Install Apache Hive 4.0 on Windows 10 and Configure 3 Metastore Modes

Author: Sri Adilakshmi M

Table of Contents

This document outlines the steps needed to install Apache Hive on Windows Operating system.

1. Overview:

Apache Hive was developed by Facebook and became an open-source ETL and data warehousing tool which is built on top of Hadoop for analyzing, querying and managing large datasets stored in HDFS. Hive uses **HQL** (Hive Query Language) as a processing engine that processes HDFS datasets such that queries executed from Hive are internally converted into MapReduce tasks for parallel computation and distribution of data.

The key components of Apache Hive include `Hive CLI`, `Beeline CLI`, `HiveServer2`, `Hive Web Interface`, `Hive Driver`, `Hive Metastore`, `HCatalog` and `WebHCat`.

Hive Metastore is a critical component of Hive because it is the central schema repository that stores Hive metadata including tables, columns, datatypes, data locations etc. created by Hive and this schema repository can be used by other data processing tools such as Spark, Pig etc.

Hive Metastore works in three different modes:

1. **Embedded Metastore**: In this mode, `Hive Metastore` service runs in the same JVM where `Hive Driver` service runs and it uses **Apache Derby** as metastore database that is stored on the local file system. This is the default metastore that comes with Hive installation and is used for testing purposes only. Only one embedded Derby database can access database files at any time so only one Hive session can be opened and if we try to start second Hive session, it errors out. To allow multiple Hive sessions, we can configure Derby to run as Network Server.

2. **Local Metastore**: In this mode, `Hive Metastore` and `Hive Driver` still run within the same JVM process but metastore service connects to a JDBC supported database such as MySQL that runs on a different JVM in the same machine or on different machine. Local metastore currently supports **Derby, MySQL, MSSQL, Oracle** and **Postgres** database systems only.

3. **Remote Metastore:** In this mode, `Hive Metastore` service runs in a different JVM but not in `Hive Driver` service JVM and metastore service connects to a remote database which could be MySQL, MSSQL, Oracle or Postgres. In Remote Metastore, Hive Client will make a connection to Hive Metastore using Thrift protocol, and Metastore server in turn communicates with the database and run queries.

This document provides instructions to install Hive 4.x version on top of Hadoop 3.x.
Note that Hive CLI has been deprecated in Hive 4.x.

2. Prerequisites:

The following prerequisites need to be installed before running Hive.

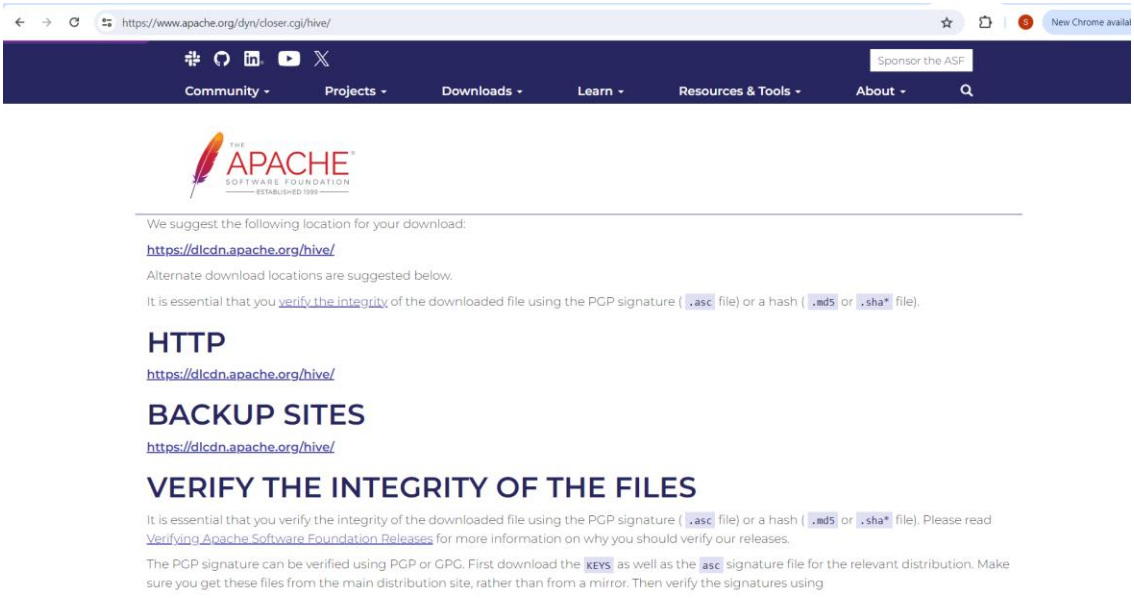
1. **Hadoop:** Before installing Hive, Hadoop cluster must have been installed and running. Go through these steps to install Hadoop on Windows operating system.

2. **File Archiver:** Any file archiver such as **7zip** or **WinRAR** is needed to unzip the downloaded Hive binaries. 7zip can be downloaded from the 7zip Downloads website and WinRAR can be downloaded from the RAR lab Downloads website.

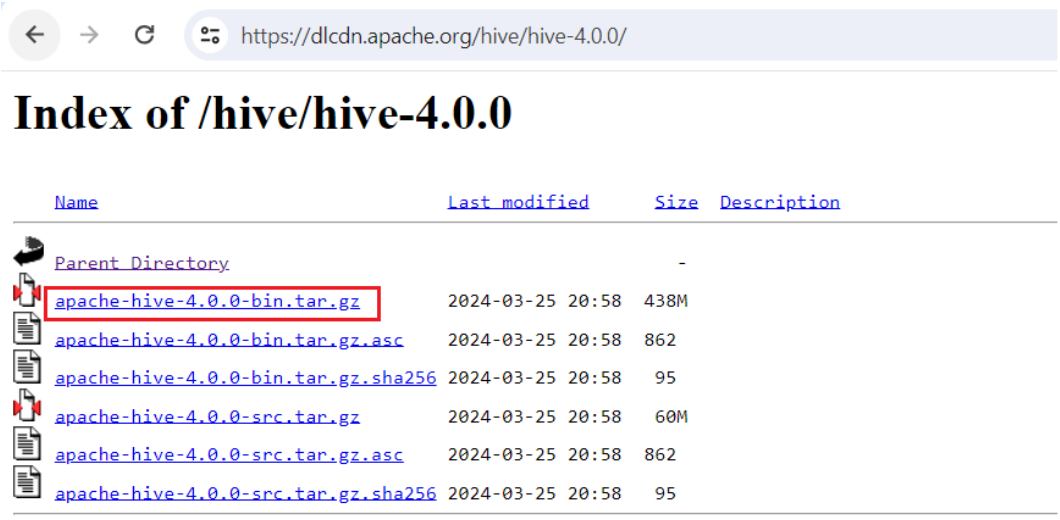
3. **Cygwin:** Since some Hive utilities are not compatible with Windows, we will need the Cygwin tool to run some Linux commands. You can go through these steps to install Cygwin.

3. Download Hive Binaries:

After installing prerequisites, download the latest Hive 4.0.0 release from the Apache Hive Downloads mirror website.

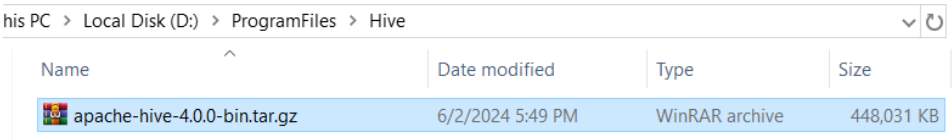


Go to the suggested location for download and click on hive-4.0.0/ from where you need to download the binary file named `apache-hive-4.0.0-bin.tar.gz` file which gets downloaded to your **Downloads** folder.

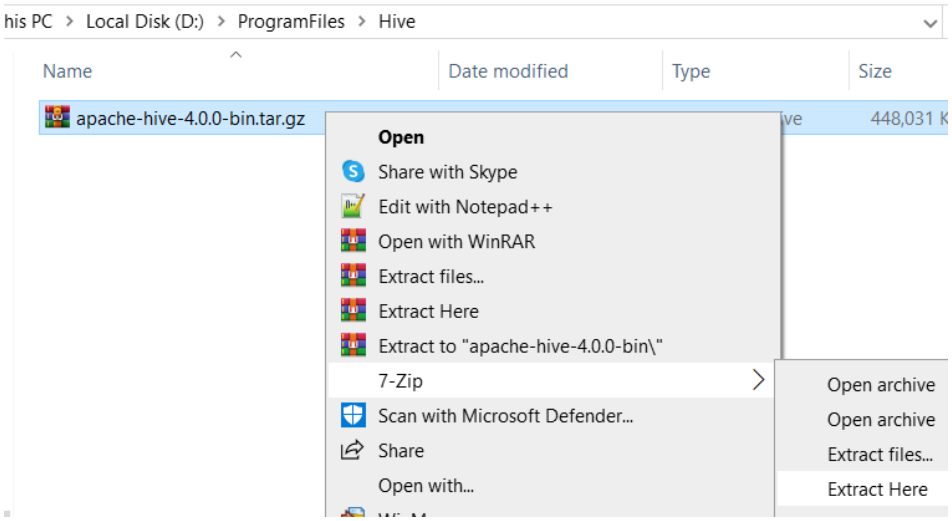


After the binary file is downloaded, unpack it using any file archiver (7zip or WinRAR) utility as below:

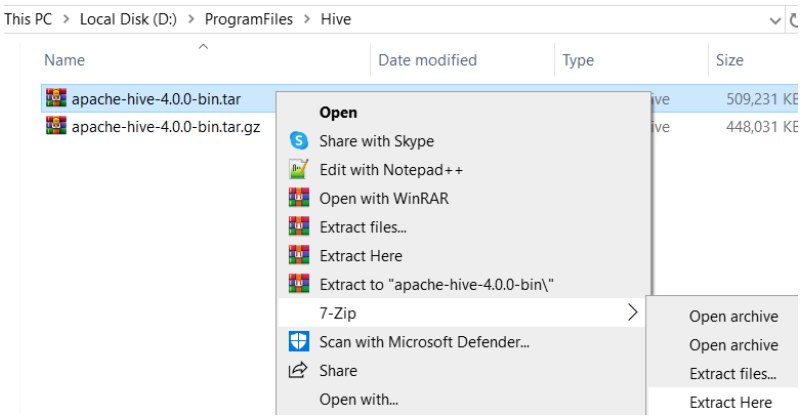
- Choose the installation directory in your machine and copy `apache-hive-4.0.0-bin.tar.gz` file to that directory. Here, we are choosing Hive installation directory as `D:\ProgramFiles\Hive`.



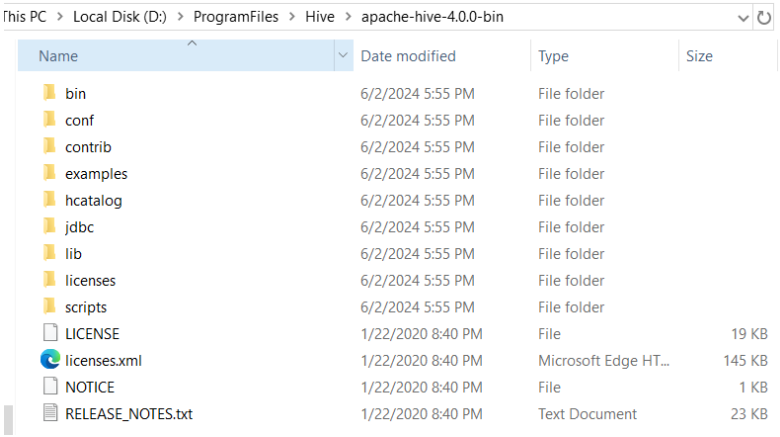
- Right click on the file and choose **7-Zip -> Extract Here** option which extracts a new packed file `apache-hive-4.0.0-bin.tar`.



- Next, unpack `apache-hive-4.0.0-bin.tar` file using 7zip utility.



- The tar file extraction may take few minutes to finish. After finishing, you see a folder named `apache-hive-4.0.0-bin` which consists of Hive binaries and libraries.

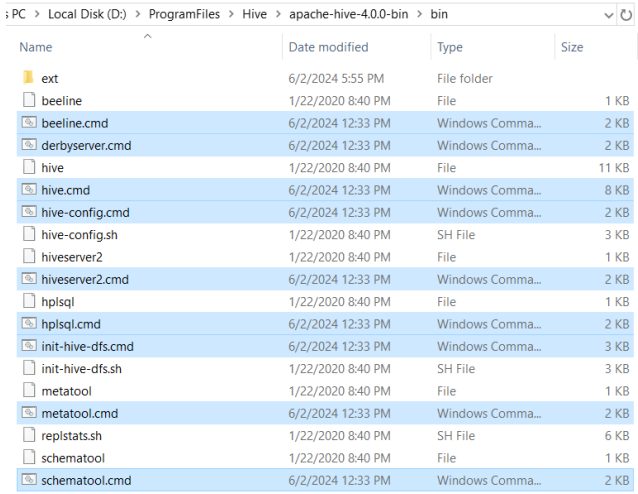


Note:

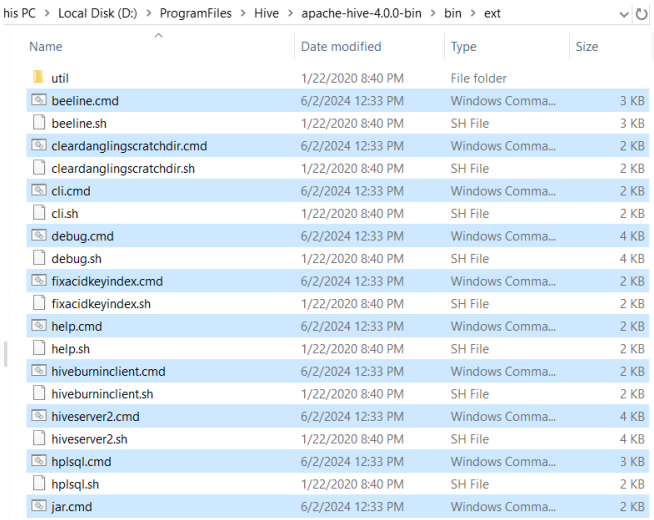
By default, Apache Hive is built to run on Linux Operating system. To make it running on Windows OS, we should use **Cygwin** utility to execute Linux commands from Windows. However, we can run some Hive utilities directly on Windows without Cygwin by downloading `.cmd` files from the HadiFadl GitHub repository. These `.cmd` files are sufficient to start Hive from Windows command line but you will not be able to run some utilities such as `schematool`, `metastore`, etc. directly in which case Cygwin is required.

Alternatively, I would suggest you to download `.cmd` files from my GitHub repository for the corresponding Hive version. For this installation, download files from hive-4.0.0 and paste under the exact folder structure where ever Hive is installed. This makes you to start Hive as well as run other utilities including `schematool`, `metastore`, `metatool`, etc. from Windows itself.

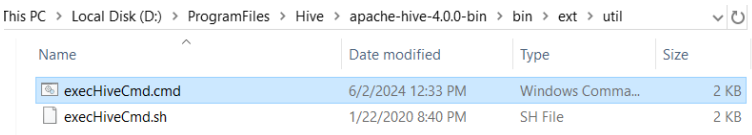
- Copy `.cmd` files in hive-4.0.0/bin folder to `D:\ProgramFiles\Hive\apache-hive-4.0.0-bin\bin` folder.



- Copy `.cmd` files in hive-4.0.0/bin/ext folder to `D:\ProgramFiles\Hive\apache-hive-4.0.0-bin\bin\ext` folder.



- Copy `.cmd` files in hive-4.0.0/bin/ext/util folder to `D:\ProgramFiles\Hive\apache-hive-4.0.0-bin\bin\ext` folder.



4. Set up Environment Variables:

After installing Hadoop prerequisite and Hive binaries, we should configure two environment variables defining Hive installation path.

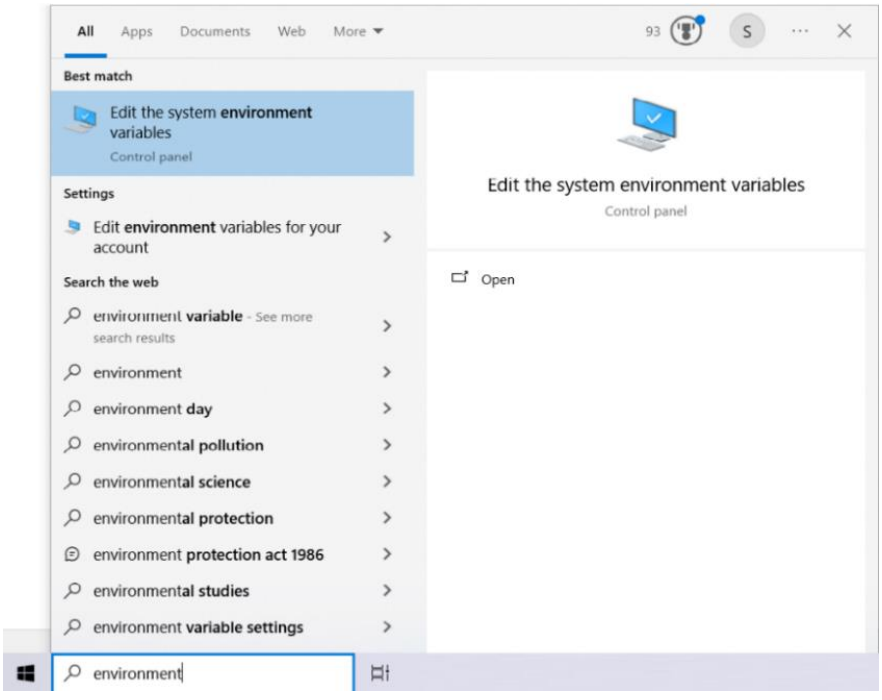
- **HIVE_HOME**: This is the Hive installation directory path in the machine *(in our case, it is* `D:\Programs\Hive\apache-hive-4.0.0-bin`)

- **HADOOP_USER_CLASSPATH_FIRST**: Set this variable value to `true` for Hive to use Hadoop user Class path first. This ensures `log4j2.x` and `jline` jars are loaded ahead of the jars pulled by Hadoop.

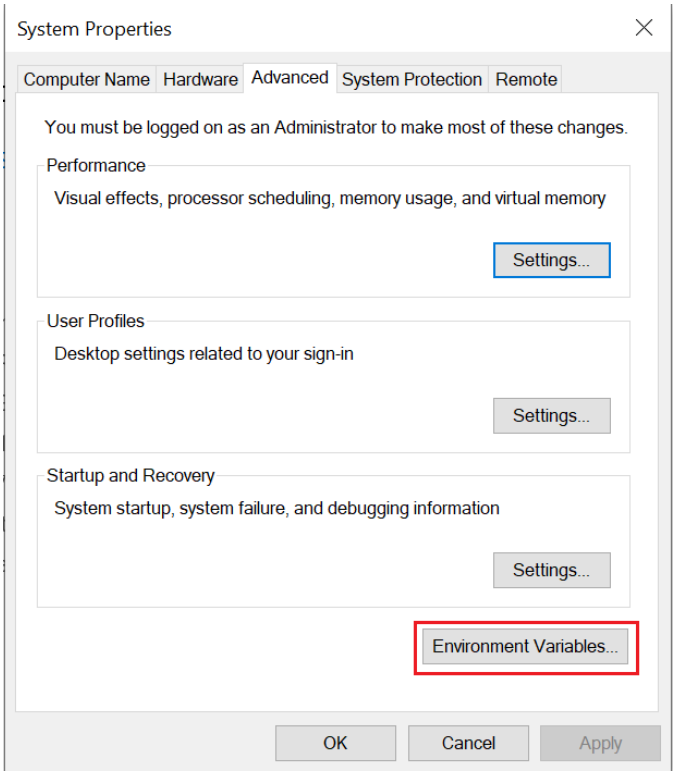
These variables need to be added to either **User environment variables** or **System environment variables** depending on Hive configuration needed **for a single user** or **for multiple users**.

In this tutorial, we will add User environment variables since we are configuring Hive for a single user. If you would like to configure Hive for multiple users, then define System environment variables.

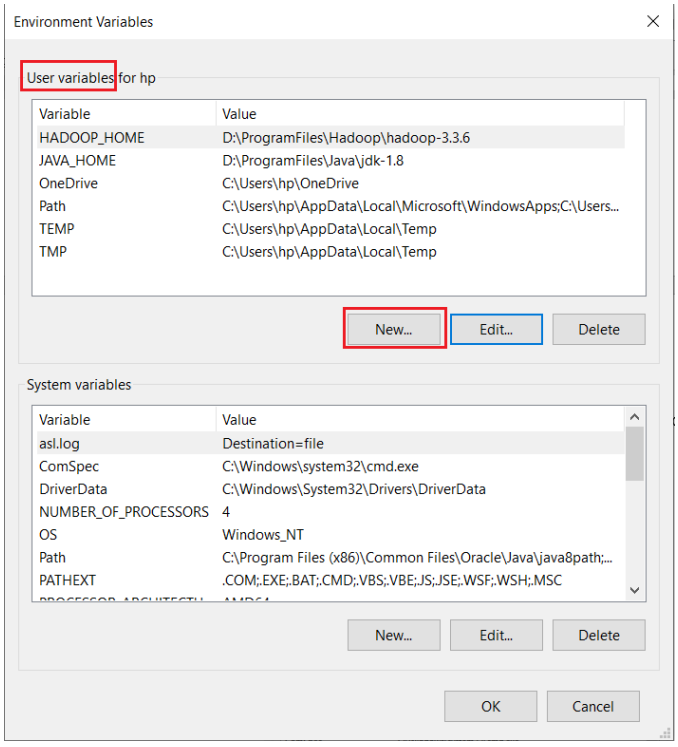
In the Windows search bar, start typing “environment variables” and select the first match which opens up **System Properties** dialog.



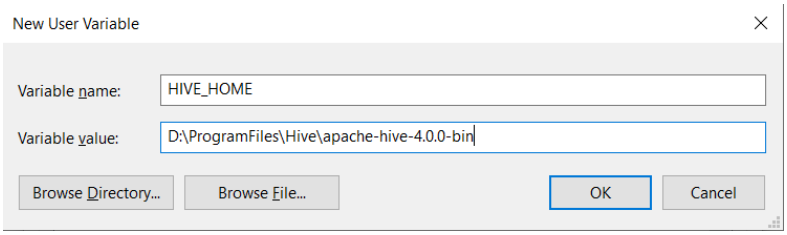
On the **System Properties** window, press **Environment Variables** button.



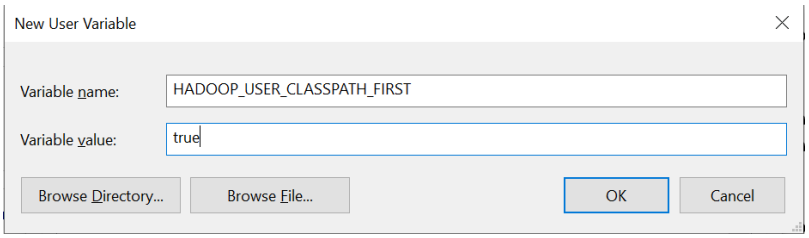
On the **Environment Variables** dialog, press **New** under **User variables** section.



Add `HIVE_HOME` variable and press OK.

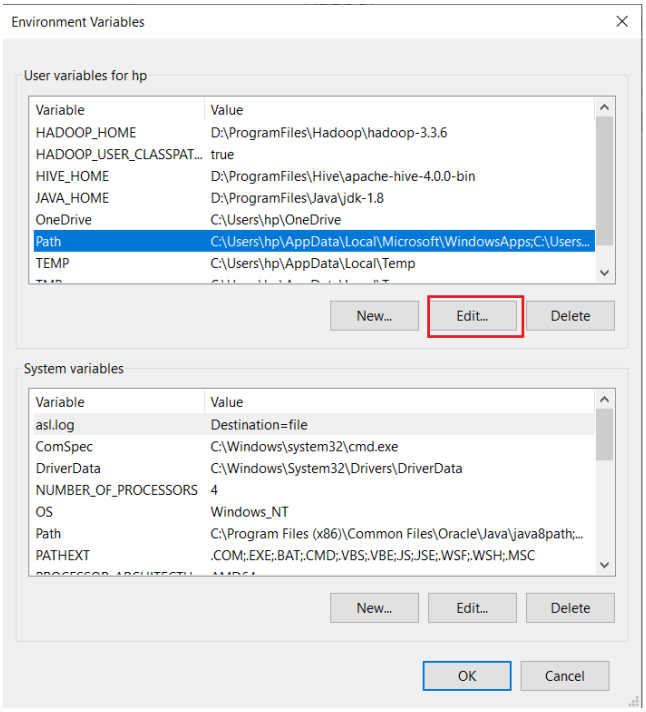


Press **New** again and add `HADOOP_USER_CLASSPATH_FIRST` variable to `true` and press OK.

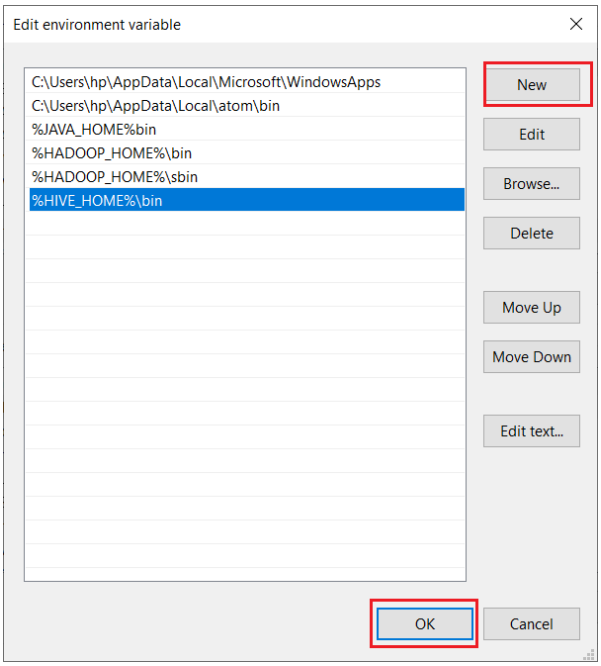


Now, we will update `PATH` variable to add Hive binary path

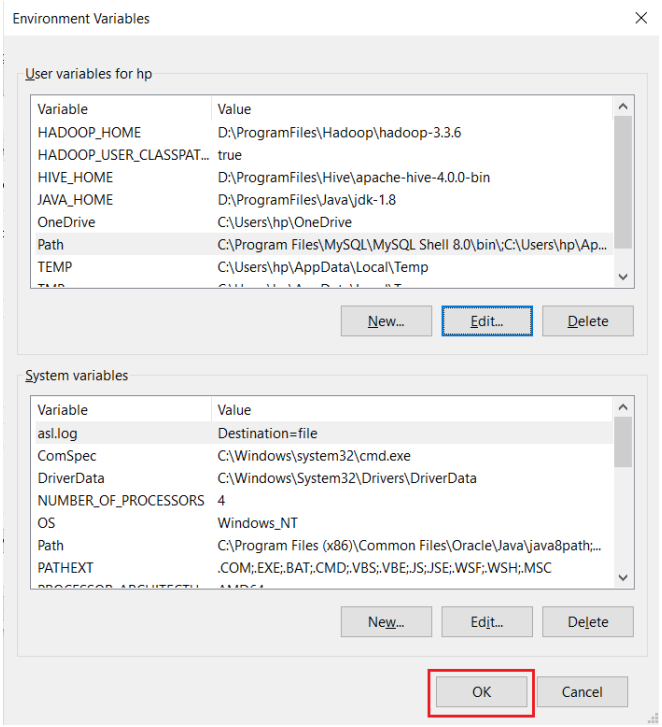
Select `PATH` variable under **User variables** and press **Edit** button.



Press **New** and add `%HIVE_HOME%\bin` path and press OK.



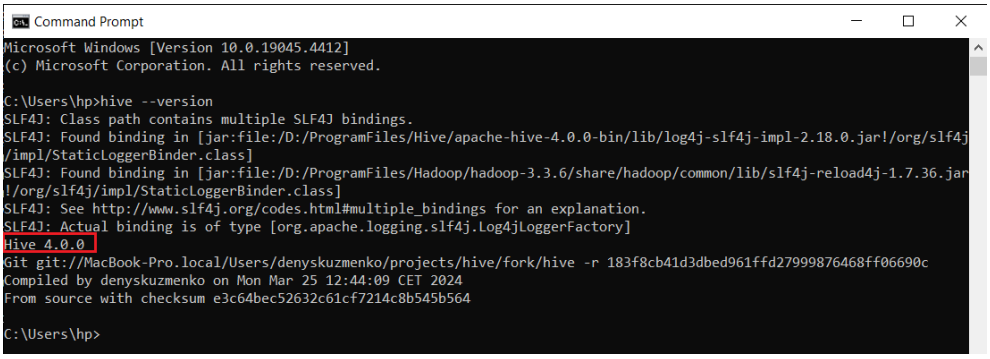
Press OK to apply environment variable changes and close window.



