQL Server 5.7\bin\

C:\Program Files\MySQL\MySQL Server 5.7\bin>mysql -u root -p

Enter password: ****

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 124

Server version: 5.7.14-log MySQL Community Server (GPL)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

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>

check server version 5.7.14 and download mysql connector

http://dev.mysql.com/downloads/connector/j/

copy jar mysql-connector-java-5.1.39-bin.jar to C:\apache-hive-2.1.0-bin\lib\mysql-connector-java-5.1.39-bin.jar

for 5.7 it is 5.1.39 in the repository hence it is fine.

Microsoft Windows [Version 6.1.7601]

Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd C:\apache-hive-2.1.0-bin\bin

C:\apache-hive-2.1.0-bin\bin>beeline

Beeline version 2.1.0 by Apache Hive

below command should be from hive-config.xml

!connect jdbc:mysql://127.0.0.1:3306/employee

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://127.0.0.1:3306/employee?ssl=false</value>
  <description>JDBC connect string for a JDBC metastore</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>com.mysql.jdbc.Driver</value>
  <description>Driver class name for a JDBC metastore</description>
</property>
```

To check whether values persist in hadoop:

see metastore default dir of hive.

```
C:\Windows\system32>hdfs dfs -ls /user/hive/warehouse
Found 2 items
drwxrwxrwx   - admin supergroup          0 2016-09-17 18:49 /user/hive/warehouse
/hive_employee.db
drwxrwxrwx   - admin supergroup          0 2016-09-17 18:00 /user/hive/warehouse
/t_hive_employee
```

https://svn.apache.org/repos/asf/hive/tags/release-0.3.0/conf/hive-default.xml

name	value	description
hive.exec.scratchdir	/tmp/hive-${user.name}	Scratch space for Hive jobs
hive.metastore.local	true	controls whether to connect to remove metastore server
javax.jdo.option.ConnectionURL	jdbc:derby:;databaseName=metastore_db;create=true	JDBC connect string for a JDBC metastore
javax.jdo.option.ConnectionDriverName	org.apache.derby.jdbc.EmbeddedDriver	Driver class name for a JDBC metastore
hive.default.fileformat	TextFile	Default file format for CREATE TABLE statement. Options are TextFile and SequenceFile. Users can explicitly say CREATE TABLE ... STORED AS <TEXTFILE|SEQUENCEFILE> to override
hive.metastore.warehouse.dir	/user/hive/warehouse	location of default database for the warehouse

How to Start

```
$HIVE_HOME/bin/hiveserver2
```

OR

```
$HIVE_HOME/bin/hive --service hiveserver2
```

Optional Environment Settings

HIVE_SERVER2_THRIFT_BIND_HOST – Optional TCP host interface to bind to. Overrides the configuration file setting.
HIVE_SERVER2_THRIFT_PORT – Optional TCP port number to listen on, default 10000. Overrides the configuration file setting.

https://cwiki.apache.org/confluence/display/Hive/Setting+Up+HiveServer2

Different ways to connect to mysql metastore

OPTION : using SqlLine format only: it behave like mysql command line interface.

C:\apache-hive-2.1.0-bin\bin>beeline

Beeline version 2.1.0 by Apache Hive

beeline> !connect jdbc:mysql://127.0.0.1:3306/employee

like connecting to jdbc via sqline not hive see it uses Driver: MySQL Connector Java .

Connecting to jdbc:mysql://127.0.0.1:3306/employee

Enter username for jdbc:mysql://127.0.0.1:3306/employee: bigdata

Enter password for jdbc:mysql://127.0.0.1:3306/employee: *******

Sat Sep 17 18:53:45 IST 2016 WARN: Establishing SSL connection without server's

identity verification is not recommended. According to MySQL 5.5.45+, 5.6.26+ and 5.7.6+ requirements SSL connection must be established by default if explicitoption isn't set. For compliance with existing applications not using SSL the verifyServerCertificate property is set to 'false'. You need either to explicitly disable SSL by setting useSSL=false, or set useSSL=true and provide truststore for server certificate verification.

Connected to: MySQL (version 5.7.14-log)

Driver: MySQL Connector Java (version mysql-connector-java-5.1.39 (Revision: 32

89a357af6d09ecc1a10fd3c26e95183e5790ad))

Transaction isolation: TRANSACTION_REPEATABLE_READ

SLF4J: Class path contains multiple SLF4J bindings.

SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/jdbc/hive-jdbc-2.1.0

-standalone.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl

-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]

SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j

-log4j12-1.7.5.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]

ERROR StatusLogger No log4j2 configuration file found. Using default configurati

on: logging only errors to the console.

0: jdbc:mysql://127.0.0.1:3306/employee>

0: jdbc:mysql://127.0.0.1:3306/employee> select * from t_employee;