5. Verify Hive Installation:

Open **Windows PowerShell** or **Command Prompt** and run the following command to verify if Hive is installed properly:

```
hive --version
```



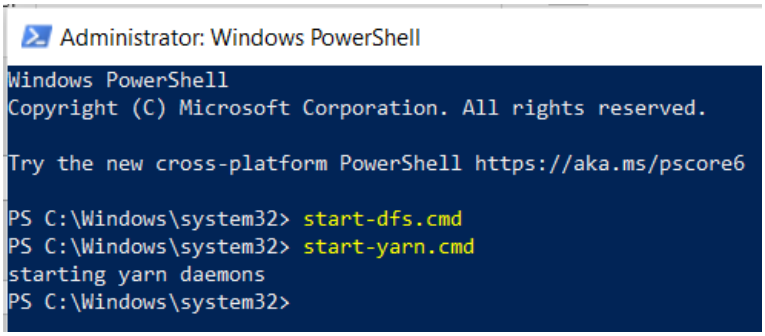
This may take couple of minutes to complete and displays **Hive 4.0.0** version installed.

6. Start Hadoop Services:

Before starting Hive, Hadoop services must be running since Hive runs on top of HDFS.

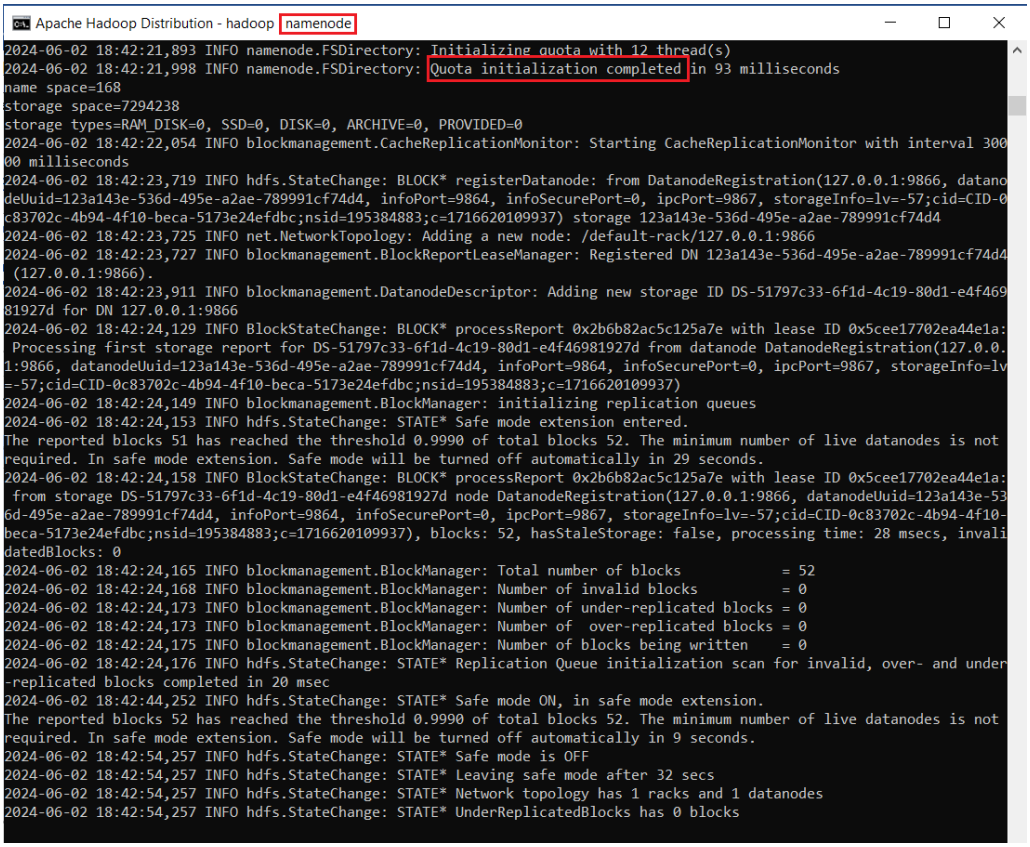
Open **Windows Command Prompt** or **Windows PowerShell** in **Administrator** mode and run the following commands to start Hadoop services.

```
start-dfs.cmd
start-yarn.cmd
```

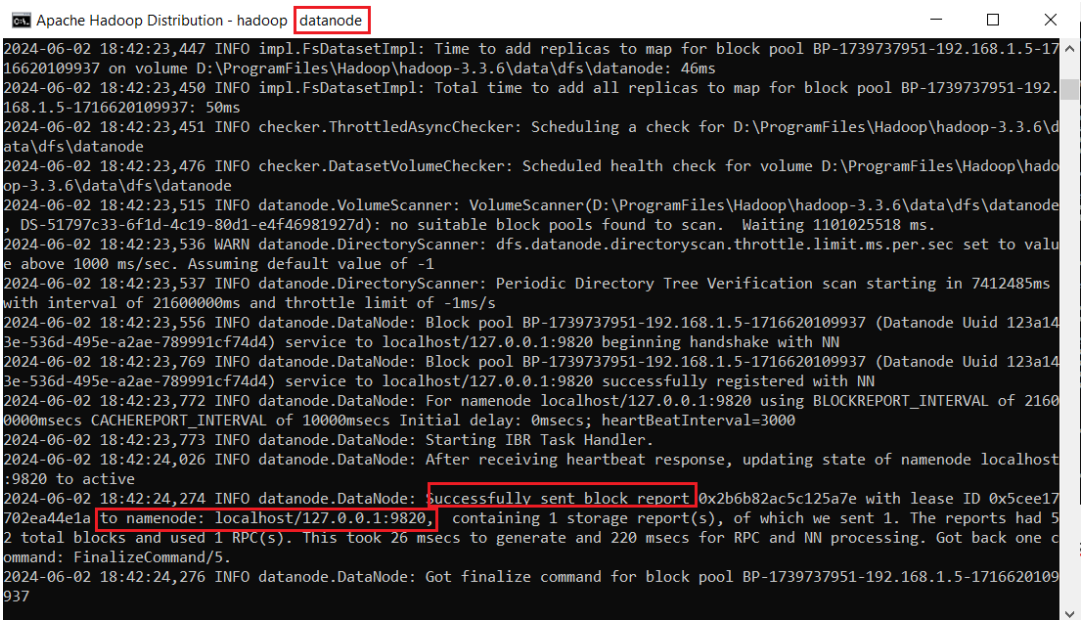


After executing the above commands, we can see four Windows command prompts opened for **namenode**, **datanode, resourcemanager and nodemanager** as below:

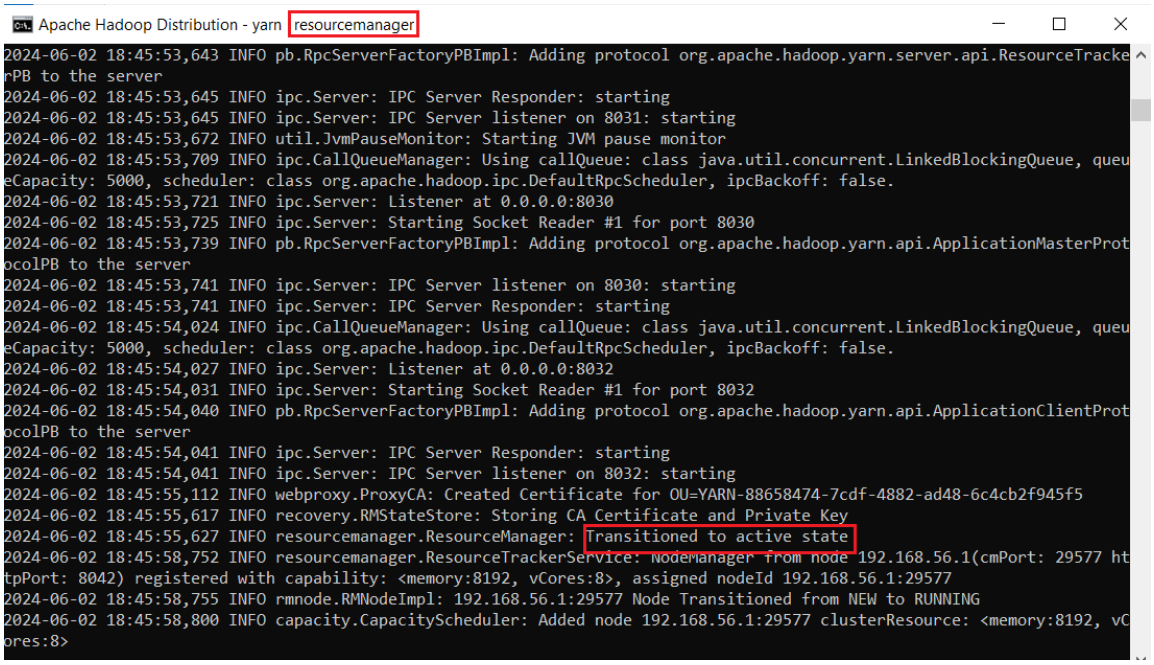
Wait for `namenode` service to say *"Quota initialization completed"*.



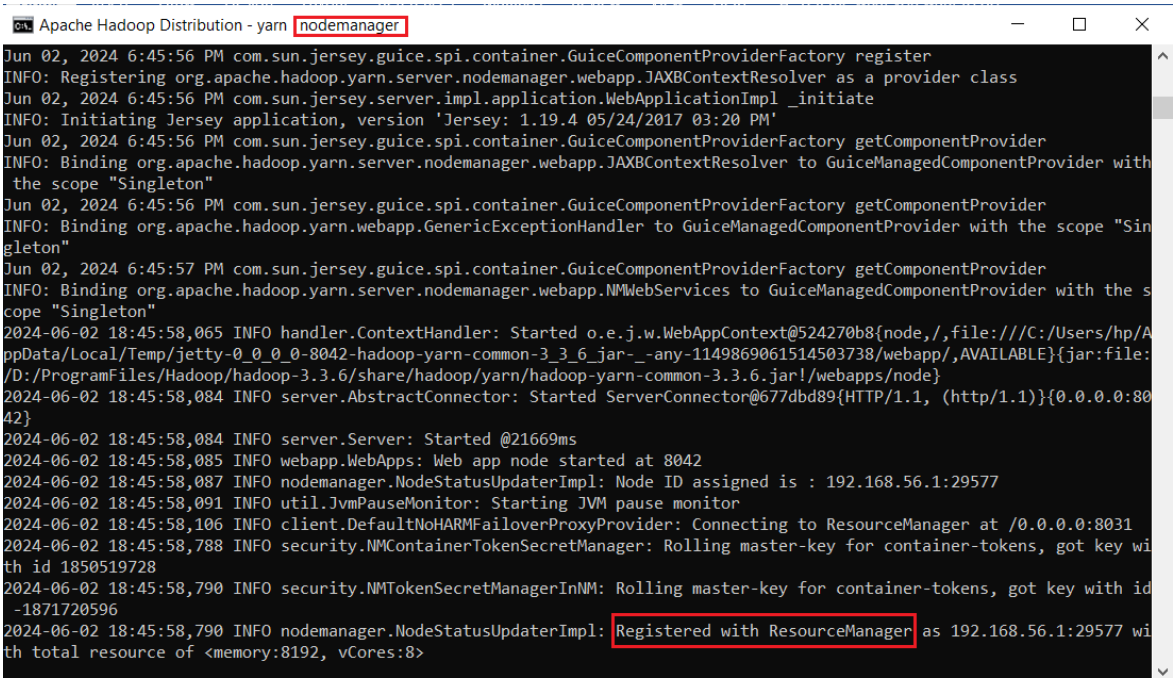
Wait for `datanode` service to say *"Successfully sent block report to namenode: localhost/127.0.0.1:9820"*.



Wait for `resourcemanager` service to say "*Transitioned to active state*".



Wait for `nodemanager` service to say "*Registered with ResourceManager*".



7. Configure Embedded Derby Metastore:

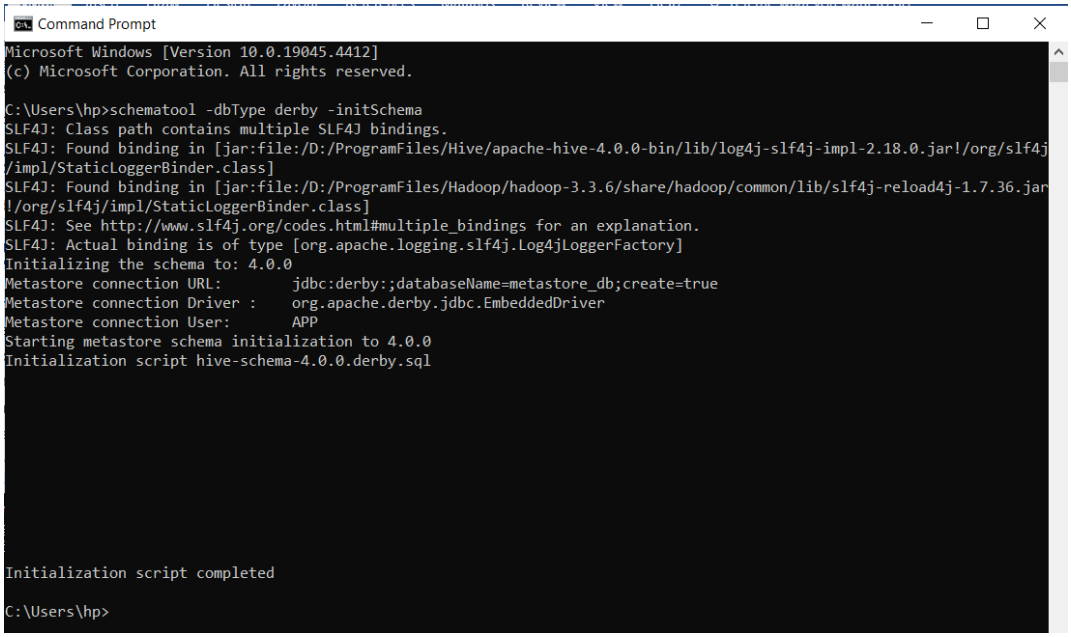
Hive by default runs with Embedded Metastore of Derby database on Hadoop File System.

7.1. Initialize Hive Metastore:

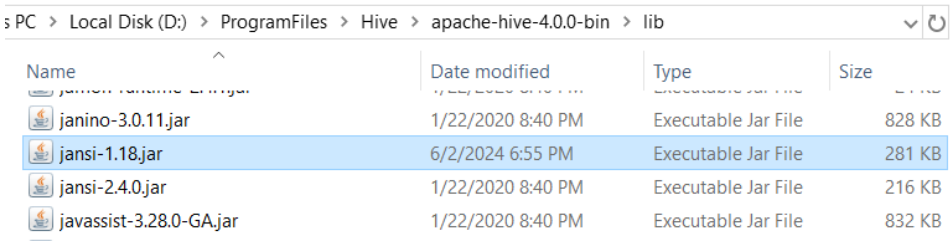
To start Hive, we need to initialize Hive Metastore (`metastore_db`) using `schematool` utility provided by Hive.

Execute the following command in Command Prompt or Windows PowerShell

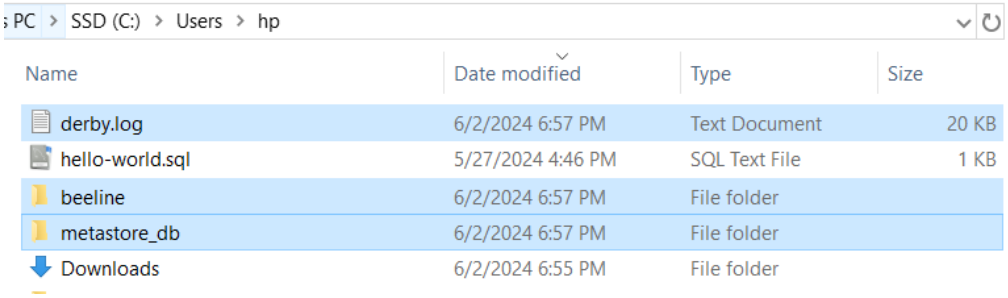
```
schematool -dbType derby -initSchema
```



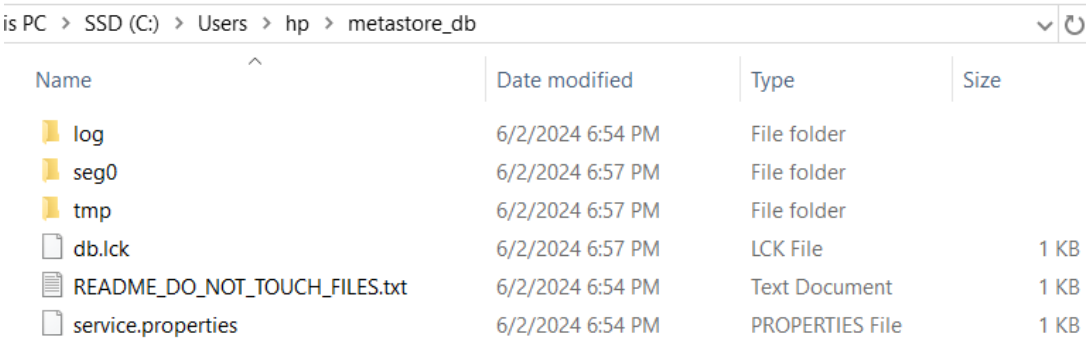
Note: You may encounter Exception in thread "main" java.lang.NoSuchMethodError: org.fusesource.jansi.AnsiConsole.wrapOutputStream(Ljava/io/OutputStream;)Ljava/io/OutputStream. To resolve this error, download `jansi-1.18.jar` file from the official Jansi Download website and copy to `HIVE_HOME\lib` directory.



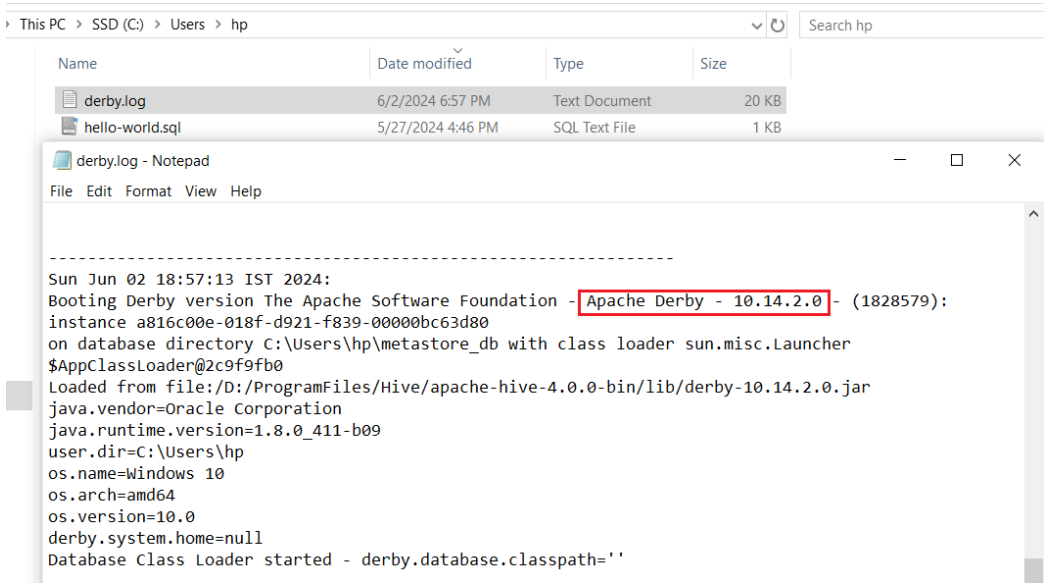
After executing the above command, it creates `metastore_db` folder, `beeline` folder and `derby.log` file in the location from where ever `schematool` utility got executed.



- `metastore_db` folder contains the database files of Hive metastore.



- Open `derby.log` file to see that Hive has booted Apache Derby database of `10.14.2.0` version.



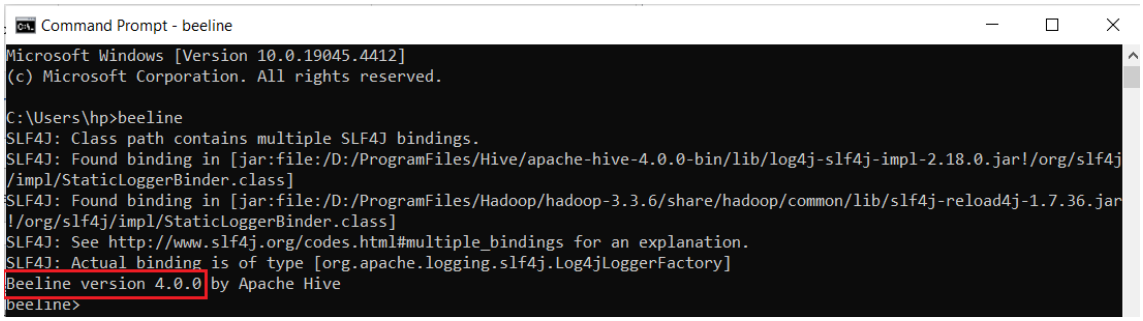
7.2. Start Beeline CLI:

Beeline is the advanced version of Command Line Interface that comes with HiveServer2 service. Beeline is a JDBC client that is based on SQLLine CLI.
From Hive 4.x version, Hive CLI has been deprecated and Beeline is used instead.

Beeline works in both embedded mode and remote mode. In embedded mode, Beeline connects to an embedded HiveServer2 service and executes Hive commands.

To start Beeline, open command prompt and run this command.

```
beeline
```



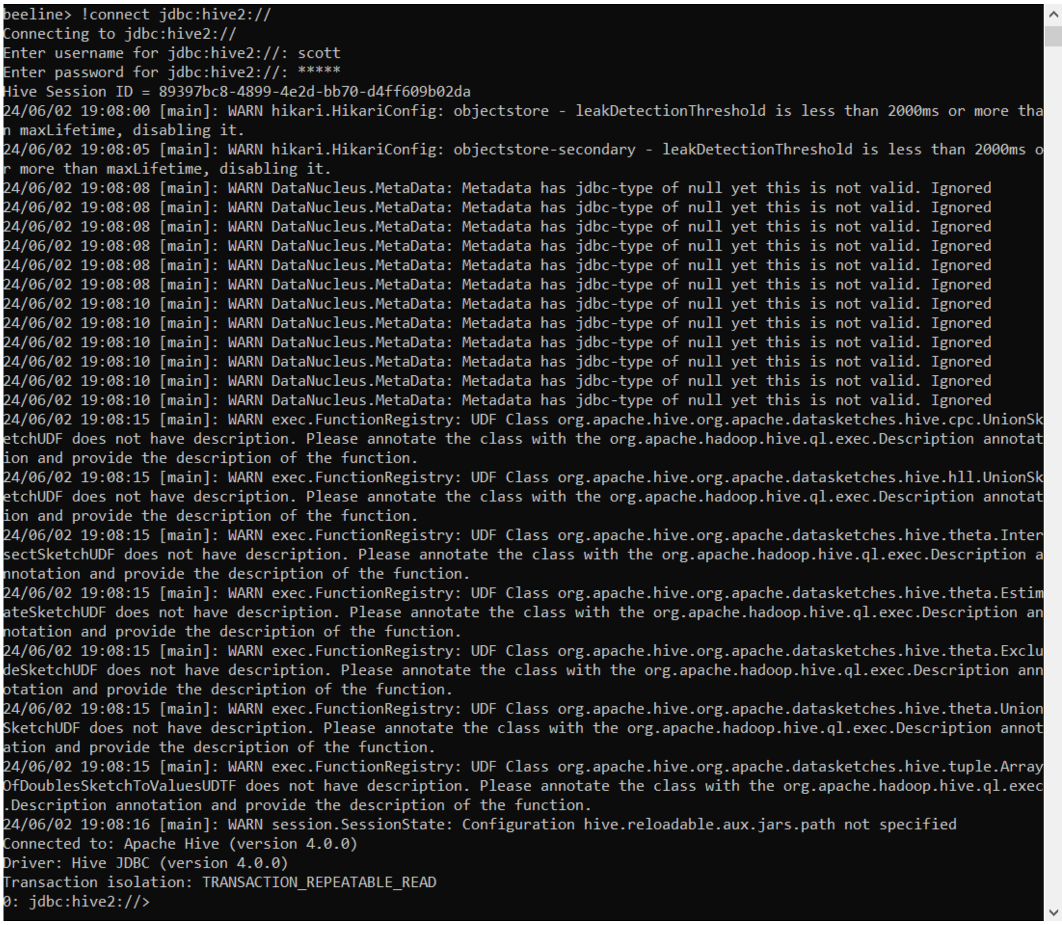
After executing the above command, it prints the **Beeline version 4.0.0 by Apache Hive** and provides us `beeline>` prompt.

To get a list of Beeline commands, type `!` on `beeline>` prompt and press Tab key. Go through this Apache Hive documentation to get more understanding on these commands.

To connect to embedded HiveServer2, enter the following command on `beeline>` prompt.

```
!connect jdbc:hive2://
```

It asks for username and password to connect. Enter the default username `scott` and password `tiger` which are provided by HiveServer2.



```
beeline> !connect jdbc:hive2://
Connecting to jdbc:hive2://
Enter username for jdbc:hive2://: scott
Enter password for jdbc:hive2://: *****
Hive Session ID = 89397bc8-4899-4e2d-bb70-d4ff609b02da
24/06/02 19:08:00 [main]: WARN hikari.HikariConfig: objectstore - leakDetectionThreshold is less than 2000ms or more tha
n maxLifetime, disabling it.
24/06/02 19:08:05 [main]: WARN hikari.HikariConfig: objectstore-secondary - leakDetectionThreshold is less than 2000ms o
r more than maxLifetime, disabling it.
24/06/02 19:08:08 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:08 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:08 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:08 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:08 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:08 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:10 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:10 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:10 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:10 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:10 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:10 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 19:08:15 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.cpc.UnionSk
etchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description annotat
ion and provide the description of the function.
24/06/02 19:08:15 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.hll.UnionSk
etchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description annotat
ion and provide the description of the function.
24/06/02 19:08:15 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.theta.Inter
sectSketchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description a
nnotation and provide the description of the function.
24/06/02 19:08:15 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.theta.Estim
ateSketchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description an
notation and provide the description of the function.
24/06/02 19:08:15 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.theta.Exclu
deSketchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description ann
otation and provide the description of the function.
24/06/02 19:08:15 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.theta.Union
SketchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description annot
ation and provide the description of the function.
24/06/02 19:08:15 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.tuple.Array
OfDoublesSketchToValuesUDTF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec
.Description annotation and provide the description of the function.
24/06/02 19:08:16 [main]: WARN session.SessionState: Configuration hive.reloadable.aux.jars.path not specified
Connected to: Apache Hive (version 4.0.0)
Driver: Hive JDBC (version 4.0.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://>
```

We can also provide HiveServer2 credentials in the `beeline connect` command itself with this command

```
!connect jdbc:hive2:// -n scott -p tiger
```
or
```
!connect jdbc:hive2:// scott tiger
```

To start beeline connecting to HiveServer2 directly, use the following command:

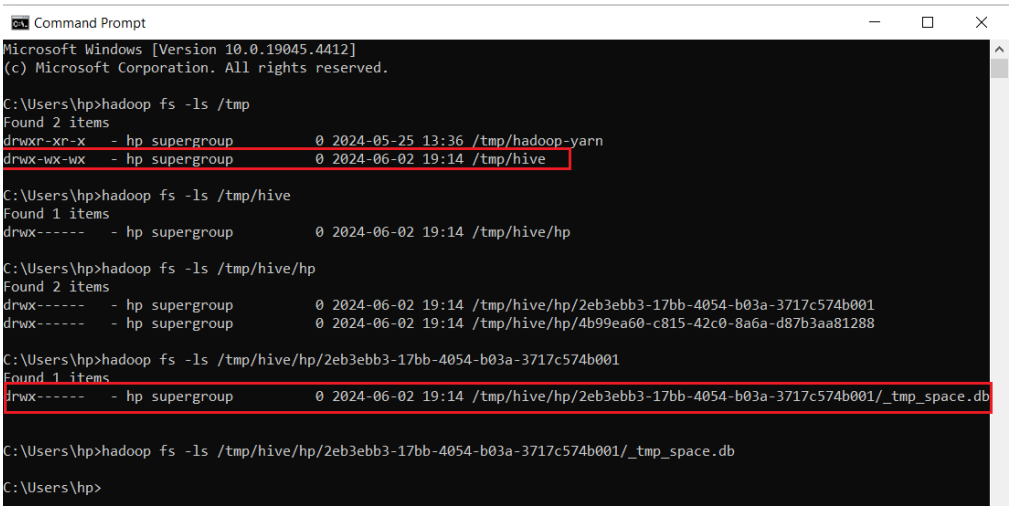
```
beeline -u jdbc:hive2:// -n scott -p tiger
```
or
```
beeline -u jdbc:hive2:// scott tiger
```

As soon as Beeline is connected, it creates a `/tmp/hive` folder on Hadoop File System.

Open another command prompt and execute this command to see `/tmp/hive` folder on HDFS

```
hadoop fs -ls /tmp
```

Hive uses `/tmp` HDFS directory for storing `temp_space` database which is available at `/tmp/hive/<userid>/<temp_folder>/_temp_space.db`



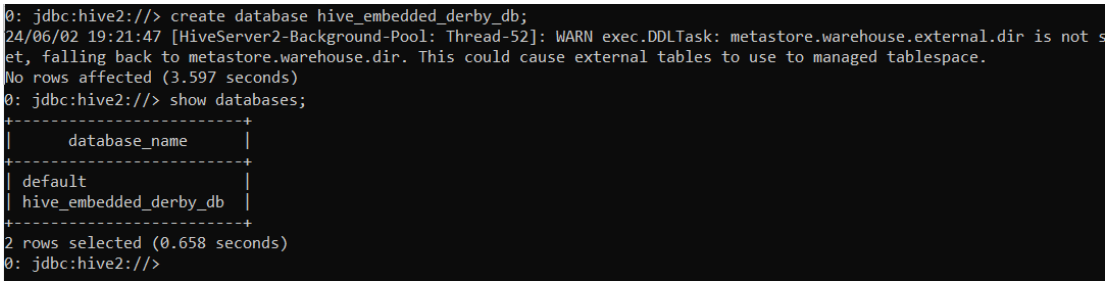
To come out of `beeline>` shell, use `!quit` command.

7.3. Run Queries on Beeline CLI:

Open Beeline CLI, connect to HiveServer2 and run the following queries to create a hive metadata database, create table, load data and select data.

- Create a database in Hive metastore:

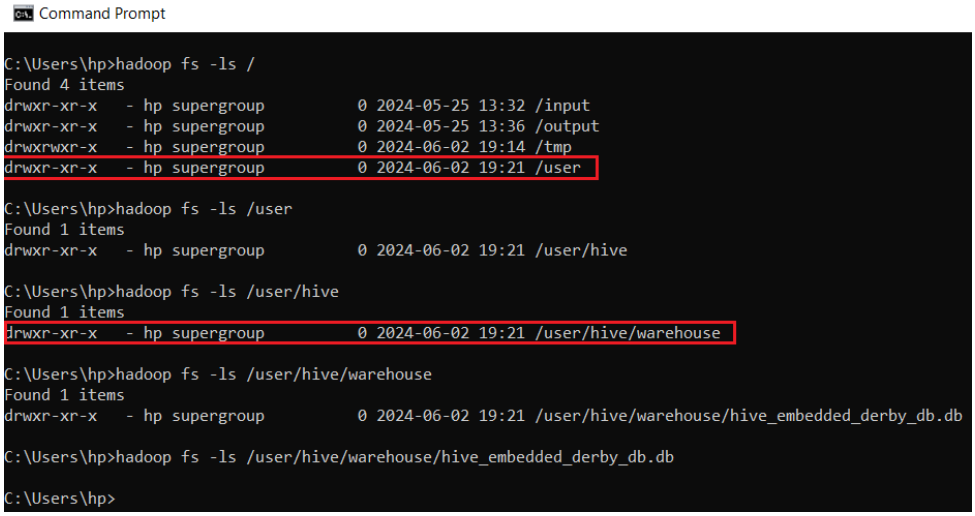
```
create database hive_embedded_derby_db;
show databases;
```



As soon as the above query is executed, Hive creates the default warehouse directory `/user/hive/warehouse` in which creates a `hive_embedded_derby_db.db` folder in Hadoop File System.

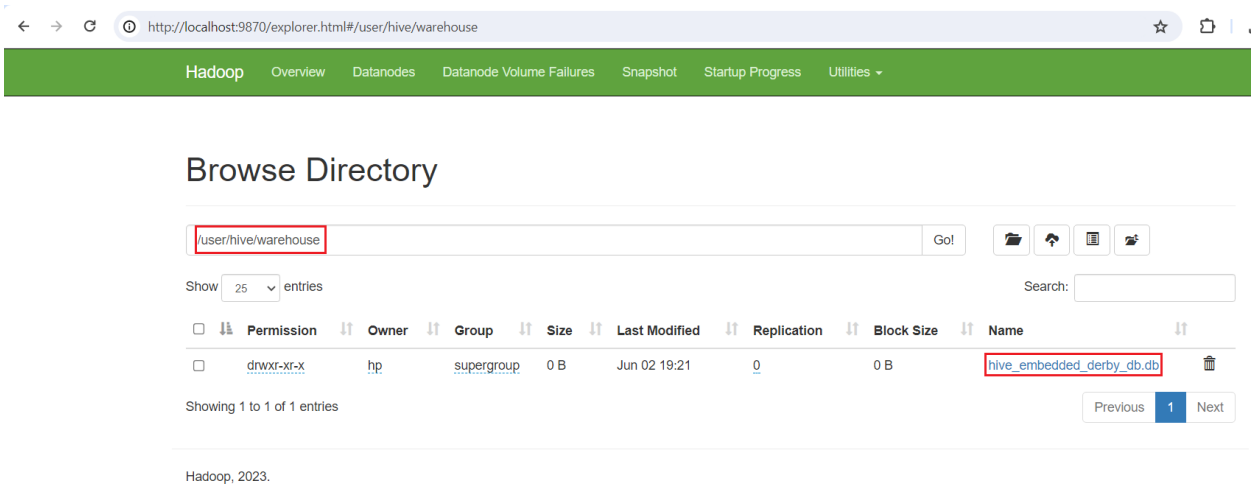
Run this command in another command prompt to validate in HDFS:

```
hadoop fs -ls /user/hive/warehouse
```



We can verify the same in NameNode UI: http://localhost:9870/dfshealth.html

Open NameNode UI, go to **Utilities** tab and select **Browse the file system** option. Enter the directory name `/user/hive/warehouse` and you can see `hive_embedded_derby_db.db` folder available.



- Create a table in Hive metastore:

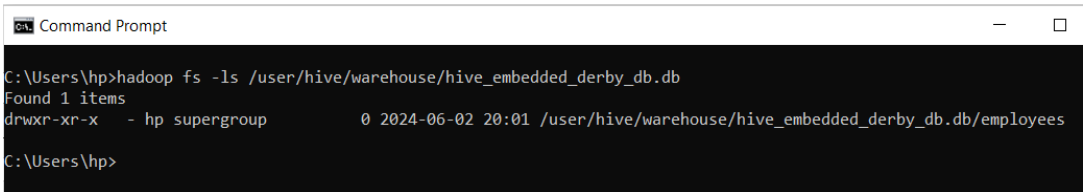
```
use hive_embedded_derby_db;
create table employees(emp_id int, emp_name string, emp_salary int);
show tables;
```

```
0: jdbc:hive2://> use hive_embedded_derby_db;
No rows affected (0.03 seconds)
0: jdbc:hive2://> create table employees(emp_id int, emp_name string, emp_salary int);
No rows affected (1.436 seconds)
0: jdbc:hive2://> show tables;
+------------+
|  tab_name  |
+------------+
| employees  |
+------------+
1 row selected (0.282 seconds)
0: jdbc:hive2://>
```

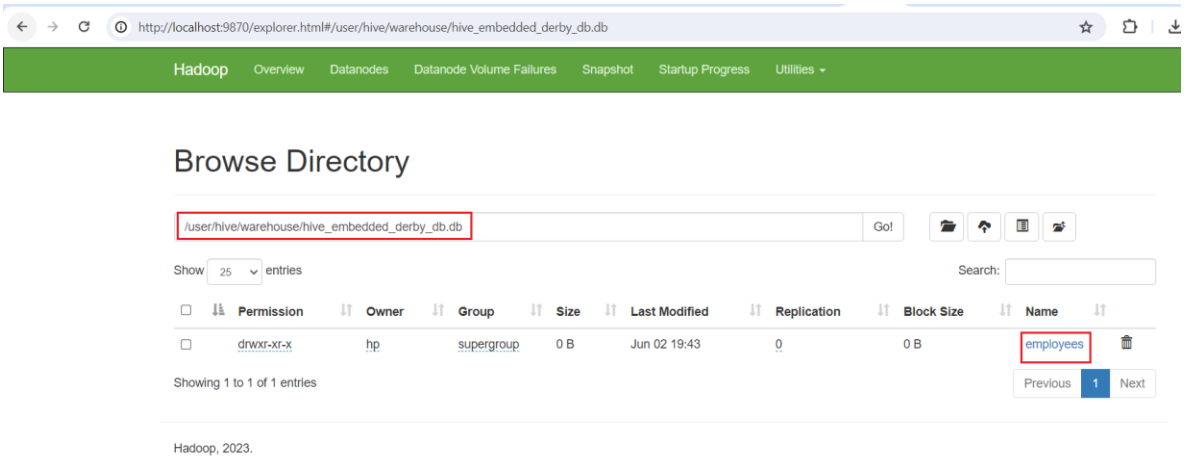
We can see the above table is saved under `/user/hive/warehouse/hive_embedded_derby_db.db` HDFS location.

Run this command in another command prompt to validate in HDFS:

```
hadoop fs -ls /user/hive/warehouse/hive_embedded_derby_db.db
```



We can verify the same in NameNode UI. Click on `hive_embedded_derby_db.db` folder in `/user/hive/warehouse` directory and you can see `employees` folder available.



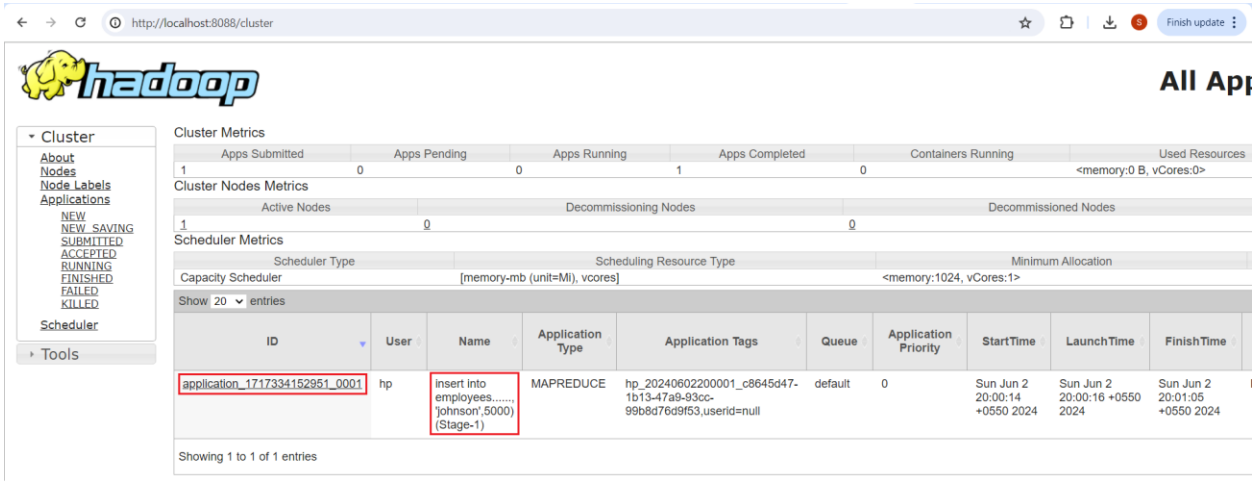
- Insert data into the table:

```
insert into employees values (101, 'johnson',5000);
```

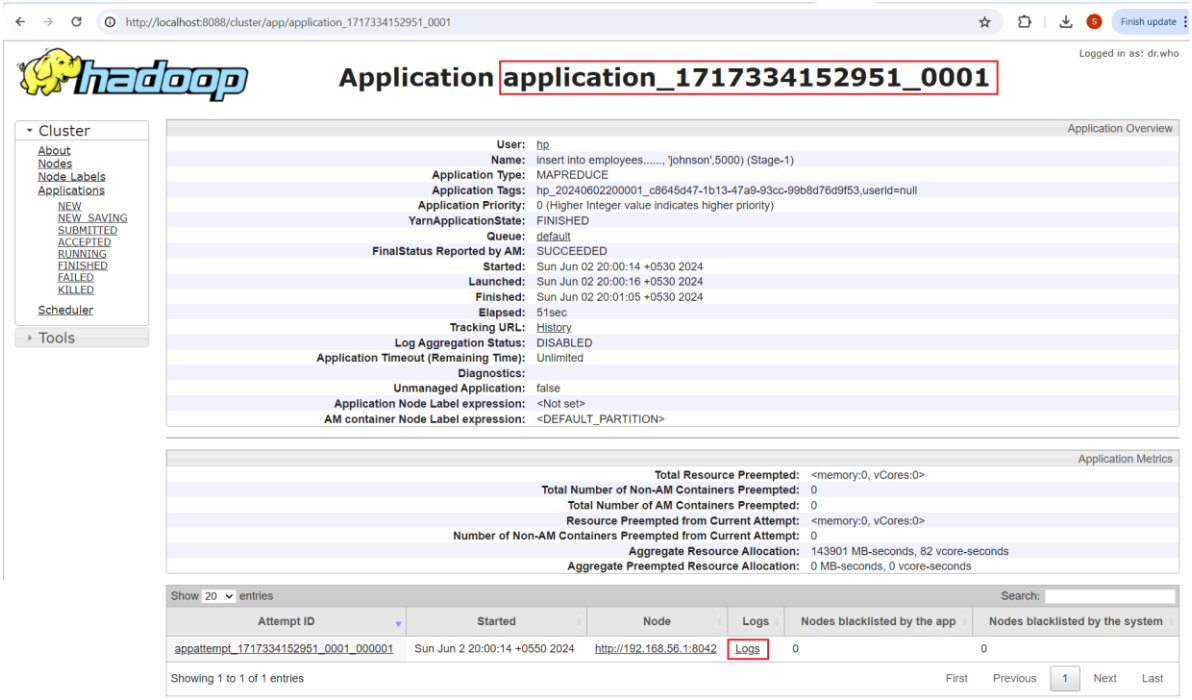
The above insert command submits the MapReduce job to get the record into table.

```
0: jdbc:hive2://> insert into employees values (101, 'johnson',5000);
24/06/02 20:00:08 [HiveServer2-Background-Pool: Thread-104]: WARN ql.Driver: Hive-on-MR is deprecated in Hive 2 and may
not be available in the future versions. Consider using a different execution engine (i.e. tez) or using Hive 1.X releas
es.
Query ID = hp_20240602200001_c8645d47-1b13-47a9-93cc-99b8d76d9f53
Total jobs = 3
Launching Job 1 out of 3
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
WARN  : Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different e
xecution engine (i.e. tez) or using Hive 1.X releases.
24/06/02 20:00:11 [HiveServer2-Background-Pool: Thread-104]: WARN mapreduce.JobResourceUploader: Hadoop command-line opt
ion parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
Starting Job = job_1717334152951_0001, Tracking URL = http://DESKTOP-KGH2E2G:8088/proxy/application_1717334152951_0001/
Kill Command = D:\ProgramFiles\Hadoop\hadoop-3.3.6\bin\mapred job  -kill job_1717334152951_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
24/06/02 20:00:34 [HiveServer2-Background-Pool: Thread-104]: WARN mapreduce.Counters: Group org.apache.hadoop.mapred.Tas
k$Counter is deprecated. Use org.apache.hadoop.mapreduce.TaskCounter instead
2024-06-02 20:00:34,557 Stage-1 map = 0%,  reduce = 0%
2024-06-02 20:00:49,409 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 9.404 sec
2024-06-02 20:01:05,364 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 19.62 sec
MapReduce Total cumulative CPU time: 19 seconds 620 msec
Ended Job = job_1717334152951_0001
Stage-4 is selected by condition resolver.
Stage-3 is filtered out by condition resolver.
Stage-5 is filtered out by condition resolver.
Moving data to directory hdfs://localhost:9820/user/hive/warehouse/hive_embedded_derby_db.db/employees/.hive-staging_hiv
e_2024-06-02_20-00-01_482_1593368538904413760-1/-ext-10000
Loading data to table hive_embedded_derby_db.employees
24/06/02 20:01:08 [HiveServer2-Background-Pool: Thread-104]: WARN metadata.Hive: Cannot get a table snapshot for employe
es
24/06/02 20:01:08 [HiveServer2-Background-Pool: Thread-104]: WARN metadata.Hive: Cannot get a table snapshot for employe
es
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 19.62 sec   HDFS Read: 24647 HDFS Write: 306 HDFS EC Read: 0 SUCCESS
Total MapReduce CPU Time Spent: 19 seconds 620 msec
1 row affected (67.214 seconds)
```

We can track the job status on YARN UI: http://localhost:8088/cluster
In YARN UI, you can see an application name with `insert into employees…` that was executed.



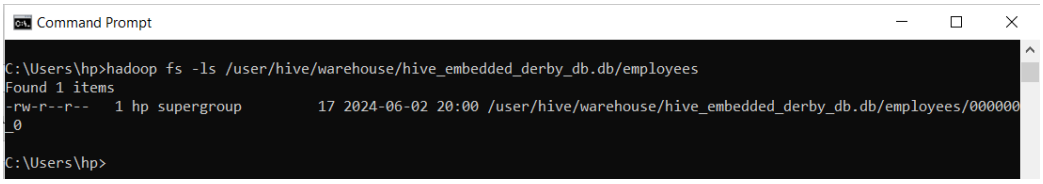
Click on the application ID to see the additional job details and logs.



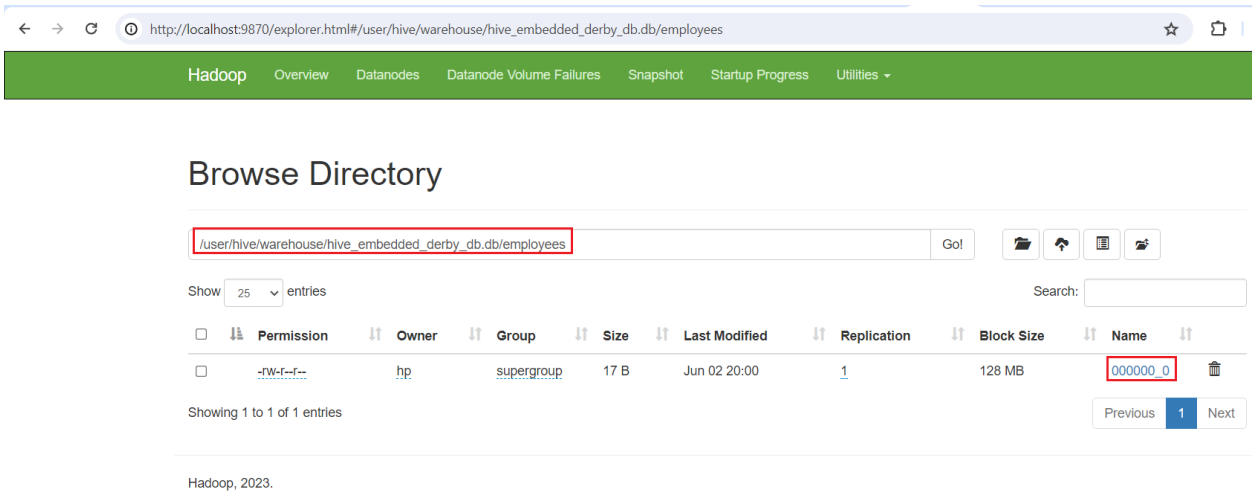
After the above job is succeeded, we can see a file created under `/user/hive/warehouse/hive_embedded_derby_db.db/employees` HDFS location.

Open another Command Prompt and run this command:

```
hadoop fs -ls /user/hive/warehouse/hive_embedded_derby_db.db/employees
```



In NameNode UI, click on `employees` folder in `/user/hive/warehouse/hive_embedded_derby_db.db` directory and you can see a file `000000_0` available.



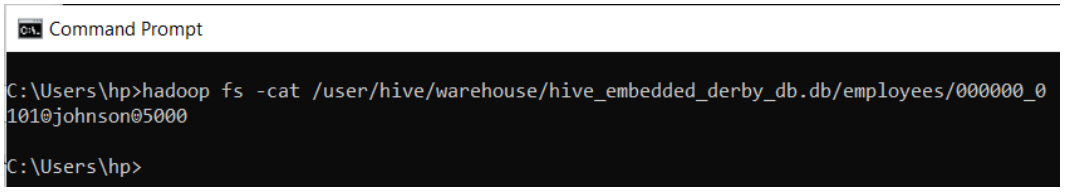
- Select data from the Hive table:

```
select * from employees;
```

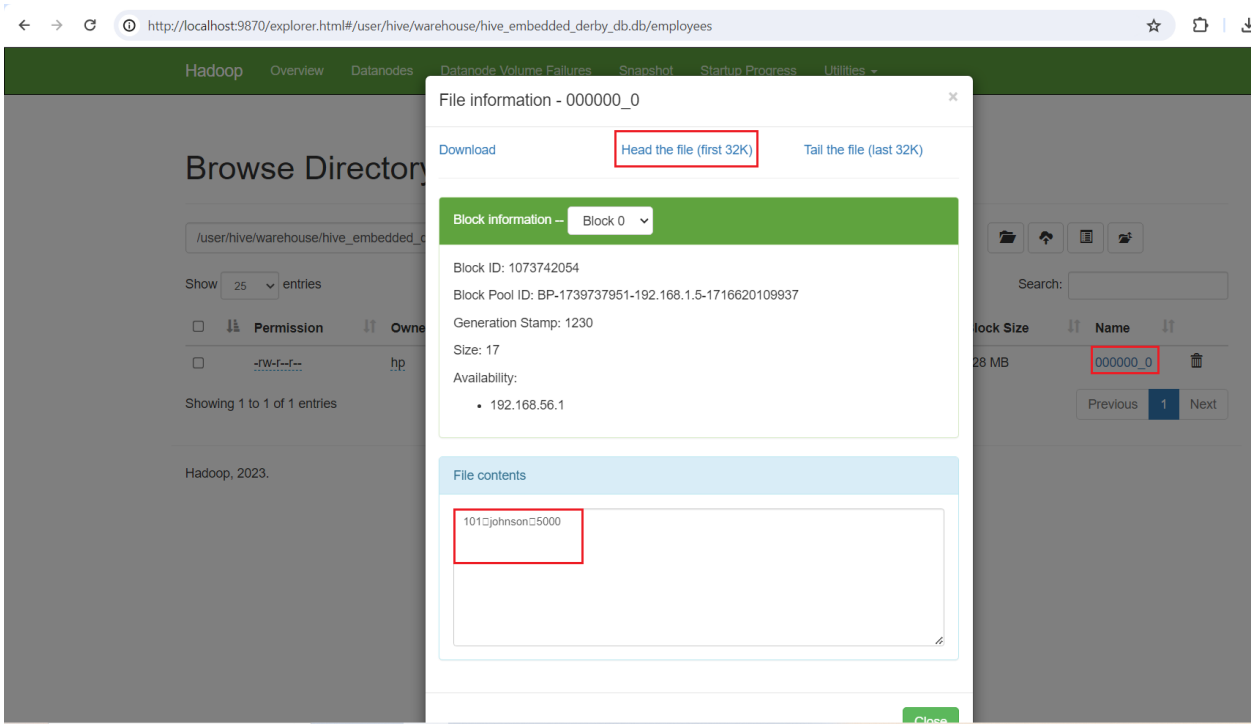
```
0: jdbc:hive2://> select * from employees;
24/06/02 20:17:45 [2eb3ebb3-17bb-4054-b03a-3717c574b001 main]: WARN optimizer.SimpleFetchOptimizer: Table hive_embedded_
derby_db@employees is external table, falling back to filesystem scan.
+-------------------+---------------------+-----------------------+
| employees.emp_id  | employees.emp_name  | employees.emp_salary  |
+-------------------+---------------------+-----------------------+
| 101               | johnson             | 5000                  |
+-------------------+---------------------+-----------------------+
1 row selected (0.561 seconds)
```

We can verify the above table output in HDFS using the following command

```
hadoop fs -cat /user/hive/warehouse/hive_embedded_derby_db.db/employees/000000_0
```



In NameNode UI, click on `000000_0` file and select **Head the file** or **Tail the file** to see the file contents. We can download this file by clicking on **Download** option.

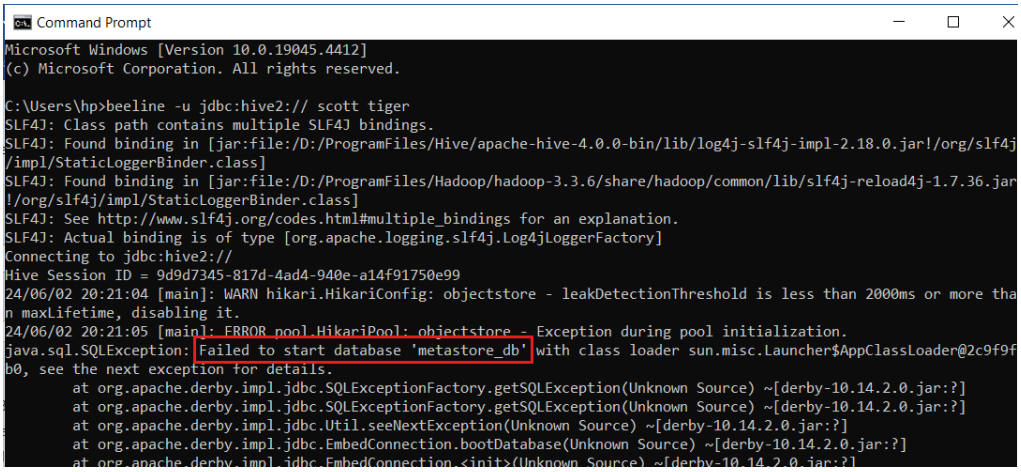


Note:

In embedded Hive metastore mode, we cannot start a second Beeline session while the current one is still active because embedded metastore allows only one connection at any time.

To test this, open another Command Prompt, start Beeline and connect to embedded HiveServer2 using the following command

```
beeline -u jdbc:hive2:// scott tiger
```



```
        at org.apache.derby.jdbc.InternalDriver$LoginCallable.call(Unknown Source) ~[derby-10.14.2.0.jar:?]
        at java.util.concurrent.FutureTask.run(FutureTask.java:266) ~[?:1.8.0_411]
        at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149) ~[?:1.8.0_411]
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624) ~[?:1.8.0_411]
        at java.lang.Thread.run(Thread.java:750) ~[?:1.8.0_411]
Caused by: org.apache.derby.iapi.error.StandardException: Failed to start database 'metastore_db' with class loader sun.
misc.Launcher$AppClassLoader@2c9f9fb0, see the next exception for details.
        at org.apache.derby.iapi.error.StandardException.newException(Unknown Source) ~[derby-10.14.2.0.jar:?]
        at org.apache.derby.impl.jdbc.SQLExceptionFactory.wrapArgsForTransportAcrossDRDA(Unknown Source) ~[derby-10.14.2
.0.jar:?]
        ... 15 more
Caused by: org.apache.derby.iapi.error.StandardException: Another instance of Derby may have already booted the database
C:\Users\hp\metastore_db.
        at org.apache.derby.iapi.error.StandardException.newException(Unknown Source) ~[derby-10.14.2.0.jar:?]
        at org.apache.derby.iapi.error.StandardException.newException(Unknown Source) ~[derby-10.14.2.0.jar:?]
        at org.apache.derby.impl.store.raw.data.BaseDataFileFactory.privGetJBMSLockOnDB(Unknown Source) ~[derby-10.14.2.
0.jar:?]
        at org.apache.derby.impl.store.raw.data.BaseDataFileFactory.run(Unknown Source) ~[derby-10.14.2.0.jar:?]
        at java.security.AccessController.doPrivileged(Native Method) ~[?:1.8.0_411]
        at org.apache.derby.impl.store.raw.data.BaseDataFileFactory.getJBMSLockOnDB(Unknown Source) ~[derby-10.14.2.0.ja
r:?]
        at org.apache.derby.impl.store.raw.data.BaseDataFileFactory.boot(Unknown Source) ~[derby-10.14.2.0.jar:?]
        at org.apache.derby.impl.services.monitor.BaseMonitor.boot(Unknown Source) ~[derby-10.14.2.0.jar:?]
        at org.apache.derby.impl.services.monitor.TopService.bootModule(Unknown Source) ~[derby-10.14.2.0.jar:?]
```

You can see that it has thrown errors “Failed to start database 'metastore_db'. Another instance of Derby may have already booted the database” because another Beeline session is active. Until the previous session is closed, we cannot start another Beeline session with Embedded metastore.