```
+--------+----------+------------------+--+
| empId  |   name   |     address      |
+--------+----------+------------------+--+
| 2      | Kannan   | Sozhia           |
```

```
| 1      | Sathish  | Thousand Lights  |
| 3      | Selvam   | velachery        |
| 4      | Raja     | Thousand Besant  |
| 5      | Raghu    | Guindy           |
+--------+----------+------------------+--+
5 rows selected (0.017 seconds)
0: jdbc:mysql://127.0.0.1:3306/employee>
```

Beeline SqlLine format AND MYSQL TABLE -both are same

sqlline format syntax and mysql table is same, insert in sqlline reflected in db and viceversa.

Insertion in sqlline reflected in mysql db:

```
0: jdbc:mysql://127.0.0.1:3306/employee> insert into t_employee values(6,'test',
'test');
1 row affected (0.083 seconds)
0: jdbc:mysql://127.0.0.1:3306/employee>
```

After a while db restart it shows.

```
mysql> select * from employee.t_employee;
+-------+----------------+-----------------+
| empId | name           | address         |
+-------+----------------+-----------------+
|     2 | Kannan         | Sozhia          |
|     1 | Sathish        | Thousand Lights |
|     3 | Selvam         | velachery       |
|     4 | Raja           | Thousand Besant |
|     5 | Raghu          | Guindy          |
|     7 | mysqlworkbench | workbenchtest   |
|     6 | test           | test            |
|     8 | Test hive      | hive            |
+-------+----------------+-----------------+
8 rows in set (0.00 sec)
```

Insertion in db reflected in sqlline:

```
mysql> insert into employee.t_employee values(9,'TestInserFromworkBench','Test');
Query OK, 1 row affected (0.03 sec)
mysql>
```

```
0: jdbc:mysql://127.0.0.1:3306/employee> select * from employee.t_employee order
 by empId;
+--------+----------------+-----------------+--+
| empId  |     name       |    address      |
+--------+----------------+-----------------+--+
| 1      | Sathish        | Thousand Lights |
| 2      | Kannan         | Sozhia          |
| 3      | Selvam         | velachery       |
| 4      | Raja           | Thousand Besant |
| 5      | Raghu          | Guindy          |
| 6      | test           | test            |
| 7      | mysqlworkbench | workbenchtest   |
+--------+----------------+-----------------+--+
7 rows selected (0.01 seconds)
```

inserted value through mysql and reflected in sqline format syntax once only after open in new session.i.e restart

```
C:\Windows\System32>!connect jdbc:mysql://127.0.0.1:3306/employee
0: jdbc:mysql://127.0.0.1:3306/employee> select * from employee.t_employee;
+--------+-----------------------+-----------------+--+
| empId  |        name           |    address      |
+--------+-----------------------+-----------------+--+
| 2      | Kannan                | Sozhia          |
| 1      | Sathish               | Thousand Lights |
| 3      | Selvam                | velachery       |
| 4      | Raja                  | Thousand Besant |
| 5      | Raghu                 | Guindy          |
| 7      | mysqlworkbench        | workbenchtest   |
| 6      | test                  | test            |
| 8      | Test hive             | hive            |
| 9      | TestInserFromworkBench | Test            |
+--------+-----------------------+-----------------+--+
9 rows selected (0.08 seconds)
```

OPTION 1: hive with Hiveserver1(Embedded Mode)

If derby db is used insted of mysql it is **(Embedded Mode)**

OPTION 2: hive with Hiveserver1(Local Mode)

it shows only meta store tables in hive it wont show t_employee table because t_employee is table in mysql table not metastore.