Close or exit out of all Beeline CLI and make sure that no sessions are active.

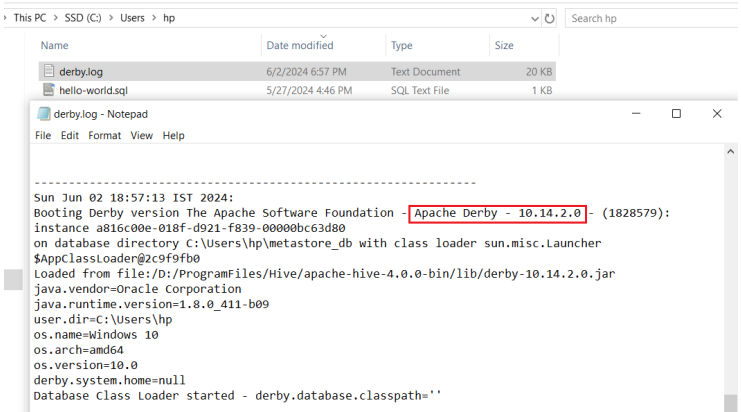
8. Configure Local Derby Metastore:

We can configure Hive metastore in local mode with Derby database by making Derby to run in network mode. For this, we should first download and install Apache Derby.

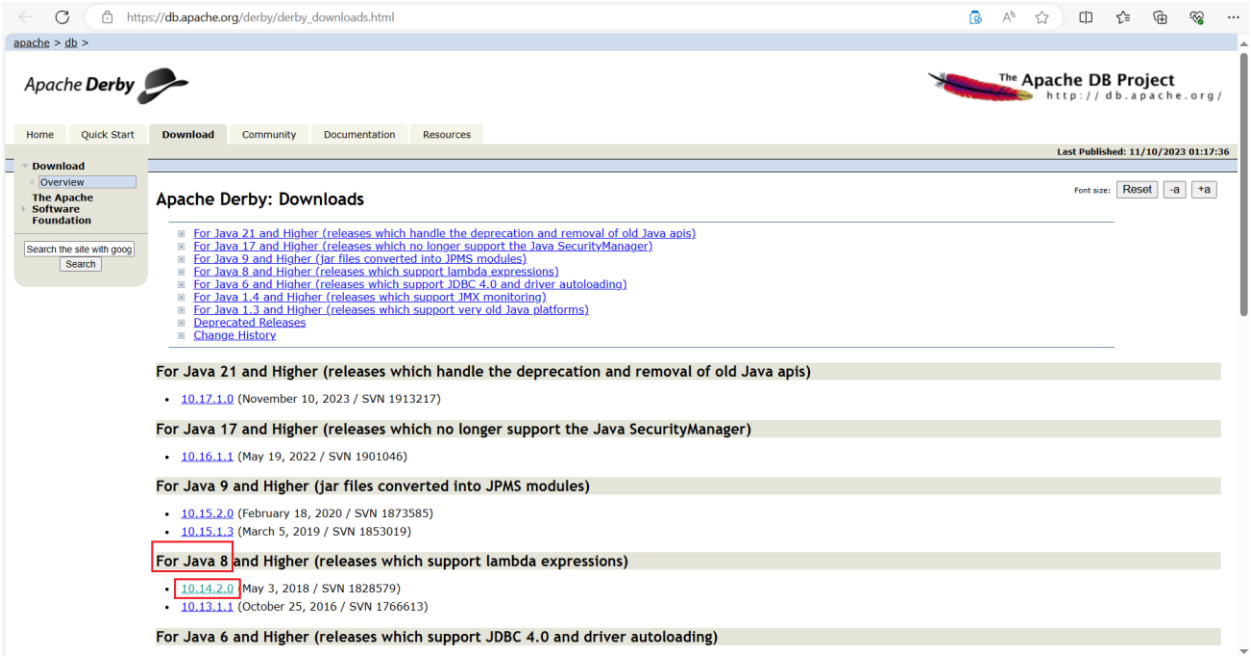
8.1. Install Apache Derby:

It is recommended to install Derby database of version that is shown for Embedded Derby installed with Hive.

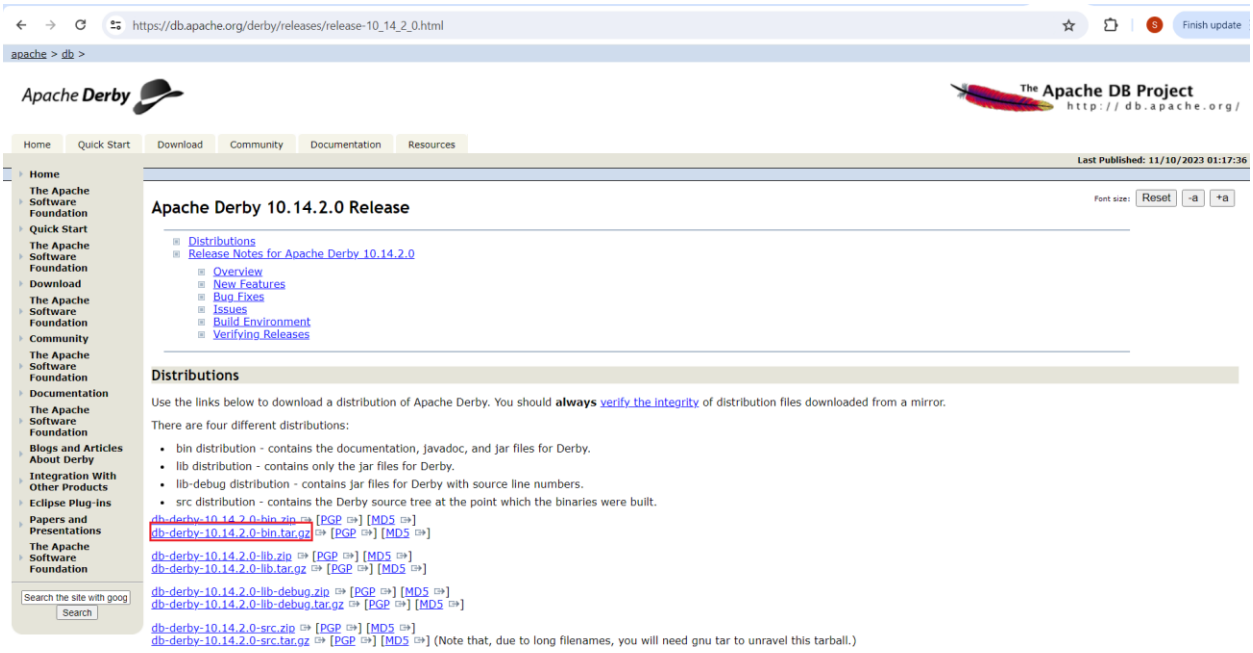
Open `derby.log` file available in the location where `metastore_db` is created to see the embedded Derby database version.



Additionally, we can check the appropriate Derby version in [Apache Derby Downloads] page for supported versions of Java releases. Since we have Java 8 to run Hadoop and Hive, the latest Derby release for Java 8 is **Apache Derby 10.14.2.0** version.

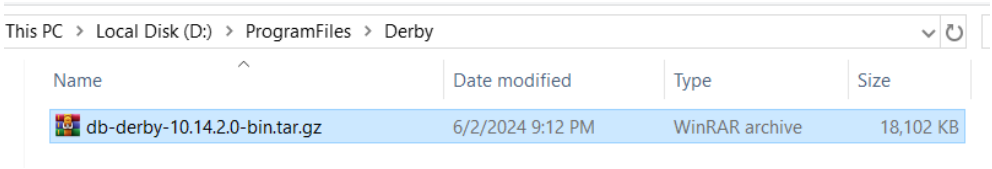


Download `db-derby-10.14.2.0-bin.tar.gz` file from the [Apache Derby 10.14.2.0 Release] website which gets downloaded to your **Downloads** folder.

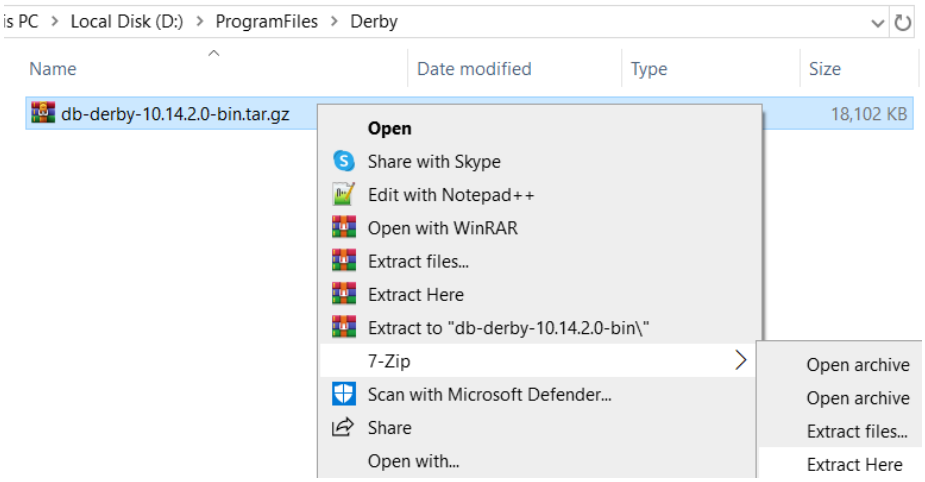


After the file is downloaded, unpack it using any file archiver (7zip or WinRAR) utility as below.

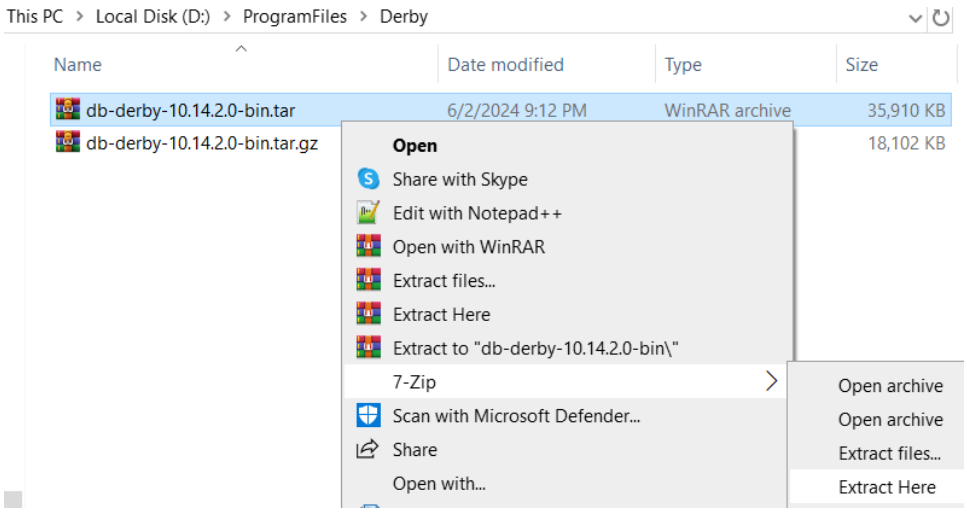
- Choose the installation directory in your machine and copy `db-derby-10.14.2.0-bin.tar.gz` file to that directory. Here, we are choosing Derby installation directory as `D:\ProgramFiles\Derby`.



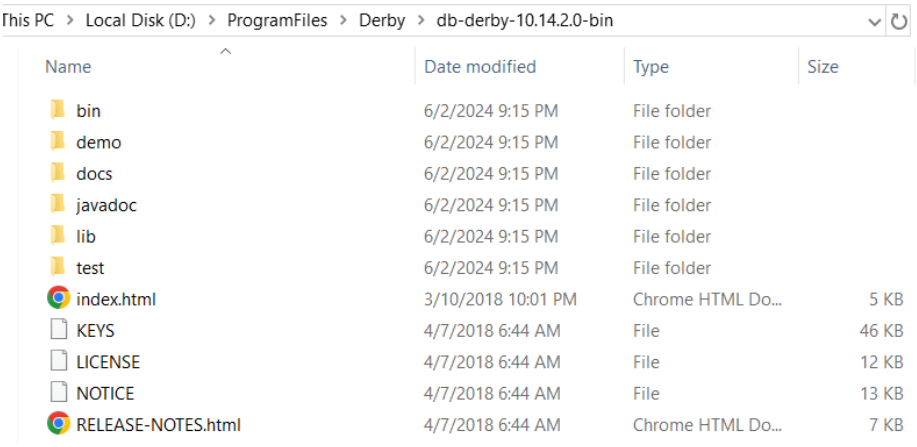
- Right click on `db-derby-10.14.2.0-bin.tar.gz` and choose **7-Zip -> Extract Here** option which extracts a new packed file `db-derby-10.14.2.0-bin.tar`.



- Next, unpack `db-derby-10.14.2.0-bin.tar` file using 7zip utility.



- The tar file extraction may take few minutes to finish. After finishing, you see a folder named `db-derby-10.14.2.0-bin` which consists of Derby binaries and libraries.



8.2. Set up Environment Variables:

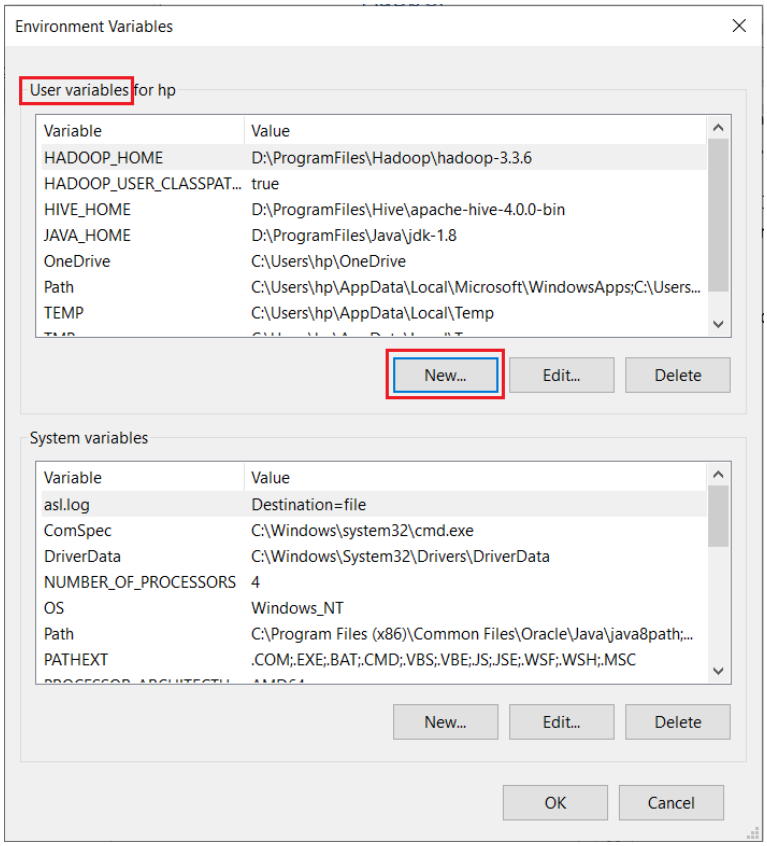
After installing Derby, we should configure two environment variables defining Derby installation path.

- **DERBY_HOME**: This is the Derby installation directory path in the machine *(in our case, it is `D:\ProgramFiles\Derby\db-derby-10.14.2.0-bin`)*

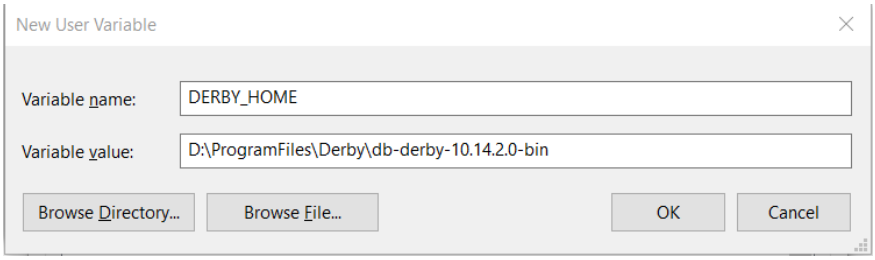
- **DERBY_OPTS**: Set it to `-Dderby.system.home=%DERBY_HOME%` location. This variable is optional but required if you want to create Hive metastore database in a custom location other than default location.

In the Windows search bar, start typing “environment variables” and select the first match which opens up **System Properties** dialog. On the **System Properties** window, press **Environment Variables** button.

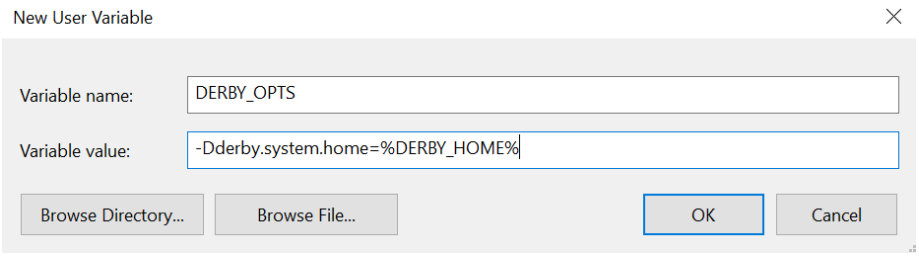
In the **Environment Variables** dialog, click on **New** under **User variables** section.



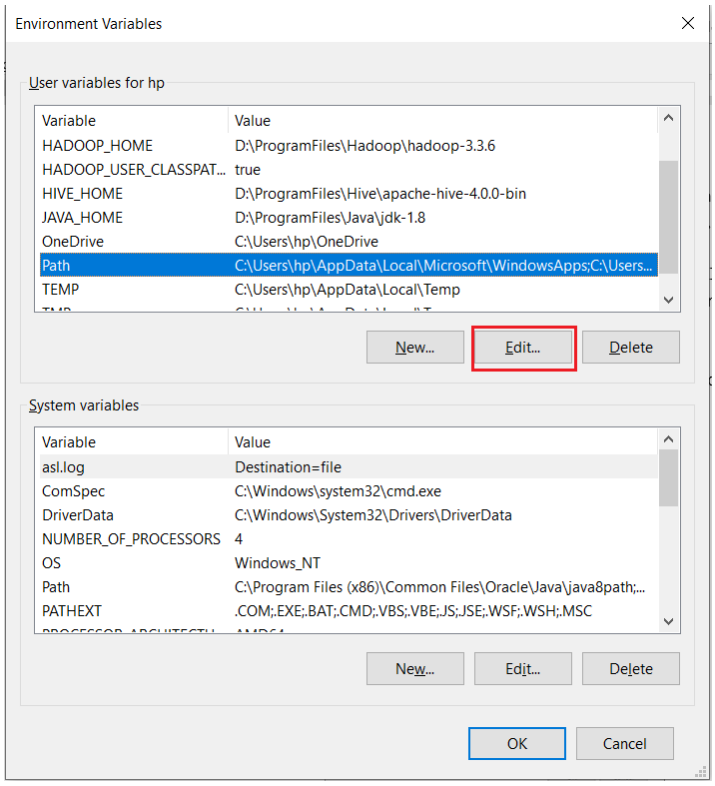
Add `DERBY_HOME` variable and press OK.



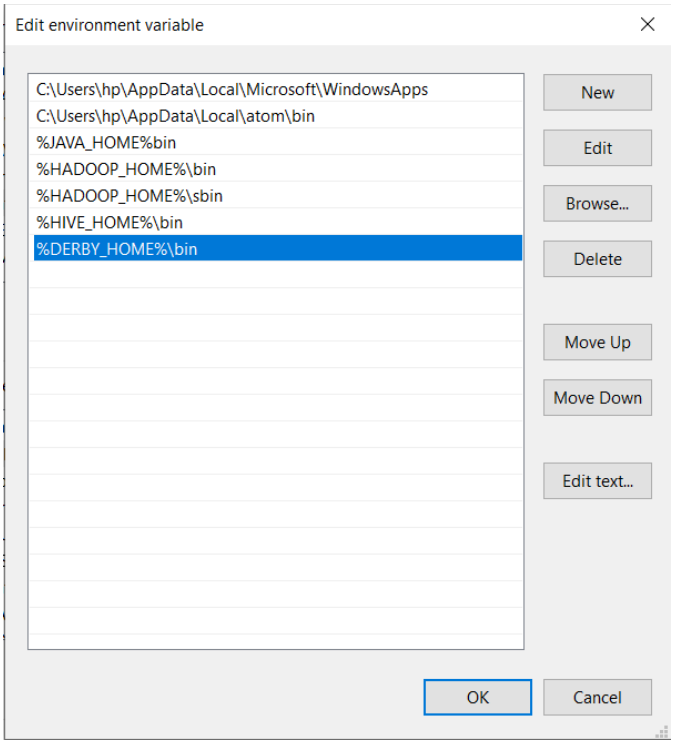
Click on **New** again and add `DERBY_OPTS` variable and press OK.



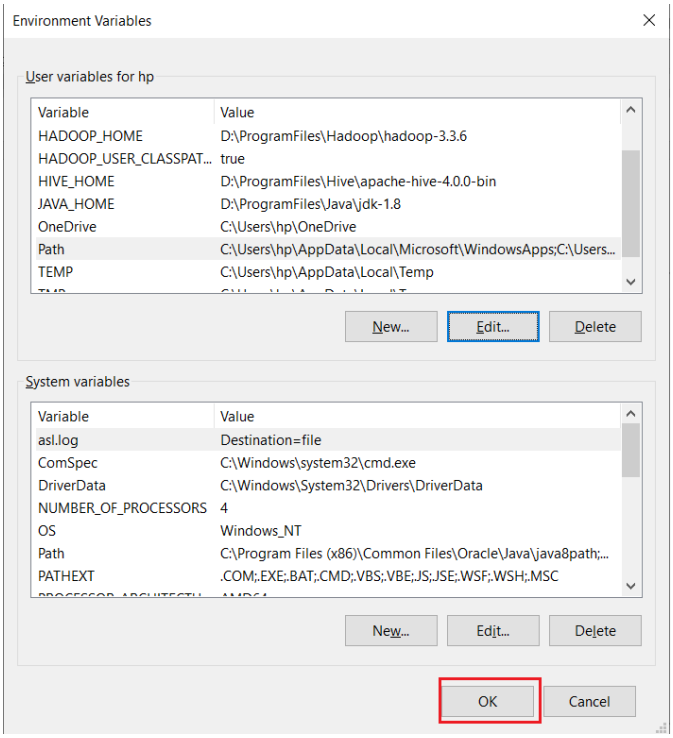
Select `PATH` variable and press **Edit** button



Press **New** and add `%DERBY_HOME%\bin` value and press OK.



Press OK to apply environment variable changes and close window.

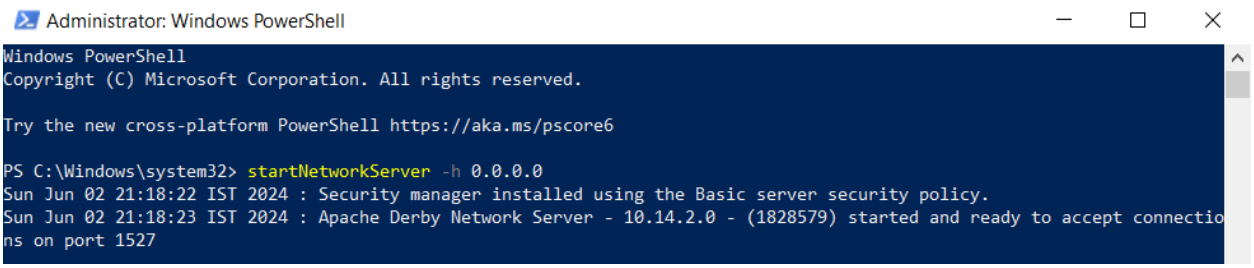


8.3. Start Derby Network Server:

Now, start the Derby network server on the local host.

Open **Command Prompt** or Windows PowerShell in **Administrator** mode and run the following command

```
startNetworkServer -h 0.0.0.0
```



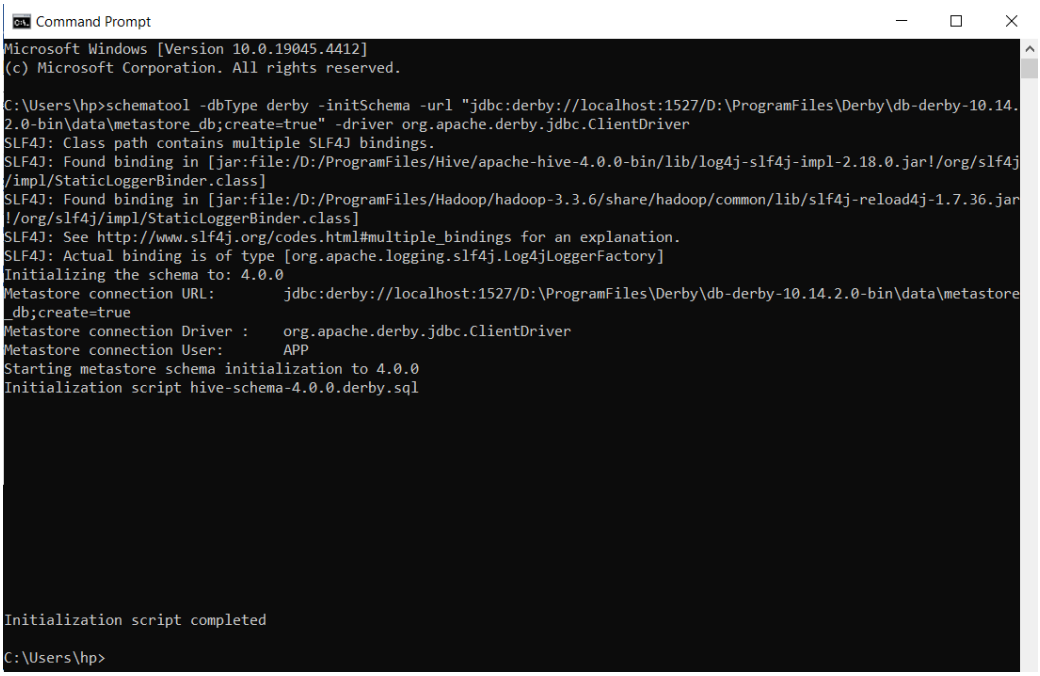
Apache Derby Network server runs on **1527** port by default

8.4. Initialize Local Metastore:

Before starting Hive, we need to initialize Hive Metastore using `schematool` utility to create metastore database.

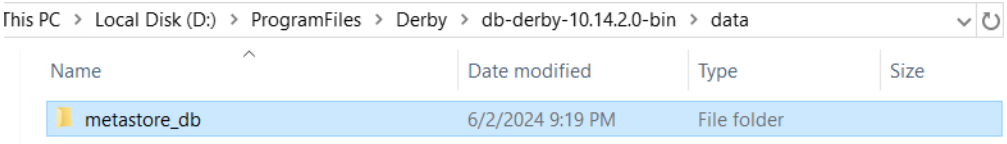
Run the following `schematool` command which connects to Derby Network Server and creates `metastore_db` in `DERBY_HOME\data` location *(i.e. `D:\ProgramFiles\Derby\db-derby-10.14.2.0-bin\data` location)*

```
schematool -dbType derby -initSchema -url
"jdbc:derby://localhost:1527/D:\ProgramFiles\Derby\db-derby-10.14.2.0-
bin\data\metastore_db;create=true" -driver org.apache.derby.jdbc.ClientDriver
```



Note: If you encounter Exception in thread "main" java.lang.NoSuchMethodError: org.fusesource.jansi.AnsiConsole.wrapOutputStream(Ljava/io/OutputStream;)Ljava/io/OutputStream error, download `jansi-1.18.jar` file from the official Jansi Download website and copy to `HIVE_HOME\lib` directory.

After executing the above command, it creates `metastore_db` folder in `DERBY_HOME\data` location.

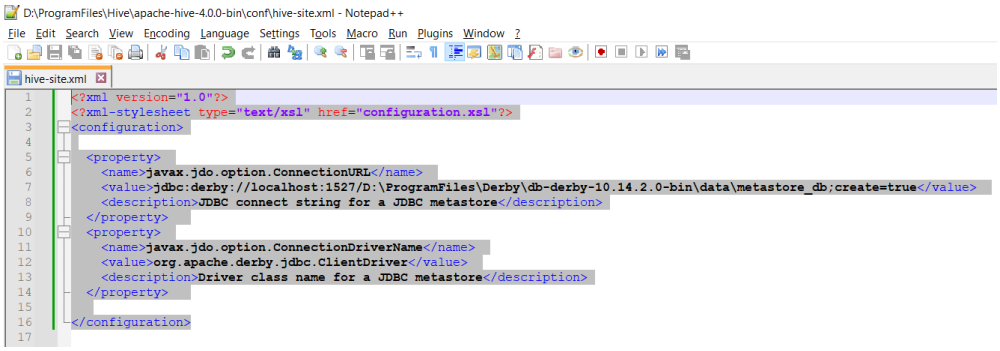


8.5. Configure Hive Site:

Now, add Derby Network database configuration settings in `hive-site.xml` file for Hive to connect.

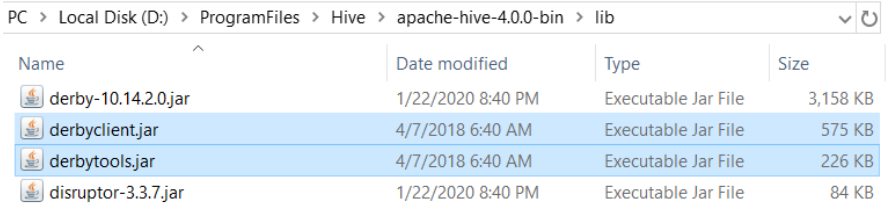
Go to `%HIVE_HOME%\conf` directory, create a file named `hive-site.xml` and paste the following code in the file. This code is referring to `metastore_db` in `DERBY_HOME\data` location.

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<configuration>

  <property>
        <name>javax.jdo.option.ConnectionURL</name>
        <value>jdbc:derby://localhost:1527/D:\ProgramFiles\Derby\db-derby-
10.14.2.0-bin\data\metastore_db;create=true</value>
        <description>JDBC connect string for a JDBC metastore</description>
  </property>
  <property>
        <name>javax.jdo.option.ConnectionDriverName</name>
        <value>org.apache.derby.jdbc.ClientDriver</value>
        <description>Driver class name for a JDBC metastore</description>
  </property>

</configuration>
```



8.6. Copy Derby Libraries:

For Hive to communitate with local Derby, we need to copy `derbyclient.jar` and `derbytools.jar` files from `%DERBY_HOME%\lib` directory and paste them into `%HIVE_HOME%\lib` directory.



8.7. Run Beeline CLI:

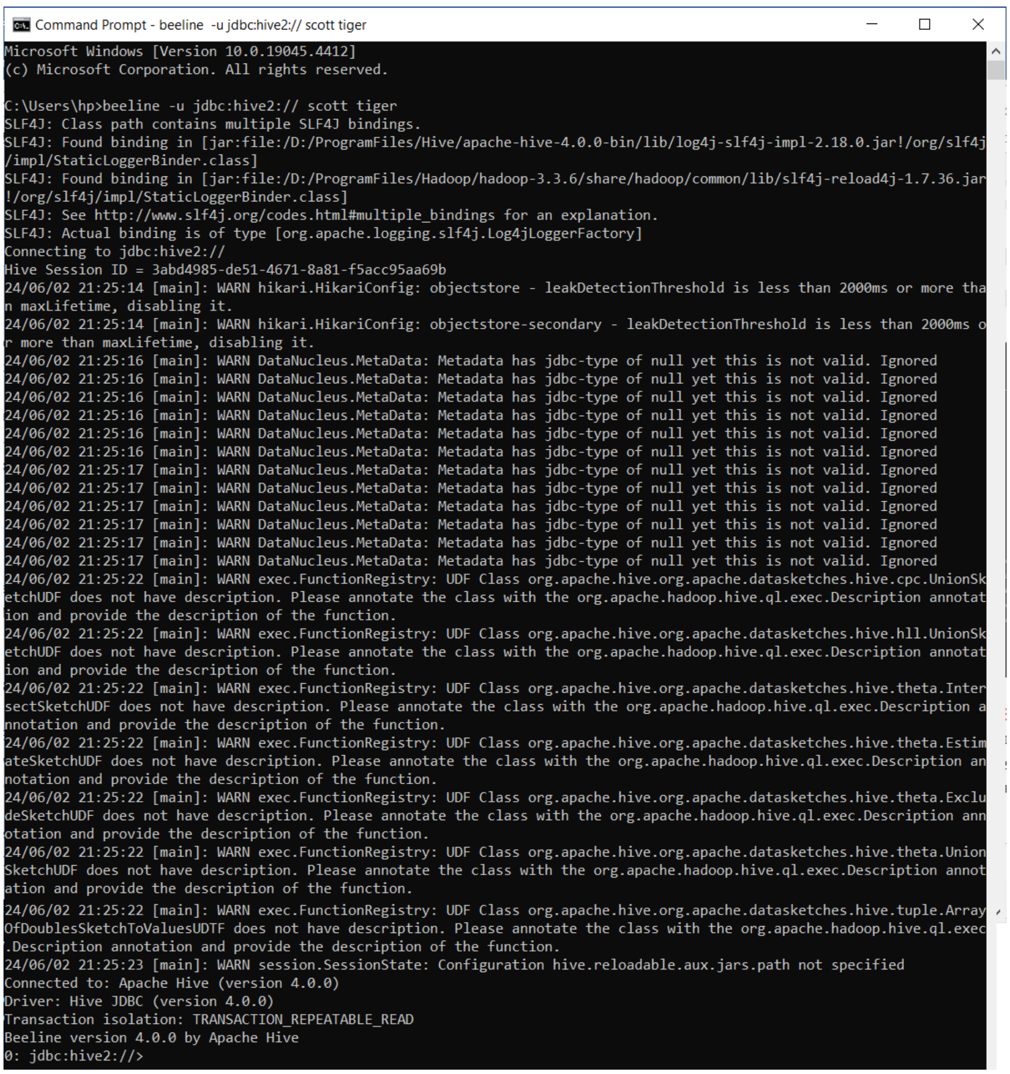
The Beeline works in embedded mode and remote mode. In embedded mode, Beeline connects to an embedded HiveServer2 and in remote mode, it connects to HiverServer2 service over Thrift. Using Beeline, we can connect to HiveServer2 running on Local or Remote server using hostname and port.

To start Beeline in embedded mode, open command prompt and run the following command.

```
beeline -u jdbc:hive2:// -n scott -p tiger
```

or

```
beeline -u jdbc:hive2:// scott tiger
```

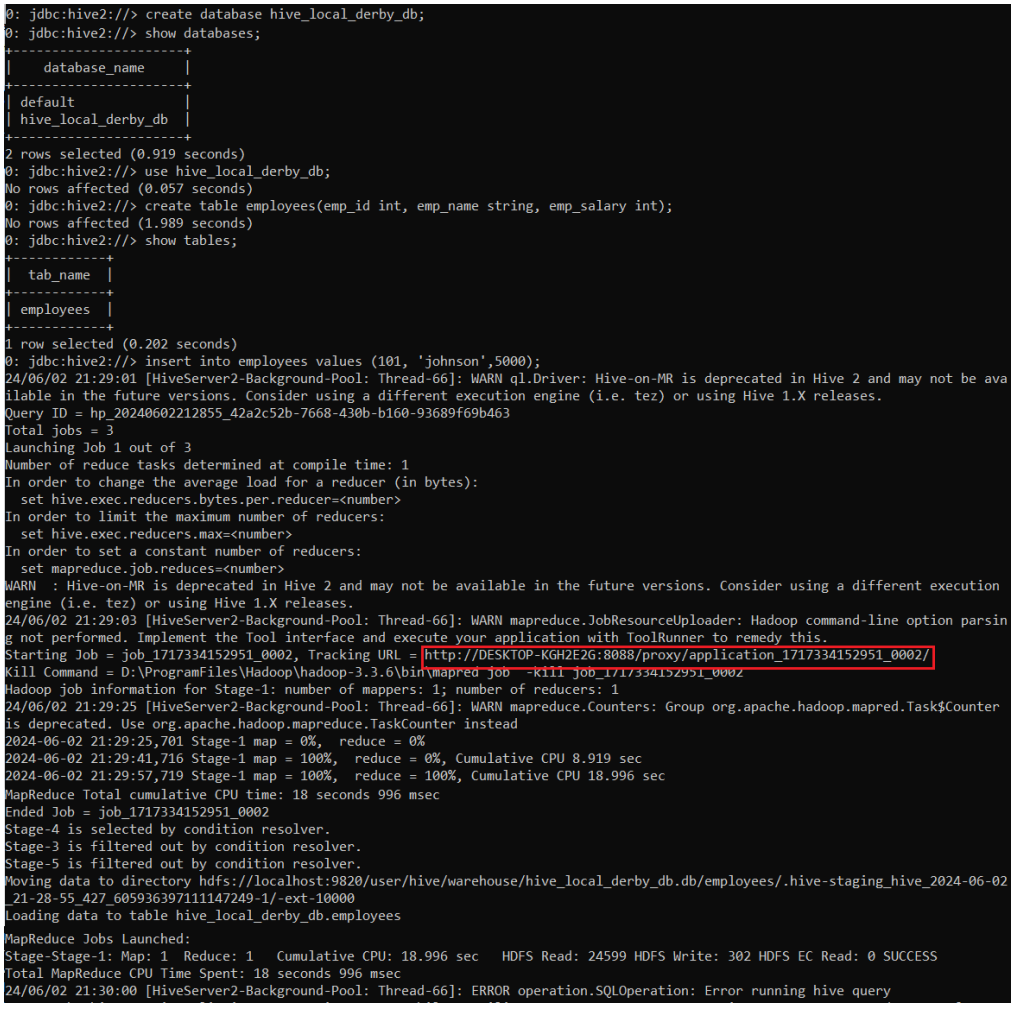


Note: If you encounter "*org.apache.hadoop.hive.metastore.api.MetaException Version information not found in metastore*" error, then make sure `metastore_db` is initialized properly. If not done, stop the Derby Network server, delete the metastore database, reinitialize it using `schematool` utility, start Derby Network server and then start beeline.

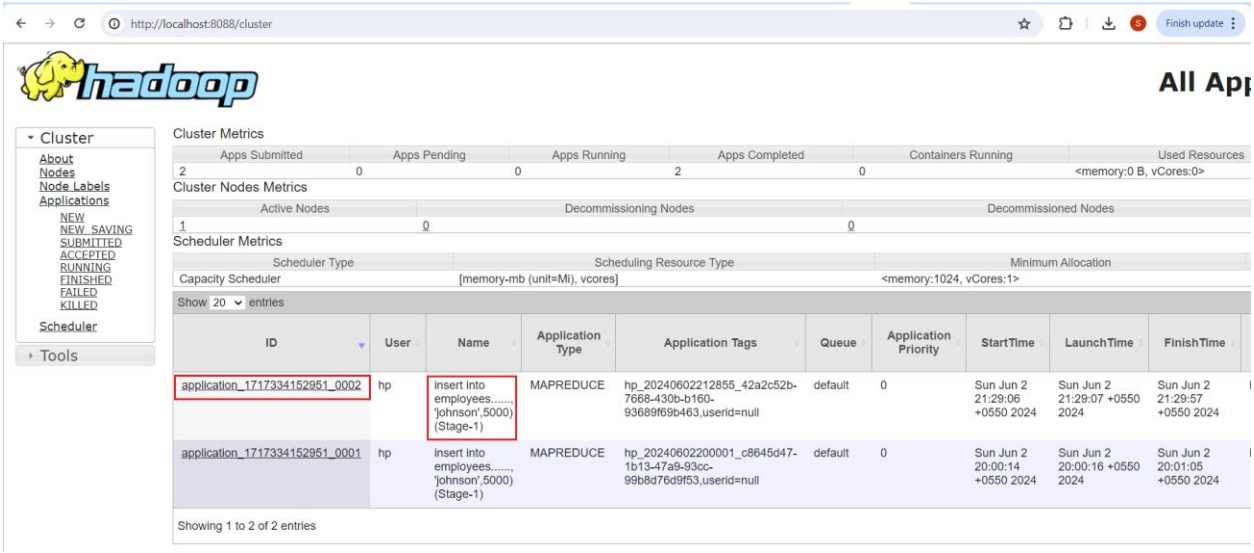
After beeline is connected, run following queries to create database, table and insert data.

```
create database hive_local_derby_db;
show databases;
use hive_local_derby_db;
create table employees(emp_id int, emp_name string, emp_salary int);
show tables;
insert into employees values (101, 'johnson',5000);
```

The above insert statement submits MapReduce job

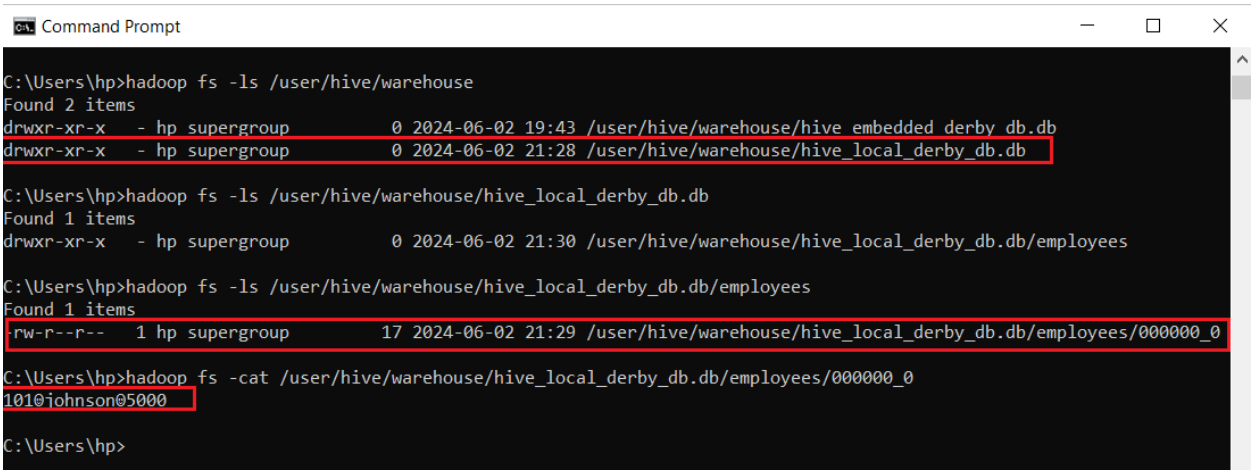


We can track the MapReduce job in YARN UI: http://localhost:8088/cluster
In YARN UI, you can see an application name with `insert into employees…` that was executed.



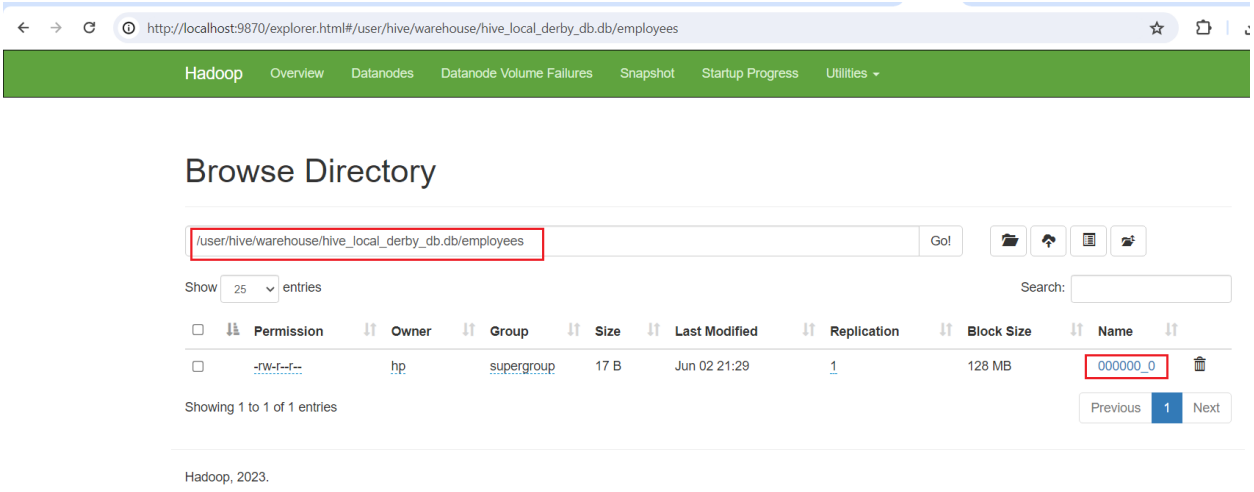
After the above insert query is completed, we can verify the output in HDFS using the following commands:

```
hadoop fs -ls /user/hive/warehouse
hadoop fs -ls /user/hive/warehouse/hive_local_derby_db.db
hadoop fs -ls /user/hive/warehouse/hive_local_derby_db.db/employees
hadoop fs -cat /user/hive/warehouse/hive_local_derby_db.db/employees/000000_0
```



The same is visible in NameNode UI: http://localhost:9870/dfshealth.html also.

Open NameNode UI, go to **Utilities** tab and select **Browse the file system** option. Enter the directory name `/user/hive/warehouse` and you can see `hive_local_derby_db.db` folder available. Click on `hive_local_derby_db.db` folder to see `employees` folder in which `000000_0` file available. Click on the file and select **Head the file** or **Tail the file** to see the file contents. We can download this file by clicking on **Download** option.



Since we enabled Derby to run as Network server, we should be able to make multiple Hive connections. To test this, open another Command Prompt and start Hive shell and try to run a sample query (For example, `show databases;`).

```
beeline -u jdbc:hive2:// scott tiger
```

Command Prompt - beeline -u jdbc:hive2:// scott tiger

```
Microsoft Windows [Version 10.0.19045.4412]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>beeline -u jdbc:hive2:// scott tiger
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/D:/ProgramFiles/Hive/apache-hive-4.0.0-bin/lib/log4j-slf4j-impl-2.18.0.jar!/org/slf4j
/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/D:/ProgramFiles/Hadoop/hadoop-3.3.6/share/hadoop/common/lib/slf4j-reload4j-1.7.36.jar
!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Connecting to jdbc:hive2://
Hive Session ID = 43e9f936-6148-4cbd-a93c-d6be30cb2e3c
24/06/02 21:44:01 [main]: WARN hikari.HikariConfig: objectstore - leakDetectionThreshold is less than 2000ms or more tha
n maxLifetime, disabling it.
24/06/02 21:44:02 [main]: WARN hikari.HikariConfig: objectstore-secondary - leakDetectionThreshold is less than 2000ms o
r more than maxLifetime, disabling it.
24/06/02 21:44:03 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:03 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:03 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:03 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:03 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:03 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:04 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:04 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:04 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:04 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:04 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:04 [main]: WARN DataNucleus.MetaData: Metadata has jdbc-type of null yet this is not valid. Ignored
24/06/02 21:44:08 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.cpc.UnionSk
etchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description annotat
ion and provide the description of the function.
24/06/02 21:44:08 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.hll.UnionSk
etchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description annotat
ion and provide the description of the function.
24/06/02 21:44:08 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.theta.Inter
sectSketchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description a
nnotation and provide the description of the function.
24/06/02 21:44:08 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.theta.Estim
ateSketchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description an
notation and provide the description of the function.
24/06/02 21:44:08 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.theta.Exclu
deSketchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description ann
otation and provide the description of the function.
24/06/02 21:44:08 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.theta.Union
SketchUDF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec.Description annot
ation and provide the description of the function.
24/06/02 21:44:08 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.tuple.Array
OfDoublesSketchToValuesUDTF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec
24/06/02 21:44:08 [main]: WARN exec.FunctionRegistry: UDF Class org.apache.hive.org.apache.datasketches.hive.tuple.Array
OfDoublesSketchToValuesUDTF does not have description. Please annotate the class with the org.apache.hadoop.hive.ql.exec
.Description annotation and provide the description of the function.
24/06/02 21:44:09 [main]: WARN session.SessionState: Configuration hive.reloadable.aux.jars.path not specified
Connected to: Apache Hive (version 4.0.0)
Driver: Hive JDBC (version 4.0.0)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 4.0.0 by Apache Hive
0: jdbc:hive2://> show databases;
+---------------------+
|    database_name    |
+---------------------+
| default             |
| hive_local_derby_db |
+---------------------+
2 rows selected (3.72 seconds)
0: jdbc:hive2://>
```

We are able to access metastore database without any issues while other Beeline session is still active and accessing the same database.

To come out of `beeline>` shell, use `!quit` command.

8.8. Start HiveServer2 Service:

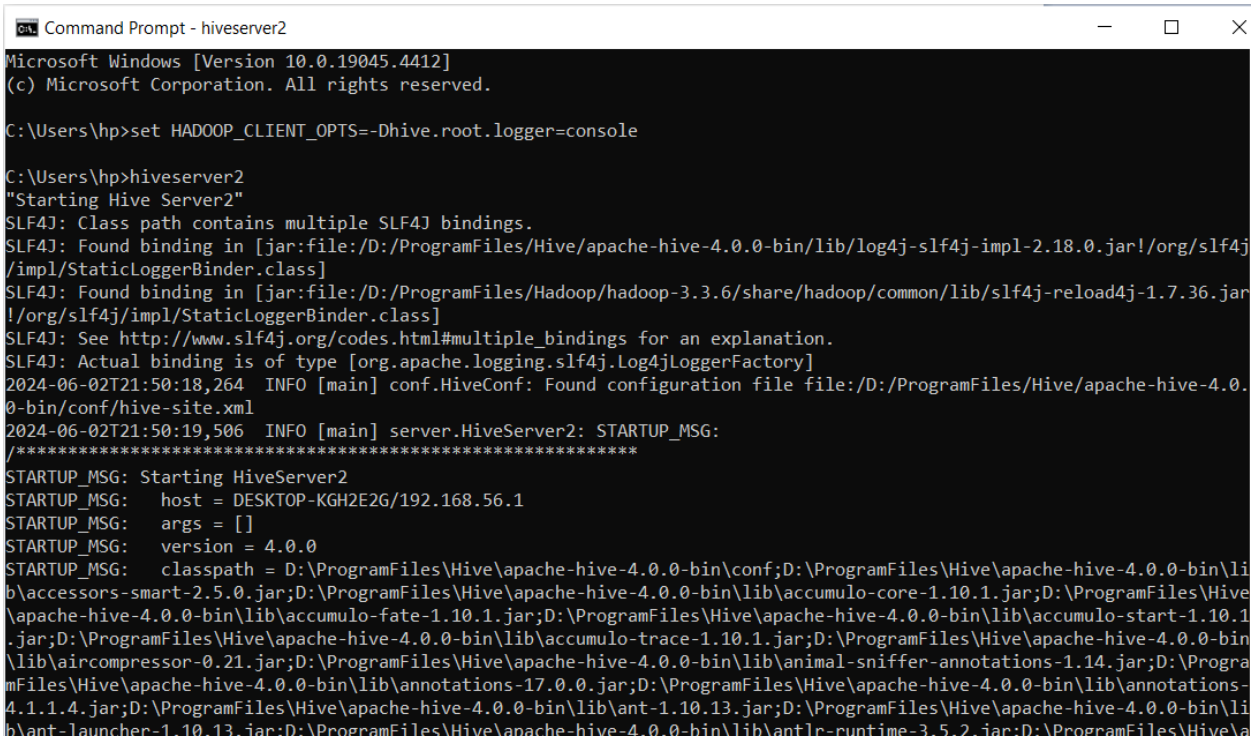
Before starting Beeline in remote mode that connects to HiveServer2, make sure the HiveServer2 service is running.

Open command prompt and start the HiveServer2 service using the following command:

```
hiveserver2
```

If you wish to see more logging of hiveserver2, then you need to set `HADOOP_CLIENT_OPTS` variable to display logging to console and trigger `hiveserver2` as below:

```
set HADOOP_CLIENT_OPTS=-Dhive.root.logger=console
hiveserver2
```



```
2024-06-02T21:50:34,461  INFO [main] http.HttpServer: ASYNC_PROFILER_HOME env or -Dasync.profiler.home not specified. Di
sabling /prof endpoint..
2024-06-02T21:50:34,474  INFO [main] service.AbstractService: Service:OperationManager is started.
2024-06-02T21:50:34,475  INFO [main] service.AbstractService: Service:SessionManager is started.
2024-06-02T21:50:34,478  INFO [main] service.AbstractService: Service:CLIService is started.
2024-06-02T21:50:34,478  INFO [main] service.AbstractService: Service:ThriftBinaryCLIService is started.
2024-06-02T21:50:34,893  INFO [main] thrift.ThriftCLIService: Starting ThriftBinaryCLIService on port 10000 with 5...500
 worker threads
2024-06-02T21:50:34,893  INFO [main] service.AbstractService: Service:HiveServer2 is started.
2024-06-02T21:50:34,901  INFO [main] server.Server: jetty-9.4.45.v20220203; built: 2022-02-03T09:14:34.105Z; git: 4a0c91
c0be53805e3fcffdcdcc9587d5301863db; jvm 1.8.0_411-b09
2024-06-02T21:50:35,334  INFO [main] server.session: DefaultSessionIdManager workerName=node0
2024-06-02T21:50:35,334  INFO [main] server.session: No SessionScavenger set, using defaults
2024-06-02T21:50:35,340  INFO [main] server.session: node0 Scavenging every 660000ms
2024-06-02T21:50:35,508  INFO [main] handler.ContextHandler: Started o.e.j.w.WebAppContext@18b58c77{hiveserver2,/,file:/
//C:/Users/hp/AppData/Local/Temp/jetty-0_0_0_0-10002-hive-service-4_0_0_jar-_-any-6173310191469174998/webapp/,AVAILABLE}
{jar:file:/D:/ProgramFiles/Hive/apache-hive-4.0.0-bin/lib/hive-service-4.0.0.jar!/hive-webapps/hiveserver2}
2024-06-02T21:50:35,510  INFO [main] handler.ContextHandler: Started o.e.j.s.ServletContextHandler@36cf6377{static,/stat
ic,jar:file:/D:/ProgramFiles/Hive/apache-hive-4.0.0-bin/lib/hive-service-4.0.0.jar!/hive-webapps/static,AVAILABLE}
2024-06-02T21:50:35,540  INFO [main] server.AbstractConnector: Started ServerConnector@4ae5ddc0{HTTP/1.1, (http/1.1)}{0.
0.0.0:10002}
2024-06-02T21:50:35,540  INFO [main] server.Server: Started @23259ms
2024-06-02T21:50:35,540  INFO [main] server.HiveServer2: Web UI has started on port 10002
2024-06-02T21:50:35,540  INFO [main] http.HttpServer: Started HttpServer[hiveserver2] on port 10002
2024-06-02T21:51:33,796  INFO [Scheduled Query Poller] HiveMetaStore.audit: ugi=hp      ip=unknown-ip-addr      cmd=sche
duled_query_poll
2024-06-02T21:51:34,061  INFO [NotificationEventPoll 0] metastore.HiveMetaStoreClient: HMS client filtering is enabled.
```

We can see that `HiveServer2` service started on port `10002` while `ThriftBinaryCLIService` started on port `10000`.

8.9. Beeline Remote Connection:

In Remote mode, Beeline connects to HiveServer2 over Thrift CLI servers that runs on 10000 port. Beeline remote connection can be made as **anonymous** user or with **specific credentials**.

Use the following command to connect Beeline as anonymous user:

```
beeline -u jdbc:hive2://localhost:10000/
```

```
C:\Users\hp>beeline -u jdbc:hive2://localhost:10000/
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/D:/ProgramFiles/Hive/apache-hive-4.0.0-bin/lib/log4j-slf4j-impl-2.18.0.jar!/org/slf4j
/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/D:/ProgramFiles/Hadoop/hadoop-3.3.6/share/hadoop/common/lib/slf4j-reload4j-1.7.36.jar
!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Connecting to jdbc:hive2://localhost:10000/
24/06/02 21:55:21 [main]: WARN jdbc.HiveConnection: Failed to connect to localhost:10000
Error: Could not open client transport with JDBC Uri: jdbc:hive2://localhost:10000/: Failed to open new session: java.la
ng.RuntimeException: org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.security.authorize.AuthorizationException):
 User: hp is not allowed to impersonate anonymous (state=08S01,code=0)
```

Here, we see an error that ”AuthorizationException: User xxx is not allowed to impersonate anonymous”. This is because HiveServer2 does not allow impersonation by default.

Use the following command to connect Beeline with default user `scott`:

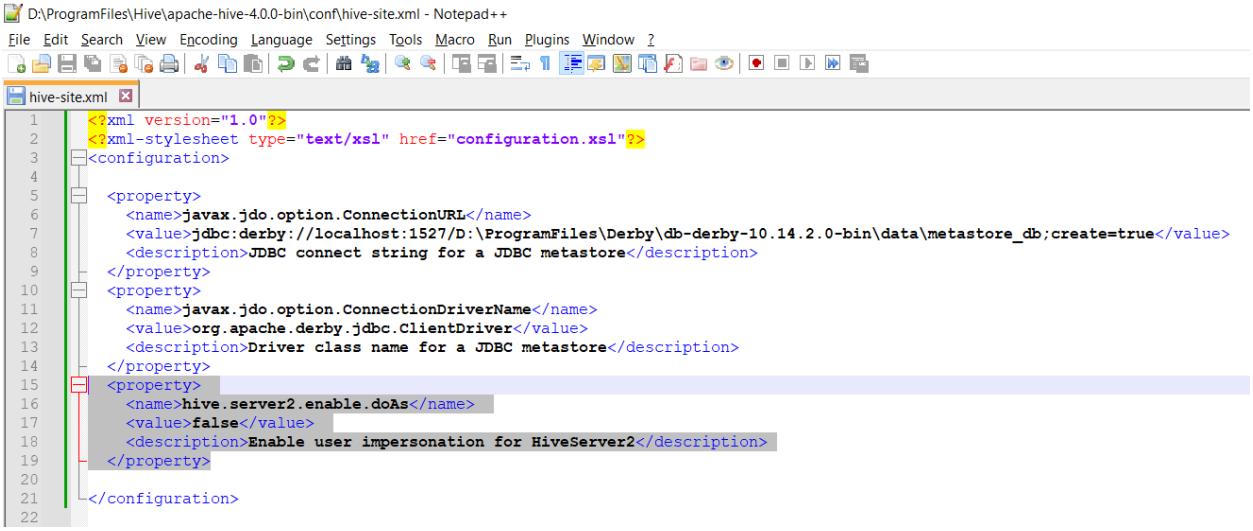
```
beeline -u jdbc:hive2://localhost:10000/ -n scott -p tiger
```

```
C:\Users\hp>beeline -u jdbc:hive2://localhost:10000/ -n scott -p tiger
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/D:/ProgramFiles/Hive/apache-hive-4.0.0-bin/lib/log4j-slf4j-impl-2.18.0.jar!/org/slf4j
/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/D:/ProgramFiles/Hadoop/hadoop-3.3.6/share/hadoop/common/lib/slf4j-reload4j-1.7.36.jar
!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Connecting to jdbc:hive2://localhost:10000/
24/06/02 21:56:11 [main]: WARN jdbc.HiveConnection: Failed to connect to localhost:10000
Error: Could not open client transport with JDBC Uri: jdbc:hive2://localhost:10000/: Failed to open new session: java.la
ng.RuntimeException: org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.security.authorize.AuthorizationException):
 User: hp is not allowed to impersonate scott (state=08S01,code=0)

C:\Users\hp>
```

Here, we see an error that "AuthorizationException: User xxx is not allowed to impersonate anonymous". This is because HiveServer2 does not allow impersonation by default.

To fix the above error, open `hive_site.xml` file in `HIVE_HOME\conf` location and add the following property between `<configuration>` and `<configuration>` tag.