```
C:\Windows\system32>hive
hive> show tables;
OK
hive_external_employee
t_hive_employee
t_hive_employee1
t_hive_employee3
t_hive_employee5
5 rows selected (0.08 seconds)
hive>
```

OPTION 3: beeline thriftserver with Hiveserver2(Remote Mode)

```
C:\Windows\system32>hive --service hiveserver2
```

IMPORTANT:since it works on remote mode it shows Remote Exception if not able to connect.

```
C:\apache-hive-2.1.0-bin\bin>beeline
Beeline version 2.1.0 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000/employee
Connecting to jdbc:hive2://localhost:10000/employee
Enter username for jdbc:hive2://localhost:10000/employee: bigdata
Enter password for jdbc:hive2://localhost:10000/employee: *******
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/jdbc/hive-jdbc-2.1.0
-standalone.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/hive-jdbc-2.1.0-
standalone.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]
ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
Error: Failed to open new session: java.lang.RuntimeException: org.apache.hadoop
.ipc.RemoteException(org.apache.hadoop.security.authorize.AuthorizationException
): User: admin is not allowed to impersonate bigdata (state=,code=0)
```

TO solve this issue:

Environment Variable	value
HADOOP_CONF_DIR	C:\hadoop-2.3.0\etc
HADOOP_HOME	C:\hadoop-2.3.0
HADOOP_HOME_DIR	C:\hadoop-2.3.0
HADOOP_USER_CLASSPATH_FIRST	TRUE
HIVE_HOME	C:\apache-hive-2.1.0-bin
HIVE_SERVER2_THRIFT_BIND_HOST	10000
HIVE_SERVER2_THRIFT_PORT	10000
Platform	X64
SPARK_HOME	C:\spark-1.6.1-bin-hadoop2.3
USE_DEPRECATED_CLI	FALSE

Path:

C:\Python27\;C:\Python27\Scripts;%SystemRoot%;C:\Windows\SysWOW64;C:\apache-maven-3.3.9\bin;%JAVA_HOME%\bin;C:\protoc32;c:\Progra~1\Microsoft SDKs\Windows\v7.1\Bin;C:\Program Files (x86)\Git\bin;C:\Program Files (x86)\Skype\Phone\;C:\hadoop-2.3.0\bin;C:\hadoop-2.3.0\sbin;C:\spark-1.6.1-bin-hadoop2.3\bin;C:\spark-1.6.1-bin-hadoop2.3\sbin;C:\Progra~2\scala\bin;C:\Progra~1\R\R-3.2.3\bin;C:\apache-hive-2.1.0-bin\bin;C:\db-derby-10.10.2.0-bin\bin;C:\kafka_2.11-0.9.0.0\bin\windows;C:\zookeeper-3.3.6\bin;C:\Mongo 2.6\bin;C:\Program Files\nodejs\;C:\Program Files\nodejs\node_modules\npm\bin;C:\Program Files\Apache Software Foundation\apache-tomcat-8.5.4\bin;C:\Program Files\MySQL\MySQL Utilities 1.6\;C:\apache-hive-2.1.0-bin\bin

```
public static void main(String[] args) throws SQLException {
            try {
              Class.forName("org.apache.hive.jdbc.HiveDriver");//hive2 thrift server
              // Class.forName("org.apache.hadoop.hive.jdbc.HiveDriver");//hive1
            } catch (ClassNotFoundException e) {
                  e.printStackTrace();
            }
            Connection con = null;
            try {
                  con =
DriverManager.getConnection("jdbc:hive://localhost:10000/employee", "bigdata",
"bigdata");
                  System.out.println("connected.............");
            } catch (Exception e) {
                  System.err.println("Error.............");
                  e.printStackTrace();
            }
}
```

IMPORTANT

only running the below command open up port 10000 .hence to start hiveserver2 below command should be given instead of C:\Windows\system32>hive alone.

C:\Windows\system32>hive --service hiveserver2

ERROR:

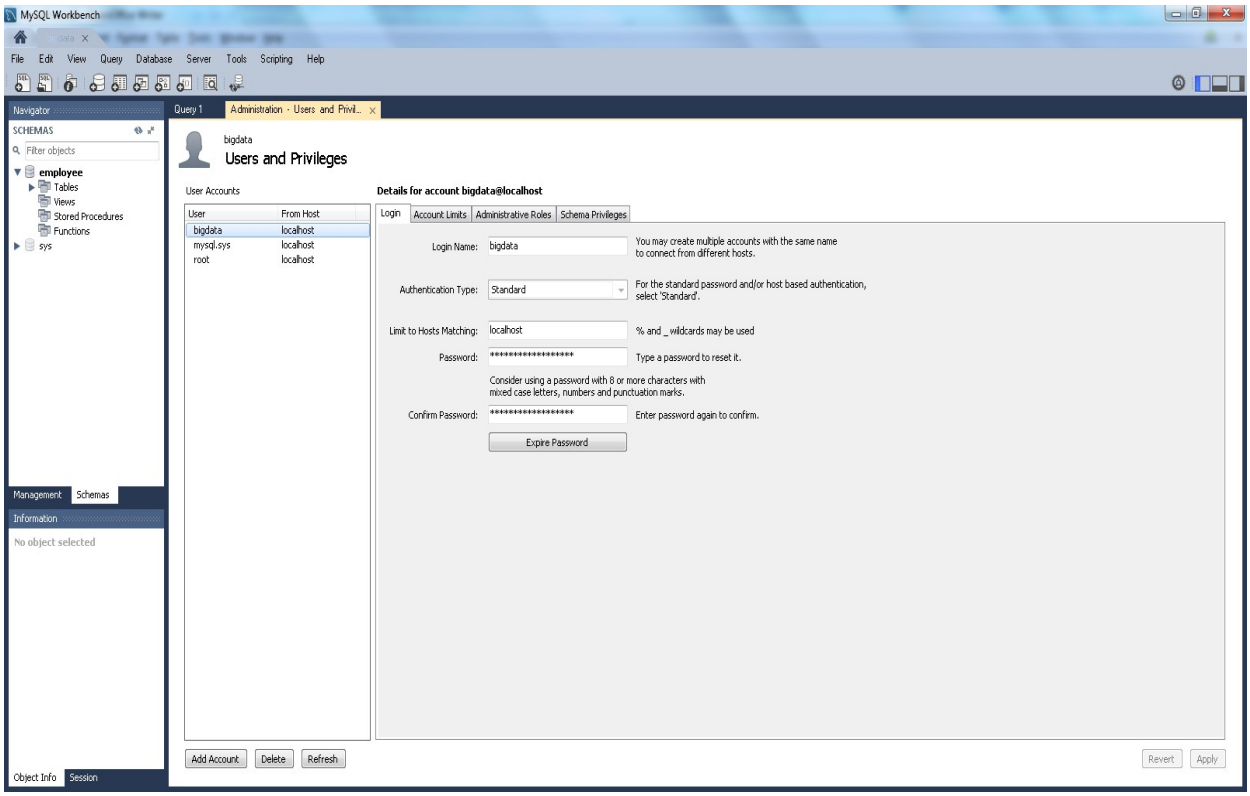
Error: Failed to open new session: java.lang.RuntimeException: org.apache.hadoop .ipc.**RemoteException**(org.apache.hadoop.security.authorize.AuthorizationException): User: admin is not allowed to impersonate admin (state=,code=0)

OPTION 2: beeline> !connect jdbc:hive2://localhost:10000/employee

Thrift server – Actual hive

edit C:\hadoop-2.3.0\etc\hadoop\core-site.xml