```
<property>
      <name>hive.server2.enable.doAs</name>
      <value>false</value>
      <description>Enable user impersonation for HiveServer2</description>
</property>
```



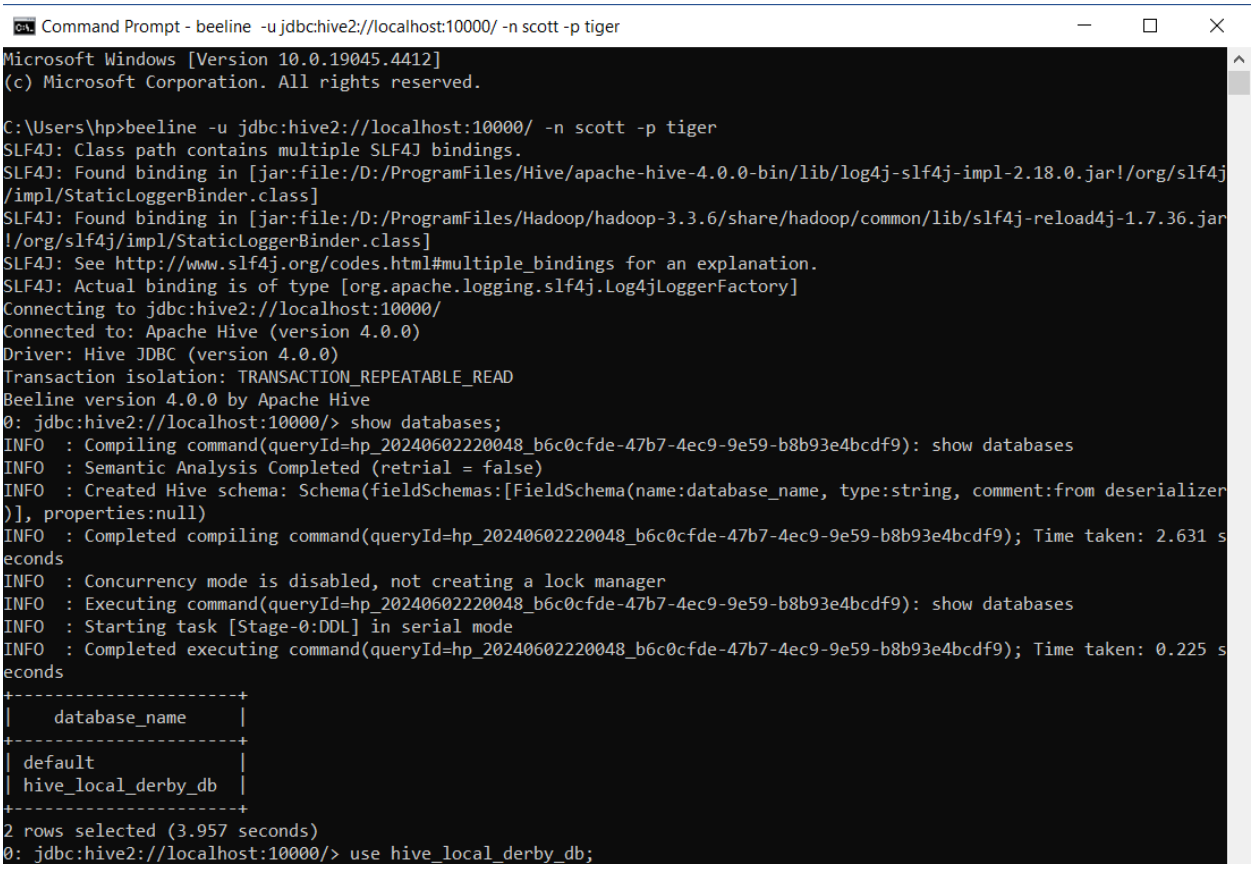
Stop and restart HiveServer2 service using the following command

```
set HADOOP_CLIENT_OPTS=-Dhive.root.logger=console
hiveserver2
```

Now, we are able to connect to Beeline in remote mode and run queries without any issue.

```
beeline -u jdbc:hive2://localhost:10000/ -n scott -p tiger
```

```
show databases;
use hive_local_derby_db;
show tables;
select * from employees;
```



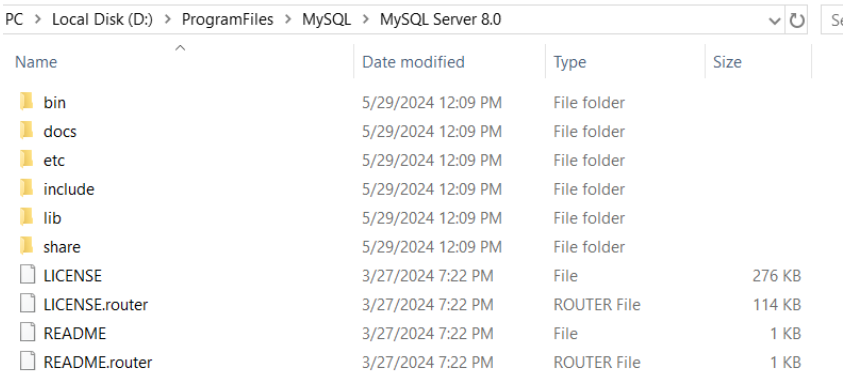
9. Configure Remote MySQL Metastore:

Now, we will see how to use MySQL as the remote metastore database.

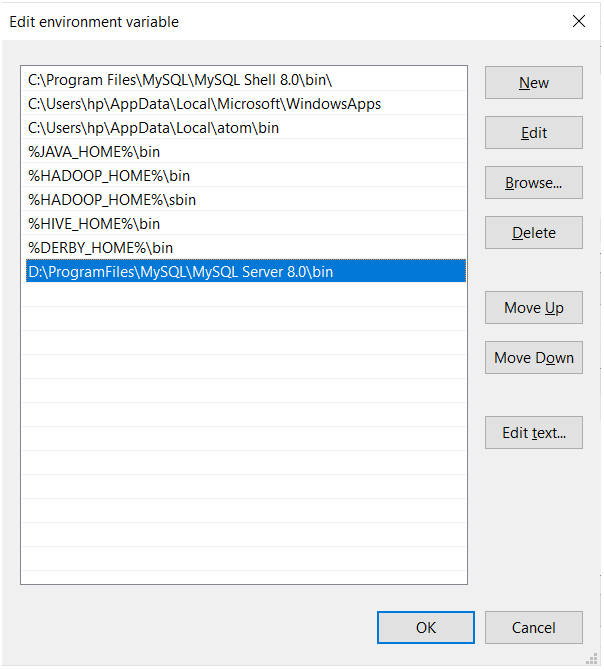
9.1. Install MySQL Server:

First, install MySQL server from the official MySQL Downloads website in the machine if was not already installed *(I installed the latest `MySQL Server 8.0` version to `D:\ProgramFiles\MySQL` directory)*

Note that MySQL runs on port **3306** by default.



Open the **Environment Variables** dialog, select `PATH` variable under **User variables** section and press **Edit** button. Press **New** and add `MySQL Install Path\bin` value *(for example, `D:\ProgramFiles\MySQL\MySQL Server 8.0\bin`)* and press OK.
Then press OK again to apply environment variable changes and close window.

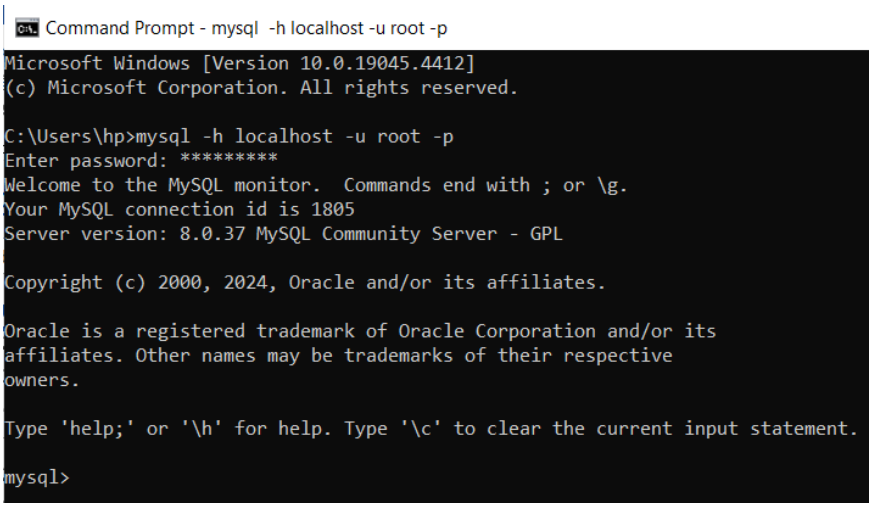


9.2. Create Metastore DB in MySQL:

In MySQL Server, create a database for Hive metastore.

Open MySQL command prompt or Workbench.
To launch `mysql>` command prompt with `root` user, execute this command:

```
mysql -h localhost -u root -p
```

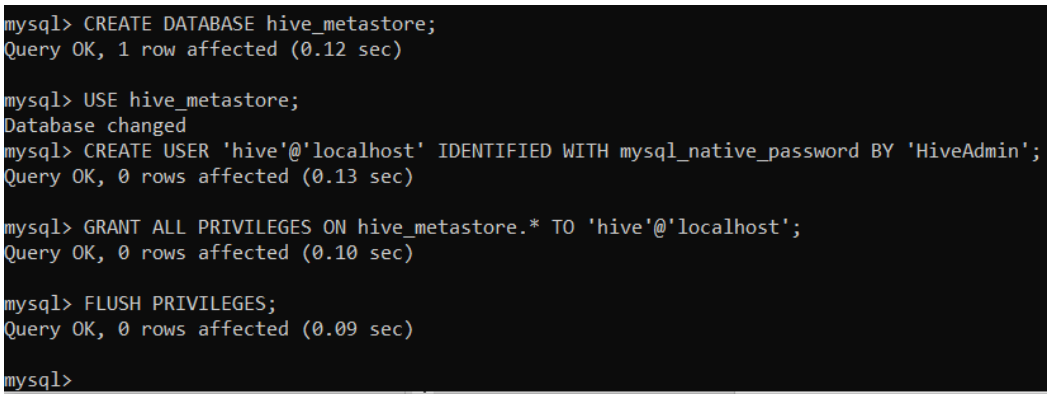


Run the following queries to create a database named `hive_metastore` and a new user for Hive named `hive` and password as `HiveAdmin` in MySQL server.

```
CREATE DATABASE hive_metastore;
USE hive_metastore;
CREATE USER 'hive'@'localhost' IDENTIFIED WITH mysql_native_password BY 'HiveAdmin';
GRANT ALL PRIVILEGES ON hive_metastore.* TO 'hive'@'localhost';
FLUSH PRIVILEGES;
```

Note:
Make sure that `hive_metastore` database and `hive` user are not already available in your MySQL server. If they are already available, use a different database name and user name to create. Otherwise, you may encounter below errors:

ERROR 1007 (HY000): Can't create database 'hive_metastore'; database exists
ERROR 1396 (HY000): Operation CREATE USER failed for 'hive'@'localhost'

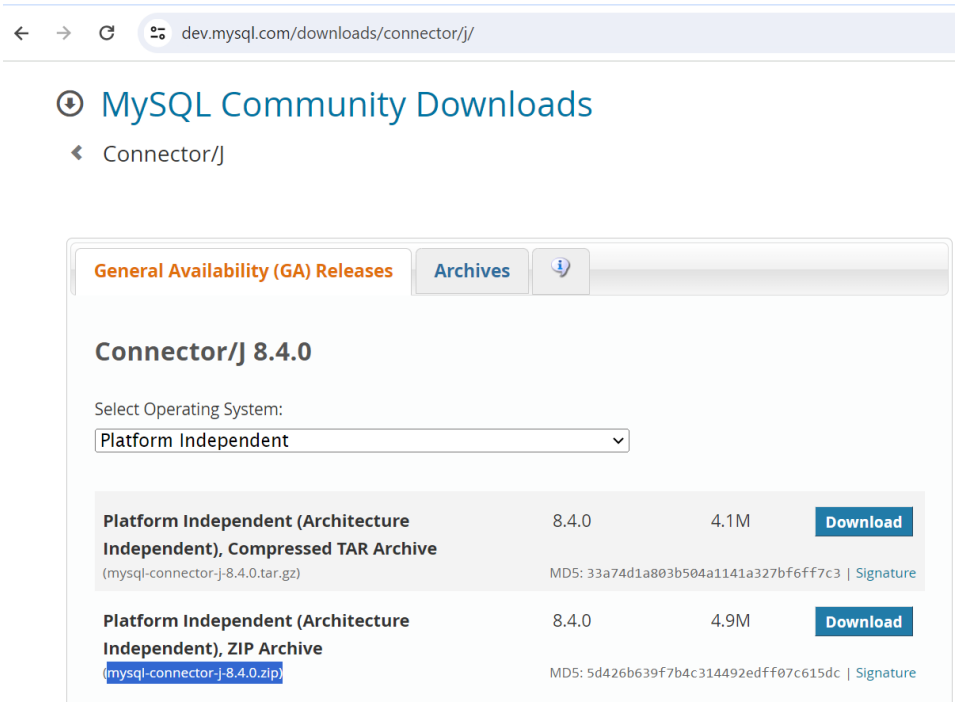


9.3. Download MySQL JDBC Driver:

Since Hive does not provide the JDBC driver for MySQL by default, we need to explicitly get the driver and place it in `HIVE_HOME\lib` directory

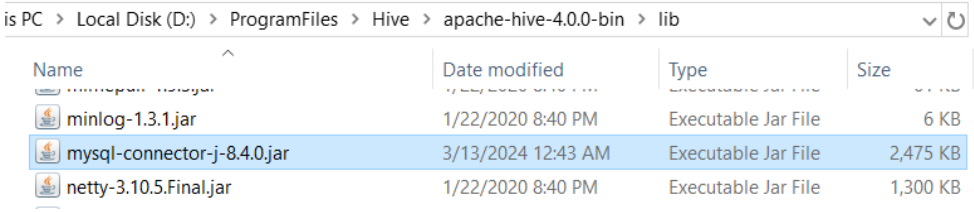
Download **JDBC Driver for MySQL (Connector/J)** from the below link
https://dev.mysql.com/downloads/connector/j/

In the above link, choose **Platform Independent** Operating System and download **`mysql-connector-j-*.zip`** file as shown below:



The latest version of zip file at the time of writing this document is **`mysql-connector-j-8.4.0.zip`**. You can download the file of whichever latest version that you could see.

After downloading the `zip` file, unzip it which creates `mysql-connector-j-*` directory. Open the directory and copy `mysql-connector-j-*.jar` file to `HIVE_HOME\lib` directory.



9.4. Configure Hive Site File:

Now, we need to configure Metastore service to communicate with MySQL database. Open `hive-site.xml` file in `HIVE_HOME\conf` directory and replace the existing properties with the following properties between `<configuration>` and `</configuration` element.

```
  <property>
        <name>javax.jdo.option.ConnectionURL</name>
        <value>jdbc:mysql://localhost:3306/hive_metastore</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>hive</value>
    <description>Username to use against metastore database</description>
  </property>
  <property>
        <name>javax.jdo.option.ConnectionPassword</name>
        <value>HiveAdmin</value>
        <description>password to use against metastore database</description>
  </property>
  <property>
        <name>hive.metastore.uris</name>
        <value>thrift://127.0.0.1:9083</value>
        <description>Thrift URI for the remote metastore. Used by metastore
client to connect to remote metastore</description>
  </property>
  <property>
        <name>hive.server2.enable.doAs</name>
        <value>false</value>
        <description>Enable user impersonation for HiveServer2</description>
  </property>
```

D:\ProgramFiles\Hive\apache-hive-4.0.0-bin\conf\hive-site.xml - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

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://localhost:3306/hive_metastore</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>hive</value>
    <description>Username to use against metastore database</description>
  </property>
  <property>
    <name>javax.jdo.option.ConnectionPassword</name>
    <value>HiveAdmin</value>
    <description>password to use against metastore database</description>
  </property>
  <property>
    <name>hive.metastore.uris</name>
    <value>thrift://127.0.0.1:9083</value>
    <description>Thrift URI for the remote metastore. Used by metastore client to connect to remote metastore</description>
  </property>
  <property>
    <name>hive.server2.enable.doAs</name>
    <value>false</value>
    <description>Enable user impersonation for HiveServer2</description>
  </property>

</configuration>
```

9.5. Initialize Metastore DB:

Now, run the `schematool` utility to create the initial DB structure in MySQL database using the following command

```
schematool -dbType mysql -initSchema
```

Command Prompt

```
Microsoft Windows [Version 10.0.19045.4412]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>schematool -dbType mysql -initSchema
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/D:/ProgramFiles/Hive/apache-hive-4.0.0-bin/lib/log4j-slf4j-impl-2.18.0.jar!/org/slf4j
/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/D:/ProgramFiles/Hadoop/hadoop-3.3.6/share/hadoop/common/lib/slf4j-reload4j-1.7.36.jar
!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Initializing the schema to: 4.0.0
Metastore connection URL:        jdbc:mysql://localhost:3306/hive_metastore
Metastore connection Driver :    com.mysql.jdbc.Driver
Metastore connection User:       hive
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The drive
r is automatically registered via the SPI and manual loading of the driver class is generally unnecessary.
Starting metastore schema initialization to 4.0.0
Initialization script hive-schema-4.0.0.mysql.sql









Initialization script completed

C:\Users\hp>
```

9.6. Verify Metastore in MySQL:

Let us connect to MySQL server and verify Hive metastore created under `hive_metastore` database.

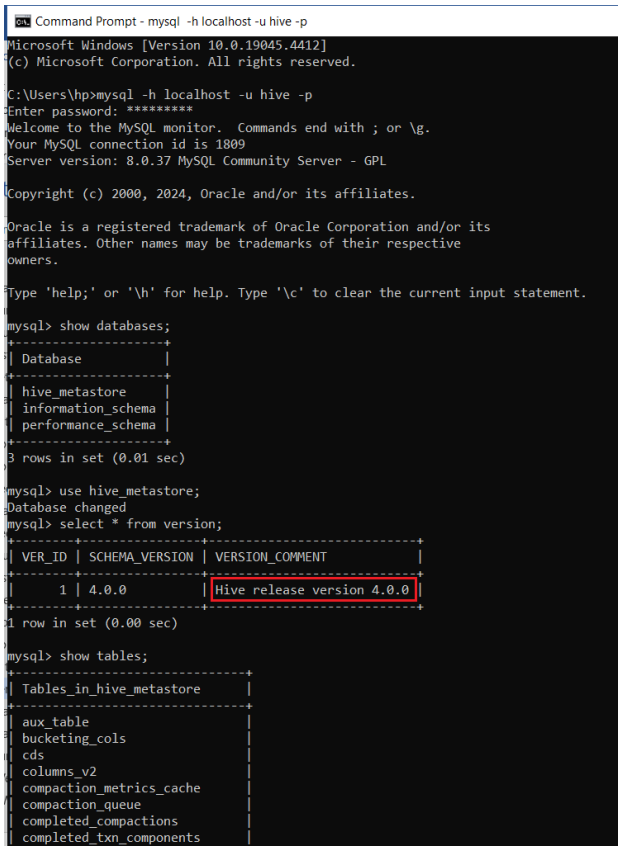
Launch MySQL command prompt using this command. Note that we are connecting with `hive` user credentials that we created earlier.

```
mysql -h localhost -u hive -p
```

Provide the password of `hive` user when asked.

In `mysql>` prompt, run the following queries:

```
show databases;
use hive_metastore;
select * from version;
show tables;
```

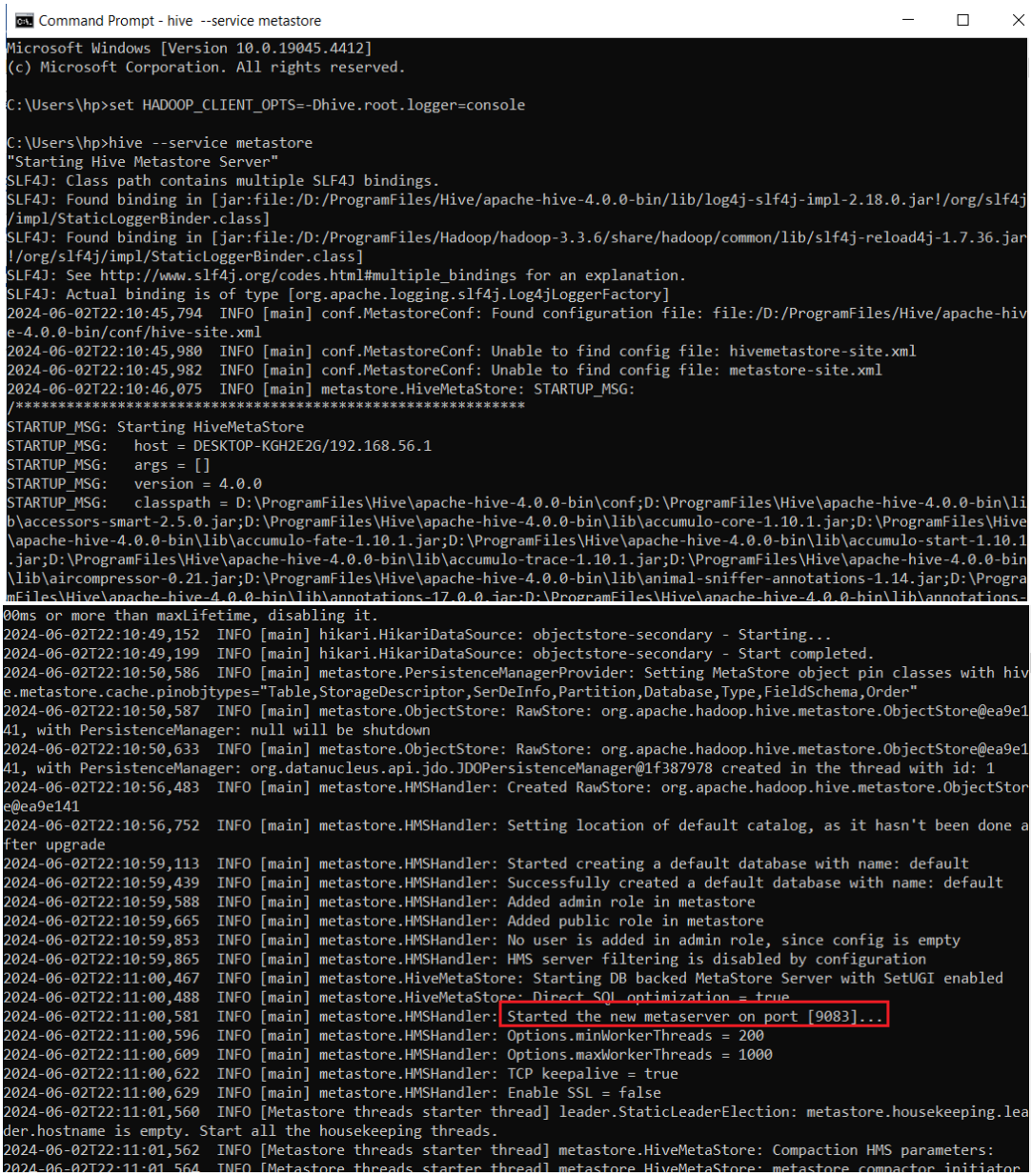


Here, we can see that Hive metastore has created tables and the version of Hive release **4.0.0**.

9.7. Start Hive Metastore service:

Hive will be able to connect to remote metastore in MySQL database using Thrift URIs.
So, let us start the `metastore` service using the below command to connect to our MySQL server.

```
set HADOOP_CLIENT_OPTS=-Dhive.root.logger=console
hive --service metastore
```



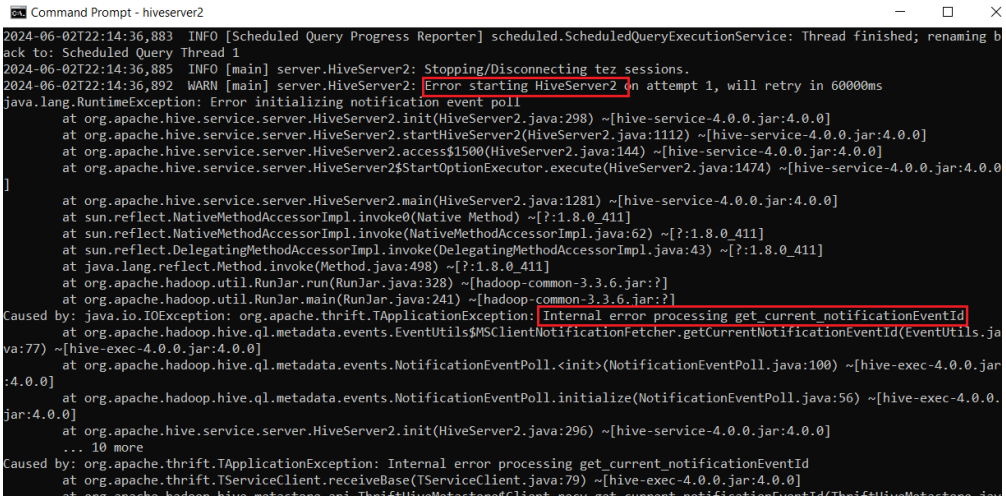
On the console, we can see that **metaserver** started on port **9083**

9.8. Start HiveServer2 Service:

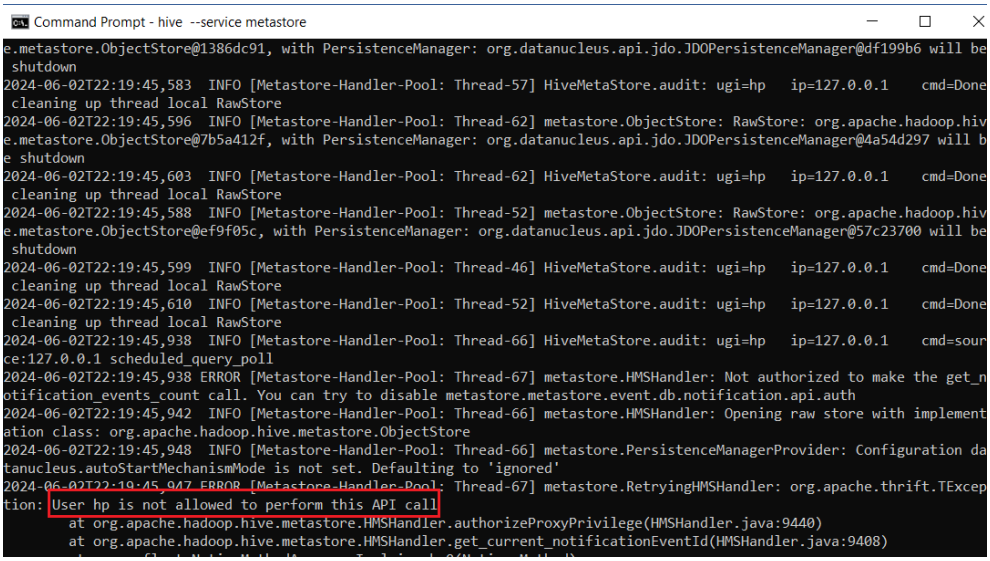
To start Beeline in remote mode, HiveServer2 service must be running. If HiveServer2 is already running, stop it and start it freshly because we made configuration changes in `hive_site.xml` that should be loaded into HiveServer2.

Open command prompt and start the HiveServer2 service using the following command

```
set HADOOP_CLIENT_OPTS=-Dhive.root.logger=console
hiveserver2
```



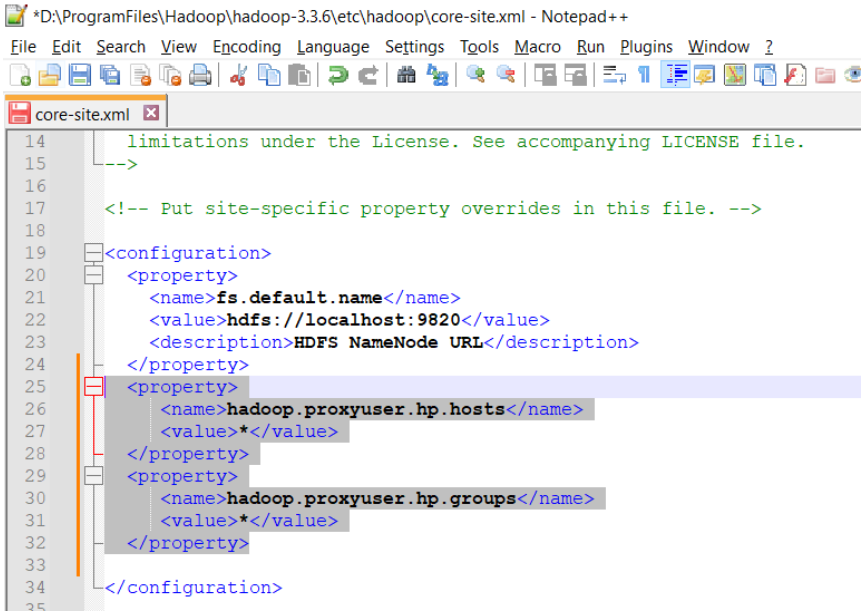
During HiveServer2 startup, we will encounter error "Internal error processing get_current_notificationEventId" and in the metastore logs *(go to the window where metastore service was started),* we can see an additional error "User xxx is not allowed to perform this API call".



Those errors are raised because that the superuser does not have access to impersonate as Hive user. To resolve this issue, we need to configure proxy user in Hadoop's `core-site.xml` file as below:

- Open `core-site.xml` file available in `%HADOOP_HOME%\etc\hadoop` location and add the following properties. In this configuration, replace `$superuser` with your user name displayed in the above error *(in my case, I am running Hadoop with `hp` user and the same is being displayed in the above error)*.

```
<property>
   <name>hadoop.proxyuser.$superuser.hosts</name>
   <value>*</value>
</property>
<property>
   <name>hadoop.proxyuser.$superuser.groups</name>
   <value>*</value>
</property>
```

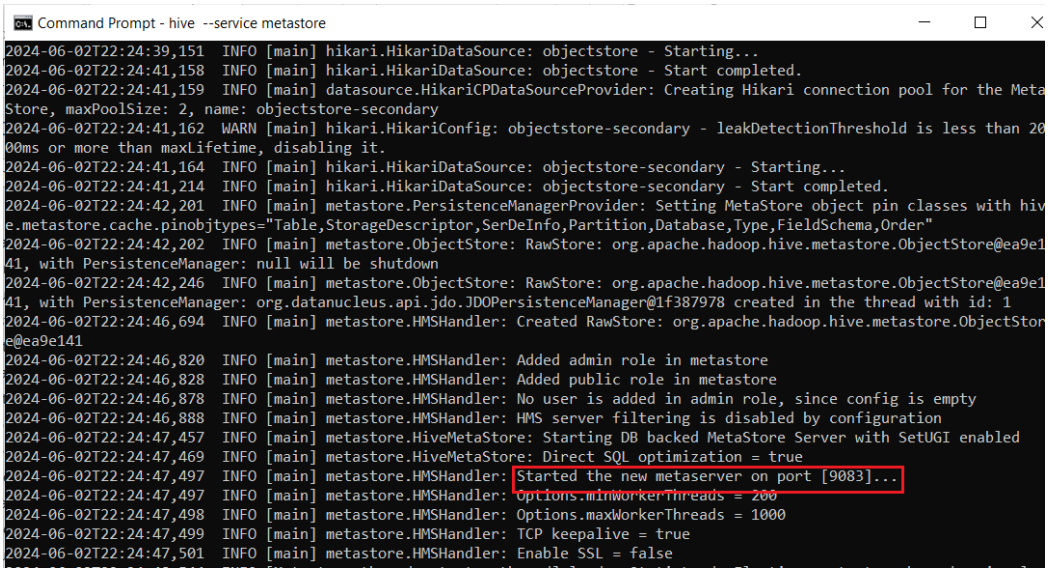


- After the above configuration change is done, restart Hadoop services first.
  Open **Windows Command Prompt** in **Administrator** mode and run the following commands.

```
stop-dfs.cmd
stop-yarn.cmd
start-dfs.cmd
start-yarn.cmd
```

- When Hadoop services are up and running, close and start Hive Metastore service using these commands.

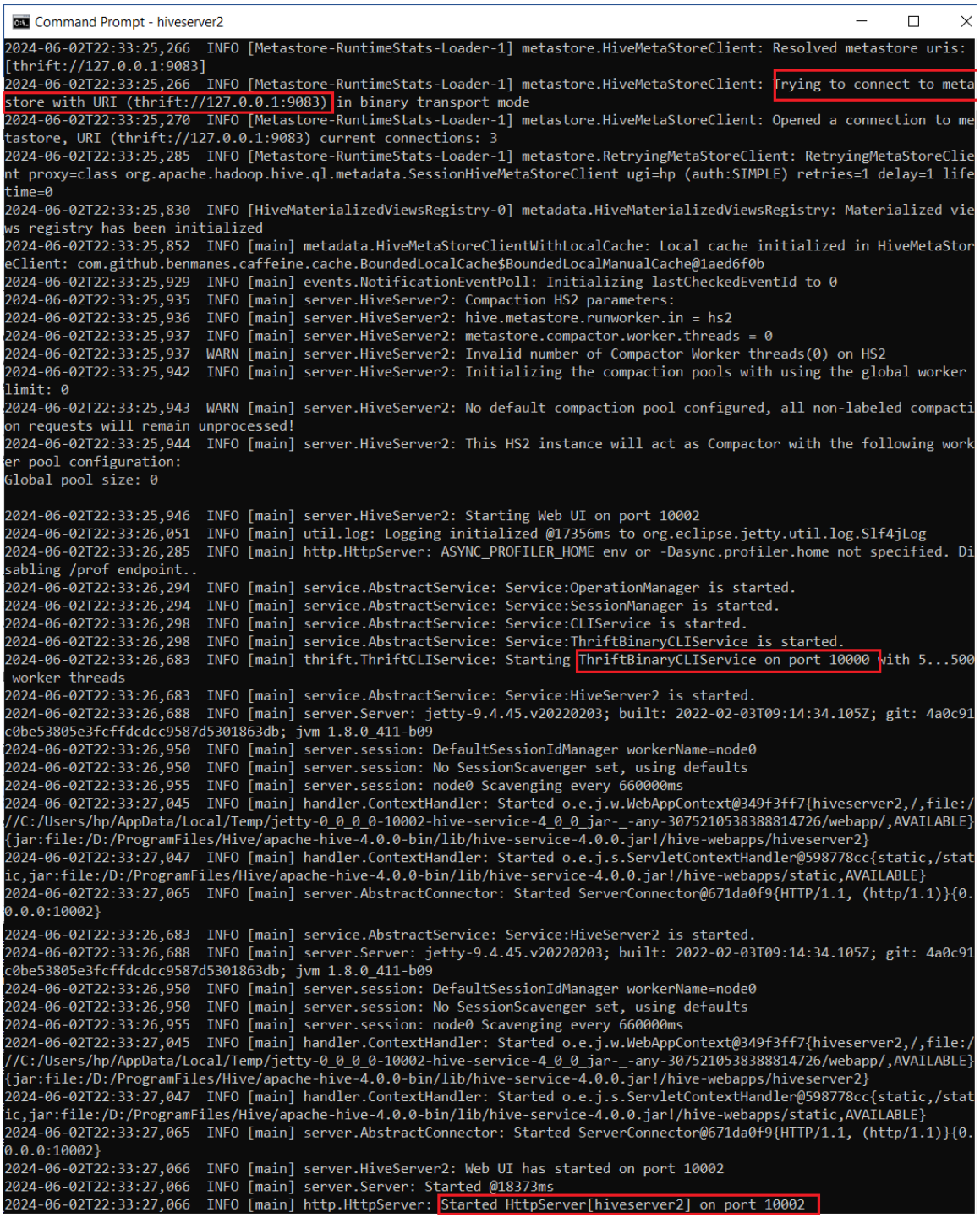
```
set HADOOP_CLIENT_OPTS=-Dhive.root.logger=console
hive --service metastore
```



Here, we can see that `metastore` service started on port `9083`.

- After Hive Metastore is running, close and start HiveServer2 using these commands.

```
set HADOOP_CLIENT_OPTS=-Dhive.root.logger=console
hiveserver2
```

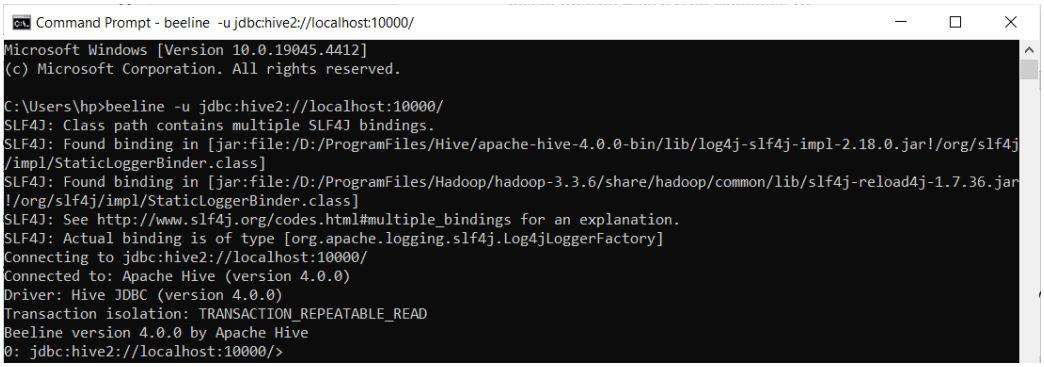


Here, we can see that `HiveServer2` service connected to metastore running over Thrift URI `thrift://127.0.01:9083` and started on port `10002` and `ThriftBinaryCLIService` started on port `10000`.

9.9. Run Queries in Beeline CLI:

Open new command prompt and connect remote Beeline CLI as anonymous user with this command

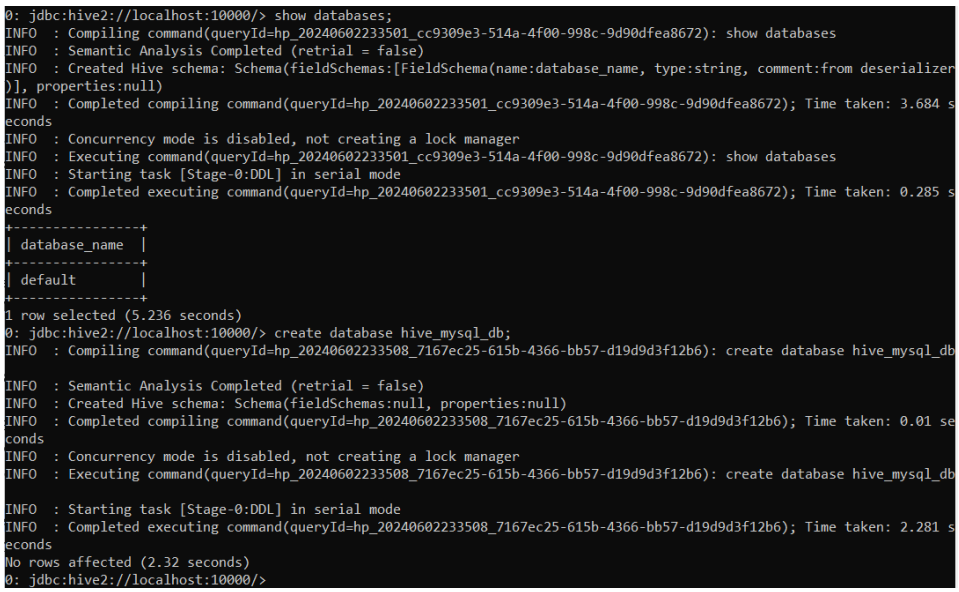
```
beeline -u jdbc:hive2://localhost:10000/
```



After beeline is connected, run these queries to create a database, table and load data into it.

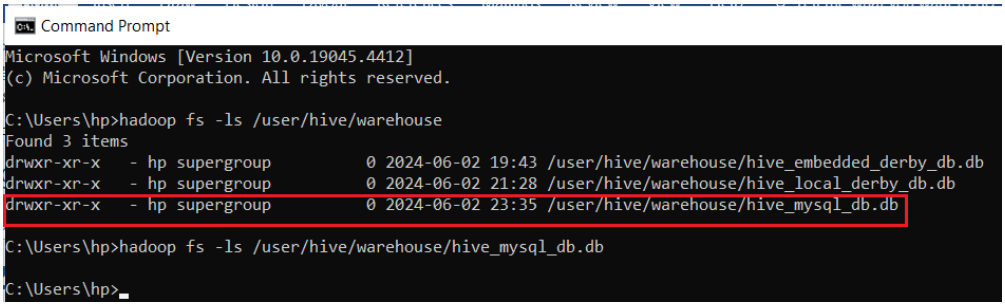
- Create a database named `hive_mysql_db`

```
show databases;
create database hive_mysql_db;
```



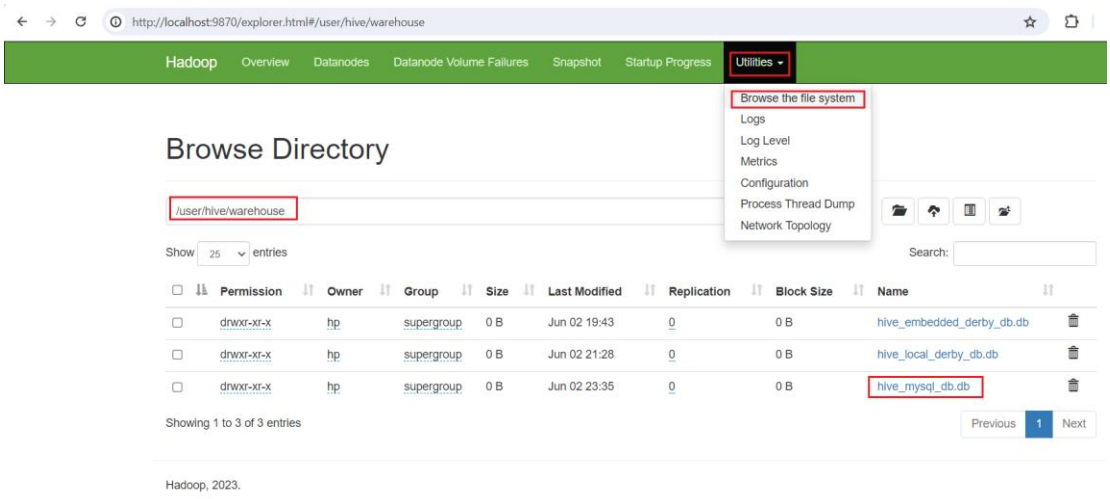
As soon as we run the above query, Hive has created `hive_mysql_db.db` database in HDFS `/user/hive/warehouse` location.

```
hadoop fs -ls /user/hive/warehouse
```



The same is visible in NameNode UI: http://localhost:9870/dfshealth.html

Open NameNode UI, go to **Utilities** tab and select **Browse the file system** option. Enter the directory name `/user/hive/warehouse` and you can see `hive_mysql_db.db` folder available.

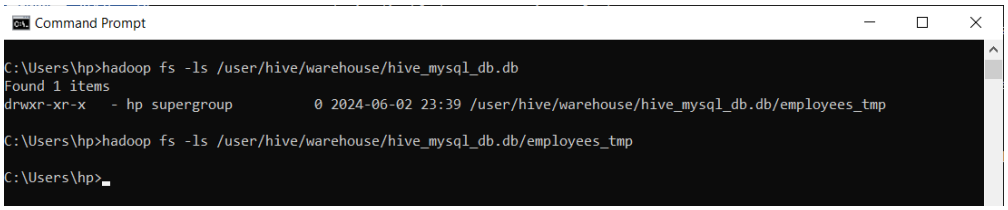


- Create a table named `employees` in `hive_mysql_db` database. This table is created based on the columns data in the CSV file that we are going to load.

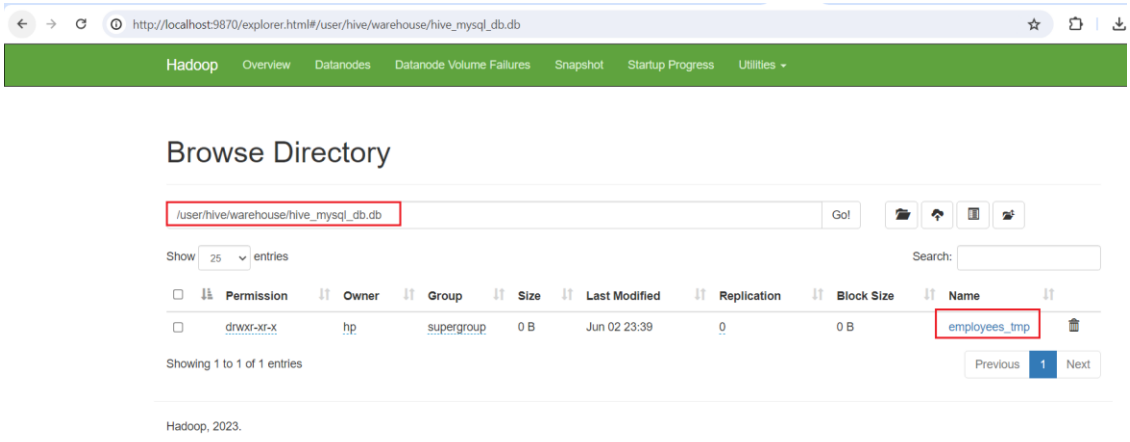
```
use hive_mysql_db;

create table employees_tmp(employee_id int, first_name string, last_name
string, email string, phone_number string, hire_date string, job_id string,
salary int, commission_pct int, manager_id int, department_id int) row format
delimited fields terminated by ','
tblproperties('skip.header.line.count'='1');
```

```
0: jdbc:hive2://localhost:10000/> use hive_mysql_db;
INFO  : Compiling command(queryId=hp_20240602233907_d87a5ab4-c0cd-4f77-b27c-555ec0f10023): use hive_mysql_db
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hp_20240602233907_d87a5ab4-c0cd-4f77-b27c-555ec0f10023); Time taken: 0.081 s
econds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240602233907_d87a5ab4-c0cd-4f77-b27c-555ec0f10023): use hive_mysql_db
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hp_20240602233907_d87a5ab4-c0cd-4f77-b27c-555ec0f10023); Time taken: 0.039 s
econds
No rows affected (0.174 seconds)
0: jdbc:hive2://localhost:10000/> create table employees_tmp(employee_id int, first_name string, last_name string, email
 string, phone_number string, hire_date string, job_id string, salary int, commission_pct int, manager_id int, departmen
t_id int) row format delimited fields terminated by ',' tblproperties ('skip.header.line.count'='1');
INFO  : Compiling command(queryId=hp_20240602233913_8893b8f9-d44d-476c-9e15-eb2a1511c1bc): create table employees_tmp(em
ployee_id int, first_name string, last_name string, email string, phone_number string, hire_date string, job_id string,
salary int, commission_pct int, manager_id int, department_id int) row format delimited fields terminated by ',' tblprop
erties ('skip.header.line.count'='1')
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hp_20240602233913_8893b8f9-d44d-476c-9e15-eb2a1511c1bc); Time taken: 0.149 s
econds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240602233913_8893b8f9-d44d-476c-9e15-eb2a1511c1bc): create table employees_tmp(em
ployee_id int, first_name string, last_name string, email string, phone_number string, hire_date string, job_id string,
salary int, commission_pct int, manager_id int, department_id int) row format delimited fields terminated by ',' tblprop
erties ('skip.header.line.count'='1')
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hp_20240602233913_8893b8f9-d44d-476c-9e15-eb2a1511c1bc); Time taken: 1.443 s
econds
No rows affected (1.626 seconds)
0: jdbc:hive2://localhost:10000/>
```

On HDFS, `employees_tmp` folder is created in `/user/hive/warehouse/hive_mysql_db.db` location.

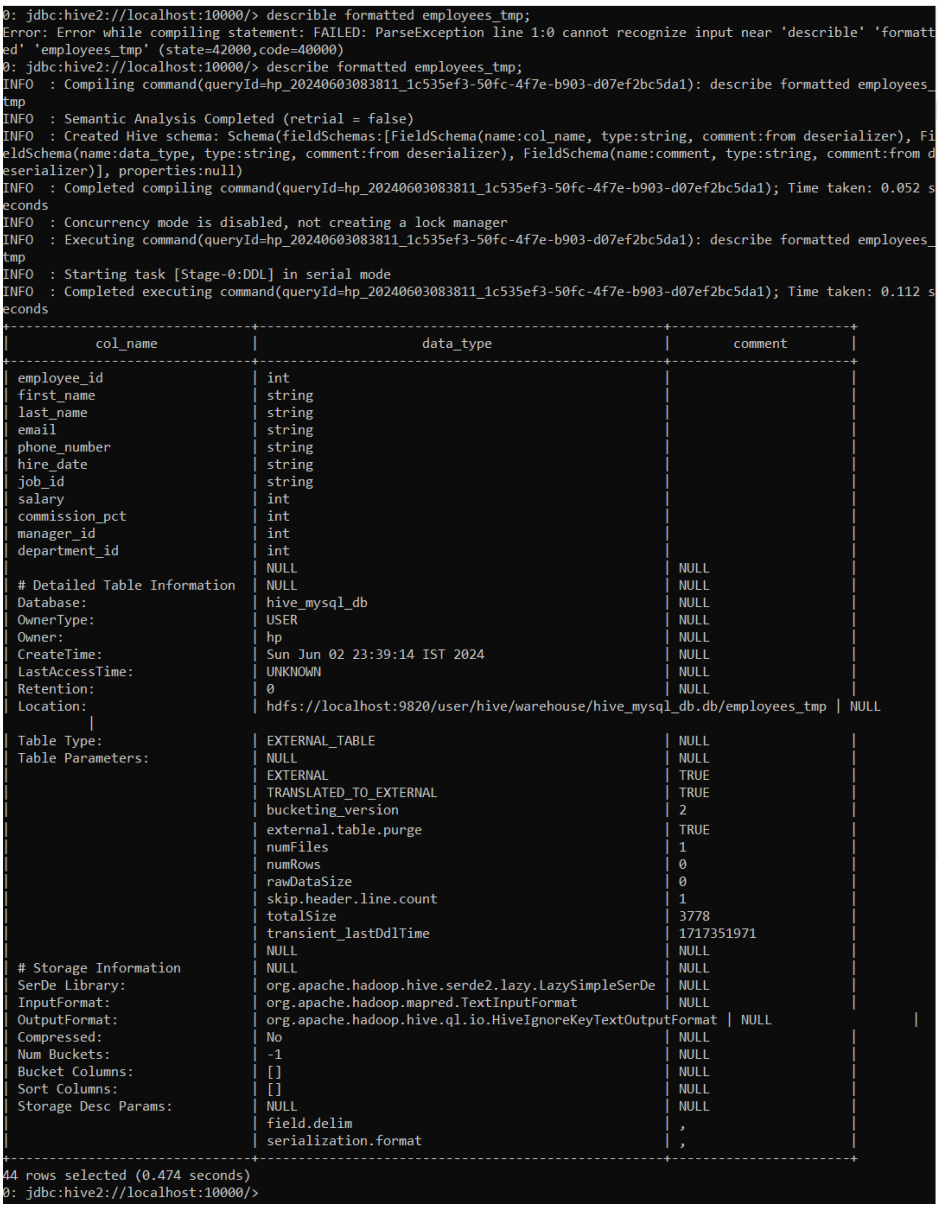


On NameNode UI, click on `hive_mysql_db.db` folder to see `employees_tmp` folder.