```
<configuration>
  <property>
     <name>fs.defaultFS</name>
     <value>hdfs://localhost:9000</value>
  </property>
<!-- not mandatory-->
  <property>
            <name>hadoop.proxyuser.bigdata.groups</name> //bigdata is user name  which is created in mysql
            <value>bigdata</value>
  </property>
 <property>
            <name>hadoop.proxyuser.bigdata.hosts</name> //bigdata is user name
            <value>bigdata</value>
  </property>
<!-- not mandatory-->
</configuration>
```



Reason for issue:

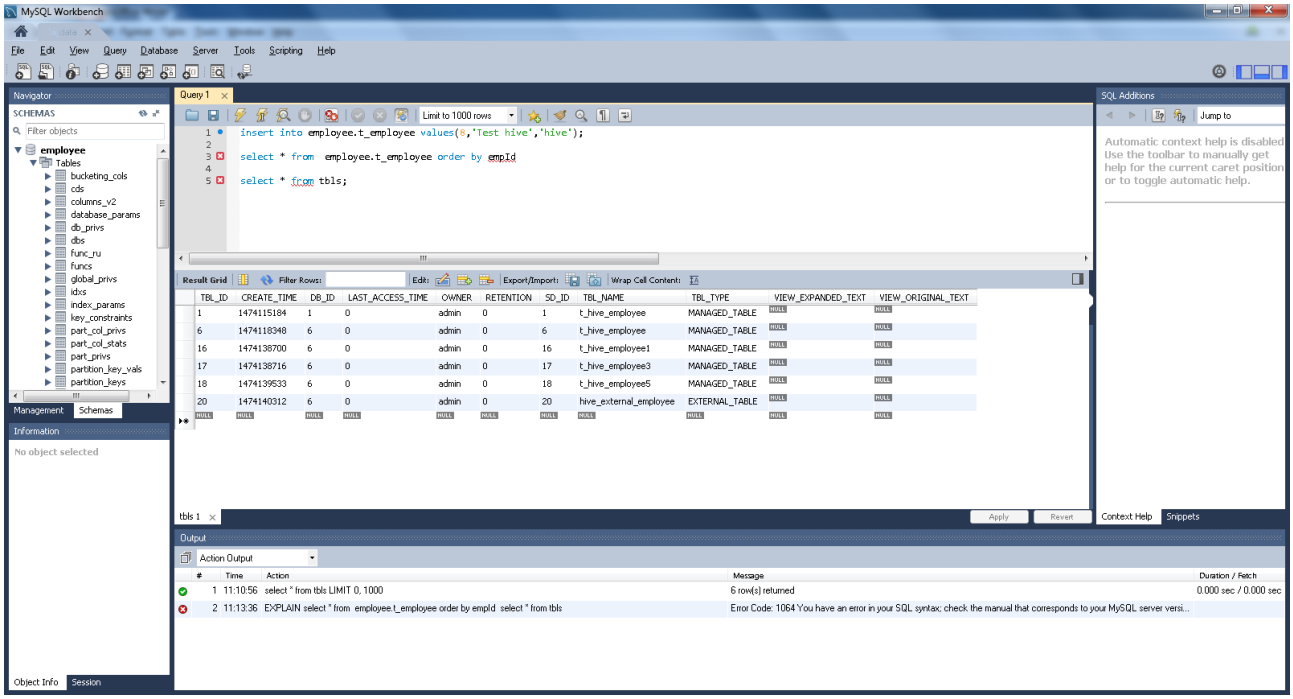
All the table store created in mysql is in **admin** owner hence it shows error when trying to connect with **bigdata** user.

Error shown in bigdata namenode server:

16/09/18 12:00:53 INFO ipc.Server: Connection from 127.0.0.1:2034 for protocol o

rg.apache.hadoop.hdfs.protocol.ClientProtocol is unauthorized for user bigdata (

auth:PROXY) via admin (auth:SIMPLE)**16/09/18 12:00:53 INFO ipc.Server: Socket Reader #1 for port 9000: readAndProcess from client 127.0.0.1** threw exception [org.apache.hadoop.security.authorize.AuthorizationException: User: admin is not allowed to impersonate bigdata]



Without connecting to thrift server hive wont allow to create role via thrift server

beeline> create role bigdata;

No current connection

IMPORTANT:

```
 <property>
    <name>hive.server2.enable.doAs</name>
    <value>false</value>
    <description>Set this property to enable impersonation in Hive Server 2</description>
  </property>
<property>
  <name>hive.server2.enable.impersonation</name>
  <description>Enable user impersonation for HiveServer2</description>
  <value>true</value>
</property>
```

works !!! finally need to set to false to allow impersonation in hive server2 .once you added the proxy configs in core-site.xml, e.g hadoop.proxyuser.hdfs.groups, where hdfs is user who started hiveserver, then add hive.server2.enable.doAs=false to impersonate other user/groups

hive2 impersonate as bigdata:

```
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>cd C:\apache-hive-2.1.0-bin\bin

C:\apache-hive-2.1.0-bin\bin>beeline
Beeline version 2.1.0 by Apache Hive
beeline> !connect jdbc:hive2://localhost:10000/employee
Connecting to jdbc:hive2://localhost:10000/employee
Enter username for jdbc:hive2://localhost:10000/employee: bigdata
Enter password for jdbc:hive2://localhost:10000/employee: *******
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/jdbc/hive-jdbc-2.1.0
-standalone.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/hive-jdbc-2.1.0-
standalone.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/apache-hive-2.1.0-bin/lib/log4j-slf4j-impl
-2.4.1.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/C:/hadoop-2.3.0/share/hadoop/common/lib/slf4j
-log4j12-1.7.5.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]
ERROR StatusLogger No log4j2 configuration file found. Using default configurati
on: logging only errors to the console.
Connected to: Apache Hive (version 2.1.0)
Driver: Hive JDBC (version 2.1.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://localhost:10000/employee>
```

Result from hiveserver 2 connection:

```
beeline> !connect jdbc:hive2://localhost:10000/employee
Connecting to jdbc:hive2://localhost:10000/employee
Enter username for jdbc:hive2://localhost:10000/employee: bigdata
Enter password for jdbc:hive2://localhost:10000/employee: *******
Connected to: Apache Hive (version 2.1.0)
Driver: Hive JDBC (version 2.1.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://localhost:10000/employee> select * from t_hive_employee;
```

```
Error: Error while compiling statement: FAILED: SemanticException [Error 10001]:
 Line 1:14 Table not found 't_hive_employee' (state=42S02,code=10001)
0: jdbc:hive2://localhost:10000/employee> show tables;
Error: Error while compiling statement: FAILED: SemanticException [Error 10072]:
 Database does not exist: employee (state=42000,code=10072)
0: jdbc:hive2://localhost:10000/employee> use employee;
Error: Error while compiling statement: FAILED: SemanticException [Error 10072]:
 Database does not exist: employee (state=42000,code=10072)
0: jdbc:hive2://localhost:10000/employee> show databases;
+----------------+--+
| database_name  |
+----------------+--+
| default        |
| hive_employee  |
+----------------+--+
2 rows selected (1.321 seconds)
0: jdbc:hive2://localhost:10000/employee> use hive_employee;
No rows affected (0.146 seconds)
0: jdbc:hive2://localhost:10000/employee> show tables;
+--------------------------+--+
|        tab_name          |
+--------------------------+--+
| hive_external_employee   |
| t_hive_employee          |
| t_hive_employee1         |
| t_hive_employee3         |
| t_hive_employee5         |
+--------------------------+--+
5 rows selected (0.271 seconds)
0: jdbc:hive2://localhost:10000/employee> select * from t_hive_employee;
+------------------------+-----------------------+---------------------------+--+

| t_hive_employee.empid  | t_hive_employee.name  | t_hive_employee.address   |
+------------------------+-----------------------+---------------------------+--+

| 1                      | Satish                | santhome                  |
| 2                      | Kannan                | sozhianullar              |
| 3                      | Raghu                 | guindy                    |
| 4                      | selvam                | velachery                 |
+------------------------+-----------------------+---------------------------+--+
```

```
4 rows selected (2.629 seconds)
0: jdbc:hive2://localhost:10000/employee>
```

sample jdbc hive:

```
public static void main(String[] args) throws SQLException {
            try {
                  Class.forName("org.apache.hive.jdbc.HiveDriver");
                  // Class.forName("org.apache.hadoop.hive.jdbc.HiveDriver");
            } catch (ClassNotFoundException e) {
                  e.printStackTrace();
            }
            Connection con = null;
            try {
                  con =
DriverManager.getConnection("jdbc:hive2://localhost:10000/employee", "bigdata",
"bigdata");
                  System.out.println("connected.............");
            } catch (Exception e) {
                  System.err.println("Error.............");
                  e.printStackTrace();
            }

          Statement stmt = con.createStatement();
          // select * query
          String sql = "select * from hive_employee.t_hive_employee"  ;
          System.out.println("=============");
          while (res.next()) {
            int empId = res.getInt("empId");
            String name = res.getString("name");
            String address = res.getString("address");
            System.out.println("empId:::"+empId +"\t Name:::"+name +"\t
address:::"+address );
          }
          System.out.println("=============");
      }
```

OUTPUT:

```
connected.............
Running: select * from hive_employee.t_hive_employee
=============
empId:::1     Name:::Satish      address:::santhome
empId:::2     Name:::Kannan      address:::sozhianullar
empId:::3     Name:::Raghu       address:::guindy
empId:::4     Name:::selvam      address:::velachery
=============
```

complete hive-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://127.0.0.1:3306/employee?ssl=false</value>
  <description>JDBC connect string for a JDBC metastore</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionDriverName</name>
  <value>com.mysql.jdbc.Driver</value>
  <description>Driver class name for a JDBC metastore</description>
</property>

<property>
     <name>javax.jdo.option.ConnectionUserName</name>
     <value>bigdata</value>
     <description>Username to use against metastore database</description>
    </property>

   <property>
     <name>javax.jdo.option.ConnectionPassword</name>
     <value>bigdata</value>
     <description>password to use against metastore database</description>
  </property>
  <property>
  <name>hive.server2.enable.impersonation</name>
  <description>Enable user impersonation for HiveServer2</description>
  <value>true</value>
</property>
  <property>
  <name>hive.server2.authentication</name>
  <value>NONE</value>
  <description>
    Client authentication types.
      NONE: no authentication check
      LDAP: LDAP/AD based authentication
      KERBEROS: Kerberos/GSSAPI authentication
      CUSTOM: Custom authentication provider
             (Use with property hive.server2.custom.authentication.class)
  </description>
</property>
<property>
 <name>datanucleus.autoCreateTables</name>
 <value>True</value>
```

```
</property>
<!--not mandatory-->
<property>
   <name>hive.server2.thrift.port</name>
   <value>10000</value>
   <description>Port number of HiveServer2 Thrift interface when hive.server2.transport.mode is 'binary'.</description>
  </property>
  <property>
     <name>hive.server2.thrift.http.port</name>
     <value>10001</value>
     <description>Port number of HiveServer2 Thrift interface when hive.server2.transport.mode is 'http'.</description>
   </property>
<!--not mandatory-->
   <property>
     <name>hive.server2.thrift.http.path</name>
     <value>cliservice</value>
     <description>Path component of URL endpoint when in HTTP mode.</description>
  </property>
<!-- to impersonate other user/groups -->
  <property>
   <name>hive.server2.enable.doAs</name>
   <value>false</value>
   <description>Set this property to enable impersonation in Hive Server 2</description>
  </property>
  <property>
   <name>hive.metastore.execute.setugi</name>
   <value>true</value>
   <description>Set this property to enable Hive Metastore service impersonation in unsecure mode. In unsecure mode, setting this property to true will cause the metastore to execute DFS operations using the client's reported user and group permissions. Note that this property must be set on both the client and server sides. If the client sets it to true and the server sets it to false, the client setting will be ignored.</description>
</property>
<property>
  <name>hive.security.authorization.enabled</name>
  <value>false</value>
  <description>enable or disable the hive client authorization</description>
</property>
<!--once schema created set to false i.e after first time schema created during 2 nd time run change to false-->
<property>
  <name>datanucleus.autoCreateSchema</name>
  <value>true</value>
</property>
<property>
  <name>hive.security.authorization.createtable.owner.grants</name>
  <value>ALL</value>
  <description>the privileges automatically granted to the owner whenever a table gets created.
   An example like "select,drop" will grant select and drop privilege to the owner of the table</description>
</property>


</configuration>
```

HIVE METASTORE and MYSQL DB

Hadoop ---> Hive --> Mysql(metastore)

Hive Db and tables:

```
hive> show databases;
OK
default
hive_employee
2 rows selected (0.06 seconds)
hive> use hive_employee;
OK
No rows affected (0.05 seconds)
hive> create table t_hive_employee3(empId int, name varchar(30),address varchar(
30)) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '|' LINES TERMINATED BY '\n';
OK
No rows affected (0.29 seconds)
hive>  create External table hive_External_employee(empId int, name varchar(30),a
ddress varchar(30)) ROW FORMAT DELIMITED  FIELDS TERMINATED BY '|' LINES TERMINA
TED BY '\n';
OK
No rows affected (0.24 seconds)
hive>
OK
No rows affected (0.27 seconds)
hive> show tables;
OK
hive_external_employee
t_hive_employee
t_hive_employee1
t_hive_employee3
t_hive_employee5
3 rows selected (0.071 seconds)
hive> use hive_employee;
OK
No rows affected (0.094 seconds)
hive> select * from t_hive_employee;
OK
1 Satish santhome
2 Kannan sozhianullar
3 Raghu guindy
4 selvam velachery
4 rows selected (0.21 seconds)
hive>
```

Mysql DB:

mysql> use employee;

Database changed

mysql> select * from tbls; // **Table store it wont show mysql created tables;**

```
+--------+-------------+-------+------------------+-------+-----------+-------+-
----------------+---------------+--------------------+--------------------+
| TBL_ID | CREATE_TIME | DB_ID | LAST_ACCESS_TIME | OWNER | RETENTION | SD_ID |
TBL_NAME        | TBL_TYPE      | VIEW_EXPANDED_TEXT | VIEW_ORIGINAL_TEXT |
+--------+-------------+-------+------------------+-------+-----------+-------+-
----------------+---------------+--------------------+--------------------+
|      1 |  1474115184 |     1 |                0 | admin |         0 |     1 |
t_hive_employee | MANAGED_TABLE | NULL               | NULL               |
|      6 |  1474118348 |     6 |                0 | admin |         0 |     6 |

t_hive_employee1 | MANAGED_TABLE | NULL               | NULL               |
|     17 |  1474138716 |     6 |                0 | admin |         0 |    17 |
t_hive_employee3 | MANAGED_TABLE | NULL               | NULL               |
hive_external_employee | EXTERNAL_TABLE | NULL               | NULL
+--------+-------------+-------+------------------+-------+-----------+-------+-
----------------+---------------+--------------------+--------------------+
2 rows in set (0.03 sec)
```

NOTE:

mysql> select * from employee.t_hive_employee3;

ERROR 1146 (42S02): Table 'employee.t_hive_employee3' doesn't exist

mysql> select * from t_hive_employee3;

ERROR 1146 (42S02): Table 'employee.t_hive_employee3' doesn't exist

mysql> select * from hive_external_employee;

ERROR 1146 (42S02): Table 'employee.hive_external_employee' doesn't exist

mysql>

After inserting value in HIVE:

hive> load data LOCAL INPATH 'c://HIVE/tables.txt' into table t_hive_employee3;

Loading data to table hive_employee.t_hive_employee3

OK

No rows affected (0.917 seconds)

hive>

```
see  after creating  and inserting table still it wont exist with values in mysql db it
just shows as managed_table or external table.
Mysql is only metastore only if table created in same name as hive metastore it gets
updates from mysql if updated backend in mysql . Updates wont move from hive to mysql.
```

NOTE:See above beeline cmd line with mysql also not showing tables t_hive_employee1,t_hive_employee2, hive_external_employee created in hive only it shows t_employee which is created in mysql database . Also all hive tables are shown as Managed_table or external tables in hive that too it shows only in (select * from tbls)

```
It shows same tables as shown in mysql db . It is same as mysql db but uses  java jbdc
to connect instead of  native odbc as in mysql db
0: jdbc:mysql://127.0.0.1:3306/employee> show tables;
```

```
+----------------------------+--+
|    Tables_in_employee    |
+----------------------------+--+
|| t_employee              |
| tab_col_stats           |
| table_params            |
|| tbls                    |
| version                 |
+----------------------------+--+
39 rows selected (0.04 seconds)
0: jdbc:mysql://127.0.0.1:3306/employee>
```

NOTE:

since we use **employee** as db to connect to mysql it creates **metastore in mysql inside employee mysql db**.

```
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://127.0.0.1:3306/employee</value>
  <description>JDBC connect string for a JDBC metastore</description>
</property>
```

```
C:\Windows\system32>hdfs dfs -ls /user/hive/warehouse
Found 2 items
drwxrwxrwx   - admin supergroup          0 2016-09-17 18:49 /user/hive/warehouse
/hive_employee.db
drwxrwxrwx   - admin supergroup          0 2016-09-17 18:00 /user/hive/warehouse
/t_hive_employee
```

Jars Needed

Only these 2 jars is needed to connect to Hive via JDBC.

hadoop-core-1.2.1.jar

hive-jdbc-2.1.0-standalone.jar