- Verify the format of the `employees_tmp` table created.

```
describe formatted employees_tmp;
```



```
0: jdbc:hive2://localhost:10000/> describe formatted employees_tmp;
Error: Error while compiling statement: FAILED: ParseException line 1:0 cannot recognize input near 'describle' 'formatt
ed' 'employees_tmp' (state=42000,code=40000)
0: jdbc:hive2://localhost:10000/> describe formatted employees_tmp;
INFO  : Compiling command(queryId=hp_20240603083811_1c535ef3-50fc-4f7e-b903-d07ef2bc5da1): describe formatted employees_
tmp
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:[FieldSchema(name:col_name, type:string, comment:from deserializer), Fi
eldSchema(name:data_type, type:string, comment:from deserializer), FieldSchema(name:comment, type:string, comment:from d
eserializer)], properties:null)
INFO  : Completed compiling command(queryId=hp_20240603083811_1c535ef3-50fc-4f7e-b903-d07ef2bc5da1); Time taken: 0.052 s
econds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240603083811_1c535ef3-50fc-4f7e-b903-d07ef2bc5da1): describe formatted employees_
tmp
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hp_20240603083811_1c535ef3-50fc-4f7e-b903-d07ef2bc5da1); Time taken: 0.112 s
econds
+-------------------------------+----------------------------------------------------+-----------------------+
|           col_name            |                     data_type                      |        comment        |
+-------------------------------+----------------------------------------------------+-----------------------+
| employee_id                   | int                                                |                       |
| first_name                    | string                                             |                       |
| last_name                     | string                                             |                       |
| email                         | string                                             |                       |
| phone_number                  | string                                             |                       |
| hire_date                     | string                                             |                       |
| job_id                        | string                                             |                       |
| salary                        | int                                                |                       |
| commission_pct                | int                                                |                       |
| manager_id                    | int                                                |                       |
| department_id                 | int                                                |                       |
|                               | NULL                                               | NULL                  |
| # Detailed Table Information  | NULL                                               | NULL                  |
| Database:                     | hive_mysql_db                                      | NULL                  |
| OwnerType:                    | USER                                               | NULL                  |
| Owner:                        | hp                                                 | NULL                  |
| CreateTime:                   | Sun Jun 02 23:39:14 IST 2024                       | NULL                  |
| LastAccessTime:               | UNKNOWN                                            | NULL                  |
| Retention:                    | 0                                                  | NULL                  |
| Location:                     | hdfs://localhost:9820/user/hive/warehouse/hive_mysql_db.db/employees_tmp | NULL
          |
| Table Type:                   | EXTERNAL_TABLE                                     | NULL                  |
| Table Parameters:             | NULL                                               | NULL                  |
|                               | EXTERNAL                                           | TRUE                  |
|                               | TRANSLATED_TO_EXTERNAL                             | TRUE                  |
|                               | bucketing_version                                  | 2                     |
|                               | external.table.purge                               | TRUE                  |
|                               | numFiles                                           | 1                     |
|                               | numRows                                            | 0                     |
|                               | rawDataSize                                        | 0                     |
|                               | skip.header.line.count                             | 1                     |
|                               | totalSize                                          | 3778                  |
|                               | transient_lastDdlTime                              | 1717351971            |
|                               | NULL                                               | NULL                  |
| # Storage Information         | NULL                                               | NULL                  |
| SerDe Library:                | org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe | NULL                  |
| InputFormat:                  | org.apache.hadoop.mapred.TextInputFormat           | NULL                  |
| OutputFormat:                 | org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat | NULL
          |
| Compressed:                   | No                                                 | NULL                  |
| Num Buckets:                  | -1                                                 | NULL                  |
| Bucket Columns:               | []                                                 | NULL                  |
| Sort Columns:                 | []                                                 | NULL                  |
| Storage Desc Params:          | NULL                                               | NULL                  |
|                               | field.delim                                        | ,                     |
|                               | serialization.format                               | ,                     |
+-------------------------------+----------------------------------------------------+-----------------------+
44 rows selected (0.474 seconds)
0: jdbc:hive2://localhost:10000/>
```

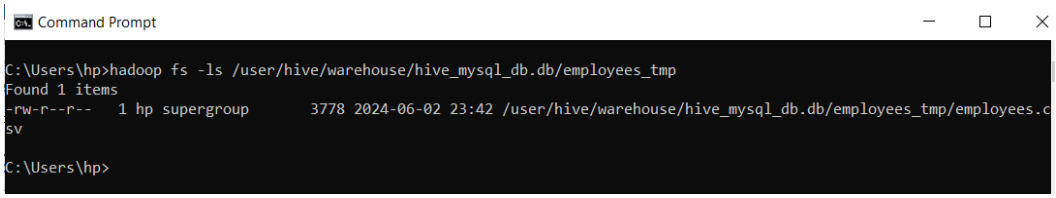
- Load data from `employees.csv` file located in the local directory `D:\Datasets` into the hive table `employees_tmp`.

  Note: This CSV file is available in [this location] that you can download and copy to `D:\Datasets` folder in your machine.

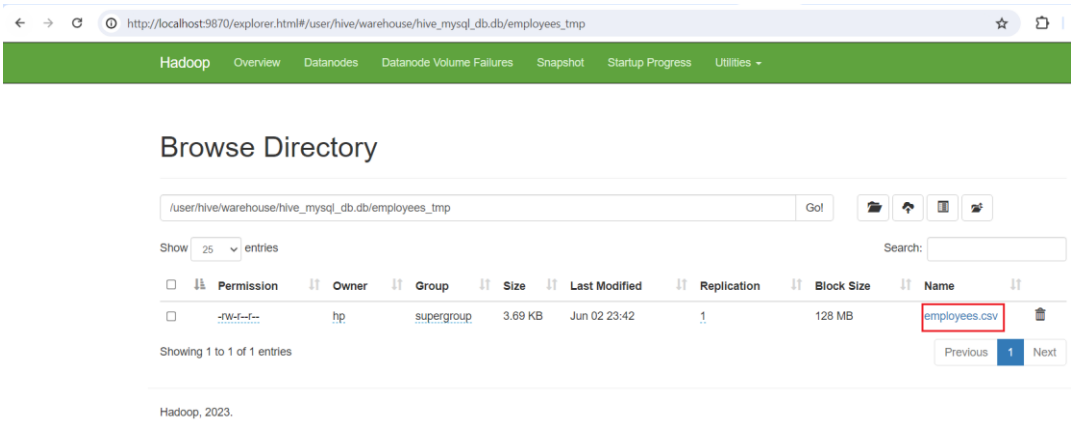
```
load data local inpath 'file:///D:\Datasets\employees.csv' into table
employees_tmp;
```

```
0: jdbc:hive2://localhost:10000/> load data local inpath 'file:///D:\Datasets\employees.csv' into table employees_tmp;
INFO  : Compiling command(queryId=hp_20240602234249_ea7bc217-eb92-4426-ba03-19207cbe6f2e): load data local inpath 'file:
///D:\Datasets\employees.csv' into table employees_tmp
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hp_20240602234249_ea7bc217-eb92-4426-ba03-19207cbe6f2e); Time taken: 0.161 s
econds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240602234249_ea7bc217-eb92-4426-ba03-19207cbe6f2e): load data local inpath 'file:
///D:\Datasets\employees.csv' into table employees_tmp
INFO  : Starting task [Stage-0:MOVE] in serial mode
INFO  : Loading data to table hive_mysql_db.employees_tmp from file:/D:/Datasets/employees.csv
INFO  : Starting task [Stage-1:STATS] in serial mode
INFO  : Executing stats task
INFO  : Table hive_mysql_db.employees_tmp stats: [numFiles=1, numRows=0, totalSize=3778, rawDataSize=0, numFilesErasureC
oded=0]
INFO  : Completed executing command(queryId=hp_20240602234249_ea7bc217-eb92-4426-ba03-19207cbe6f2e); Time taken: 2.123 s
econds
No rows affected (2.328 seconds)
0: jdbc:hive2://localhost:10000/>
```

In HDFS, there is a `employees.csv` file created in `/user/hive/warehouse/hive_mysql_db.db/employees_tmp` location.



On NameNode UI, click on `employees_tmp` folder and you can see `employees.csv` file is created.



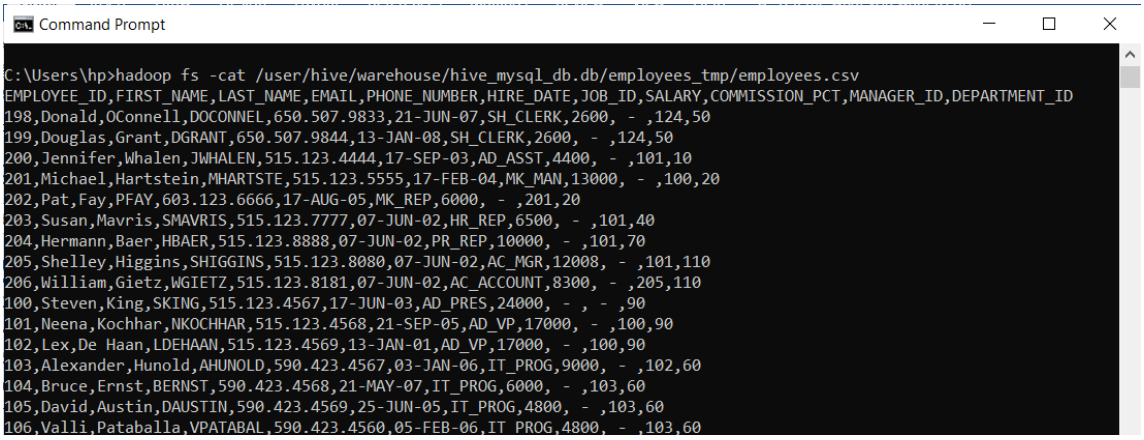
- Select top 5 records in `employees_tmp` table from Hive CLI.

```
set hive.cli.print.header=true;
select * from employees_tmp limit 5;
```

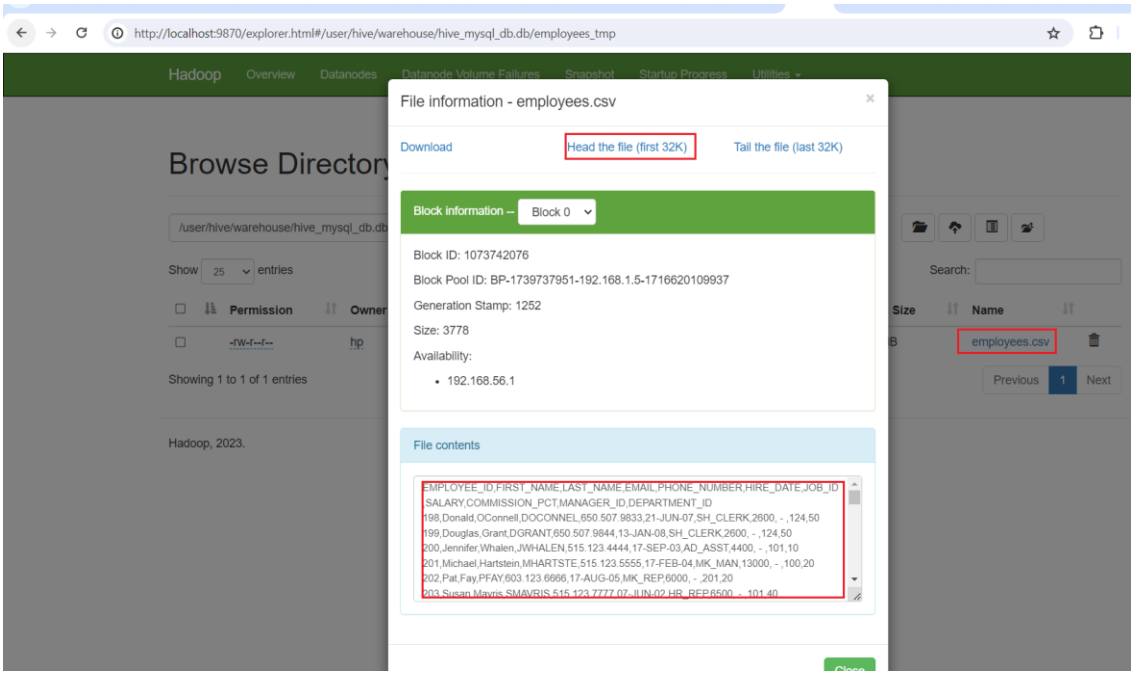
```
0: jdbc:hive2://localhost:10000/> set hive.cli.print.header=true;
No rows affected (0.043 seconds)
0: jdbc:hive2://localhost:10000/> select * from employees_tmp limit 5;
INFO  : Compiling command(queryId=hp_20240602234441_8d5c05fb-e7fe-474c-9719-3695b7232254): select * from employees_tmp l
imit 5
INFO  : No Stats for hive_mysql_db@employees_tmp, Columns: commission_pct, manager_id, department_id, job_id, employee_i
d, last_name, phone_number, hire_date, salary, first_name, email
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:[FieldSchema(name:employees_tmp.employee_id, type:int, comment:null), F
ieldSchema(name:employees_tmp.first_name, type:string, comment:null), FieldSchema(name:employees_tmp.last_name, type:str
ing, comment:null), FieldSchema(name:employees_tmp.email, type:string, comment:null), FieldSchema(name:employees_tmp.pho
ne_number, type:string, comment:null), FieldSchema(name:employees_tmp.hire_date, type:string, comment:null), FieldSchema
(name:employees_tmp.job_id, type:string, comment:null), FieldSchema(name:employees_tmp.salary, type:int, comment:null),
FieldSchema(name:employees_tmp.commission_pct, type:int, comment:null), FieldSchema(name:employees_tmp.manager_id, type:
int, comment:null), FieldSchema(name:employees_tmp.department_id, type:int, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hp_20240602234441_8d5c05fb-e7fe-474c-9719-3695b7232254); Time taken: 4.038 s
econds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240602234441_8d5c05fb-e7fe-474c-9719-3695b7232254): select * from employees_tmp l
imit 5
INFO  : Completed executing command(queryId=hp_20240602234441_8d5c05fb-e7fe-474c-9719-3695b7232254); Time taken: 0.001 s
econds
+----------------------------+-------------------------+-------------------------+---------------------+------------
-----------------+-------------------------+----------------------+-----------------------+--------------------------
----+-------------------------+-----------------------------+
| employees_tmp.employee_id  | employees_tmp.first_name  | employees_tmp.last_name  | employees_tmp.email  | employees_t
mp.phone_number  | employees_tmp.hire_date  | employees_tmp.job_id  | employees_tmp.salary  | employees_tmp.commission_p
ct  | employees_tmp.manager_id  | employees_tmp.department_id  |
+----------------------------+-------------------------+-------------------------+---------------------+------------
-----------------+-------------------------+----------------------+-----------------------+--------------------------
----+-------------------------+-----------------------------+
| 198                        | Donald                  | OConnell                | DOCONNEL            | 650.507.983
3                | 21-JUN-07               | SH_CLERK             | 2600                | NULL
    | 124                      | 50                      |
| 199                        | Douglas                 | Grant                   | DGRANT              | 650.507.984
4                | 13-JAN-08               | SH_CLERK             | 2600                | NULL
    | 124                      | 50                      |
| 200                        | Jennifer                | Whalen                  | JWHALEN             | 515.123.444
4                | 17-SEP-03               | AD_ASST              | 4400                | NULL
    | 101                      | 10                      |
| 201                        | Michael                 | Hartstein               | MHARTSTE            | 515.123.555
5                | 17-FEB-04               | MK_MAN               | 13000               | NULL
    | 100                      | 20                      |
| 202                        | Pat                     | Fay                     | PFAY                | 603.123.666
6                | 17-AUG-05               | MK_REP               | 6000                | NULL
    | 201                      | 20                      |
+----------------------------+-------------------------+-------------------------+---------------------+------------
-----------------+-------------------------+----------------------+-----------------------+--------------------------
----+-------------------------+-----------------------------+
5 rows selected (4.315 seconds)
0: jdbc:hive2://localhost:10000/>
```

We can verify the same in HDFS using `hadoop cat` command.

```
hadoop fs -cat
/user/hive/warehouse/hive_mysql_db.db/employees_tmp/employees.csv
```



In the NameNode UI, click on `employees.csv` file and go to **Head the file** tab to see first few records or **Tail the file** tab to see last few records.

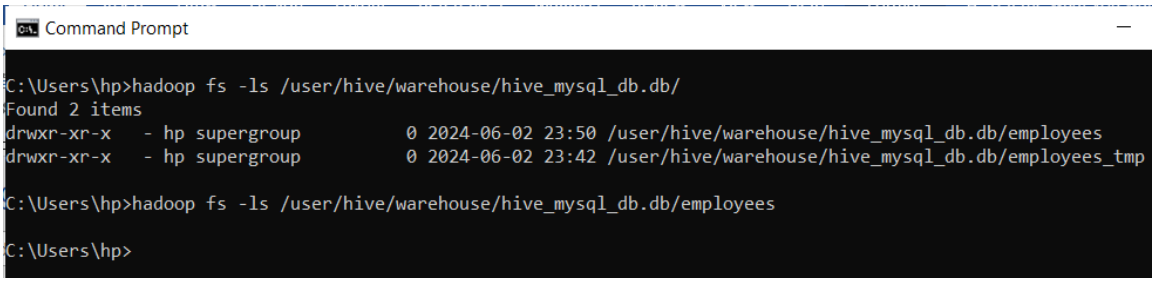


- Now, let us create a new table named `employees` similar to `employees_tmp` table except that data type of `hire_date` column would be `date` instead of `string`.

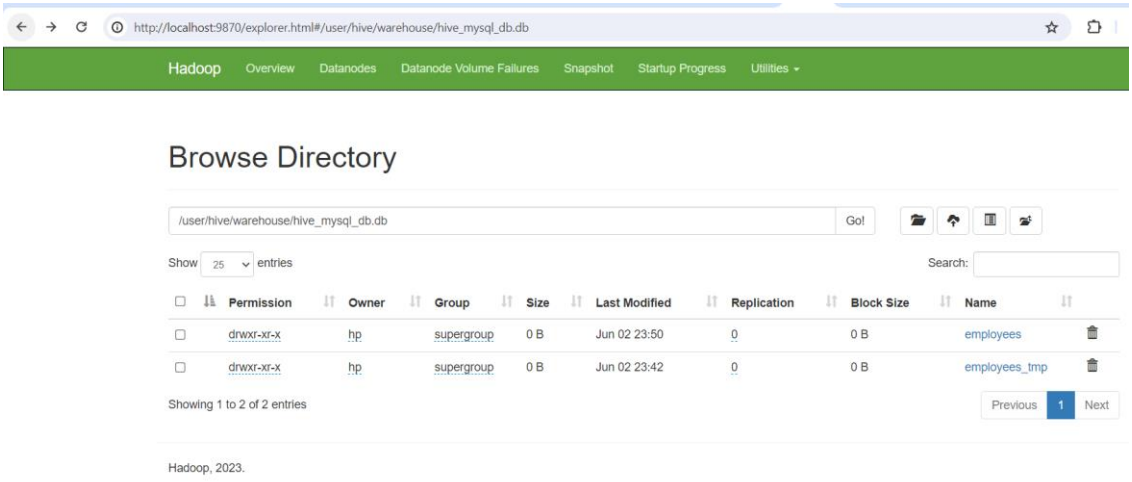
```
create table employees(employee_id int, first_name string, last_name string,
email string, phone_number string, hire_date date, job_id string, salary int,
commission_pct int, manager_id int, department_id int)
tblproperties('skip.header.line.count'='1');
```

```
0: jdbc:hive2://localhost:10000/> create table employees(employee_id int, first_name string, last_name string, email string, phone_number string, hire_date date, job_id string, salary int, commission_pct int, manager_id int, department_id int) tblproperties("skip.header.line.count"="1");
INFO  : Compiling command(queryId=hp_20240603085427_f936390c-3327-476a-93e8-03be269792c2): create table employees(employee_id int, first_name string, last_name string, email string, phone_number string, hire_date date, job_id string, salary int, commission_pct int, manager_id int, department_id int) tblproperties("skip.header.line.count"="1")
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hp_20240603085427_f936390c-3327-476a-93e8-03be269792c2); Time taken: 0.015 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240603085427_f936390c-3327-476a-93e8-03be269792c2): create table employees(employee_id int, first_name string, last_name string, email string, phone_number string, hire_date date, job_id string, salary int, commission_pct int, manager_id int, department_id int) tblproperties("skip.header.line.count"="1")
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hp_20240603085427_f936390c-3327-476a-93e8-03be269792c2); Time taken: 0.26 seconds
No rows affected (0.315 seconds)
0: jdbc:hive2://localhost:10000/>
```

On HDFS, `employees` folder is created in `/user/hive/warehouse/hive_mysql_db.db` location.



On NameNode UI, go to `/user/hive/warehouse/hive_mysql_db.db` folder to see `employees` folder.



- Load data into `employees` table from `employees_tmp` table by transforming `hire_date` column in `employees_tmp` table into the Hive accepted date format.

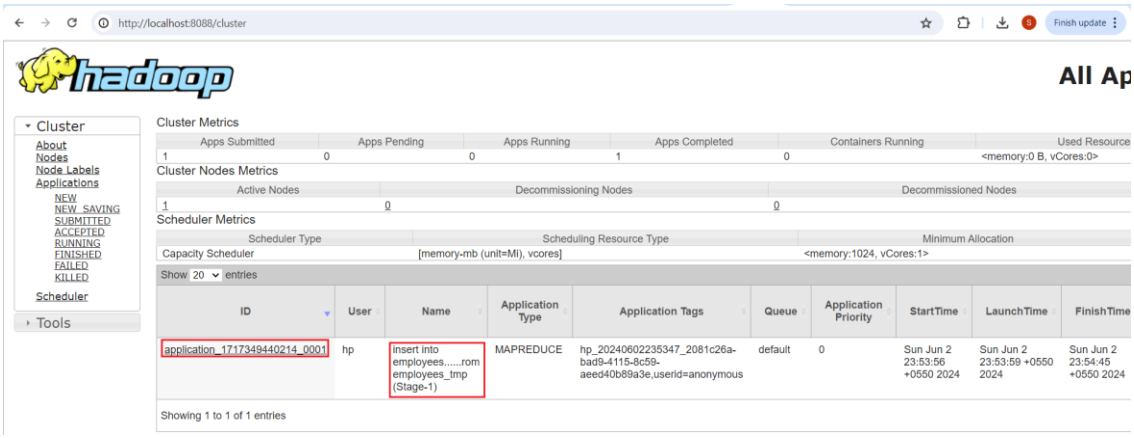
```
insert into employees select employee_id, first_name, last_name, email,
phone_number, from_unixtime(unix_timestamp(hire_date,'dd-MMM-yy'),'yyyy-MM-
```

```
dd') as hire_date, job_id, salary, commission_pct, manager_id, department_id
from employees_tmp;
```

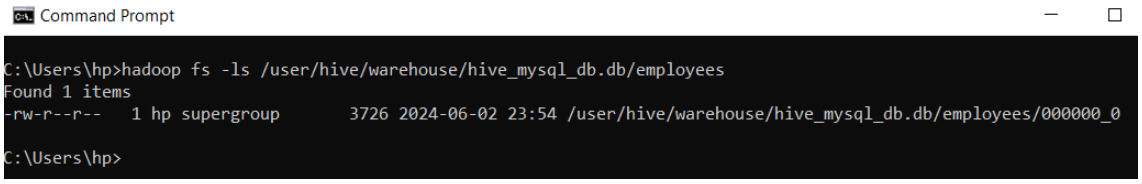
```
0: jdbc:hive2://localhost:10000/> insert into employees select employee_id, first_name, last_name, email, phone_number, from_unixtime(unix_timest
amp(hire_date,'dd-MMM-yy'),'yyyy-MM-dd') as hire_date, job_id, salary, commission_pct, manager_id, department_id from employees_tmp;
INFO  : Compiling command(queryId=hp_20240602235347_2081c26a-bad9-4115-8c59-aeed40b89a3e): insert into employees select employee_id, first_name,
last_name, email, phone_number, from_unixtime(unix_timestamp(hire_date,'dd-MMM-yy'),'yyyy-MM-dd') as hire_date, job_id, salary, commission_pct, m
anager_id, department_id from employees_tmp
INFO  : No Stats for hive_mysql_db@employees_tmp, Columns: commission_pct, manager_id, department_id, job_id, employee_id, last_name, phone_numbe
r, hire_date, salary, first_name, email
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:[FieldSchema(name:_col0, type:int, comment:null), FieldSchema(name:_col1, type:string, comment:n
ull), FieldSchema(name:_col2, type:string, comment:null), FieldSchema(name:_col3, type:string, comment:null), FieldSchema(name:_col4, type:string
, comment:null), FieldSchema(name:_col5, type:date, comment:null), FieldSchema(name:_col6, type:string, comment:null), FieldSchema(name:_col7, ty
pe:int, comment:null), FieldSchema(name:_col8, type:int, comment:null), FieldSchema(name:_col9, type:int, comment:null), FieldSchema(name:_col10,
 type:int, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hp_20240602235347_2081c26a-bad9-4115-8c59-aeed40b89a3e); Time taken: 1.61 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240602235347_2081c26a-bad9-4115-8c59-aeed40b89a3e): insert into employees select employee_id, first_name,
last_name, email, phone_number, from_unixtime(unix_timestamp(hire_date,'dd-MMM-yy'),'yyyy-MM-dd') as hire_date, job_id, salary, commission_pct, m
anager_id, department_id from employees_tmp
WARN  : Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. tez
) or using Hive 1.X releases.
INFO  : Query ID = hp_20240602235347_2081c26a-bad9-4115-8c59-aeed40b89a3e
INFO  : Total jobs = 3
INFO  : Launching Job 1 out of 3
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Number of reduce tasks determined at compile time: 1
INFO  : In order to change the average load for a reducer (in bytes):
INFO  :   set hive.exec.reducers.bytes.per.reducer=<number>
INFO  : In order to limit the maximum number of reducers:
INFO  :   set hive.exec.reducers.max=<number>
INFO  : In order to set a constant number of reducers:
INFO  :   set mapreduce.job.reduces=<number>
INFO  : number of splits:1
INFO  : Submitting tokens for job: job_1717349440214_0001
INFO  : Executing with tokens: []
INFO  : The url to track the job: http://DESKTOP-KGH2E2G:8088/proxy/application_1717349440214_0001/
INFO  : Starting Job = job_1717349440214_0001, Tracking URL = http://DESKTOP-KGH2E2G:8088/proxy/application_1717349440214_0001/
INFO  : Kill Command = D:\ProgramFiles\Hadoop\hadoop-3.3.6\bin\mapred job  -kill job_1717349440214_0001
INFO  : Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
INFO  : 2024-06-02 23:54:17,177 Stage-1 map = 0%,  reduce = 0%
INFO  : 2024-06-02 23:54:28,873 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 6.514 sec
INFO  : 2024-06-02 23:54:44,697 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 16.434 sec
INFO  : MapReduce Total cumulative CPU time: 16 seconds 434 msec
INFO  : Ended Job = job_1717349440214_0001
INFO  : Starting task [Stage-7:CONDITIONAL] in serial mode
INFO  : Stage-4 is selected by condition resolver.
INFO  : Stage-3 is filtered out by condition resolver.
INFO  : Stage-5 is filtered out by condition resolver.
INFO  : Starting task [Stage-4:MOVE] in serial mode
INFO  : Moving data to directory hdfs://localhost:9820/user/hive/warehouse/hive_mysql_db.db/employees/.hive-staging_hive_2024-06-02_23-53-47_068_
2888876849100680296-1/-ext-10000 from hdfs://localhost:9820/user/hive/warehouse/hive_mysql_db.db/employees/.hive-staging_hive_2024-06-02_23-53-47
_068_2888876849100680296-1/-ext-10002
INFO  : Starting task [Stage-0:MOVE] in serial mode
INFO  : Loading data to table hive_mysql_db.employees from hdfs://localhost:9820/user/hive/warehouse/hive_mysql_db.db/employees/.hive-staging_hiv
e_2024-06-02_23-53-47_068_2888876849100680296-1/-ext-10000
INFO  : Starting task [Stage-2:STATS] in serial mode
INFO  : Executing stats task
INFO  : Table hive_mysql_db.employees stats: [numFiles=1, numRows=51, totalSize=3726, rawDataSize=3675, numFilesErasureCoded=0]
INFO  : MapReduce Jobs Launched:
INFO  : Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 16.434 sec   HDFS Read: 65633 HDFS Write: 7841 HDFS EC Read: 0 SUCCESS
INFO  : Total MapReduce CPU Time Spent: 16 seconds 434 msec
INFO  : Completed executing command(queryId=hp_20240602235347_2081c26a-bad9-4115-8c59-aeed40b89a3e); Time taken: 59.85 seconds
51 rows affected (61.543 seconds)
0: jdbc:hive2://localhost:10000/>
```

The above query submitted a MapReduce job to Hadoop YARN which we can track using the application tracking URL provided.

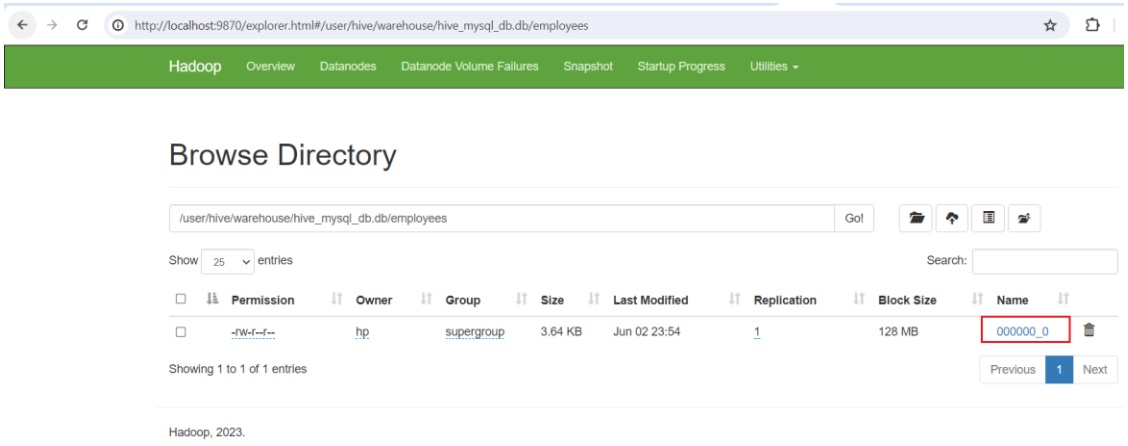
Open Hadoop YARN UI: http://localhost:8088/cluster where we can see the application has been submitted which is running the application *"insert into employees .. from employees_tmp (Stage-1)"*



After the above query is completed, we can see a file named `000000_0` created in `/user/hive/warehouse/hive_mysql_db.db/employees` HDFS location.



On NameNode UI, click on `employees` folder to see `000000_0` file.



- Select top 5 records in `employees` table in Hive CLI.

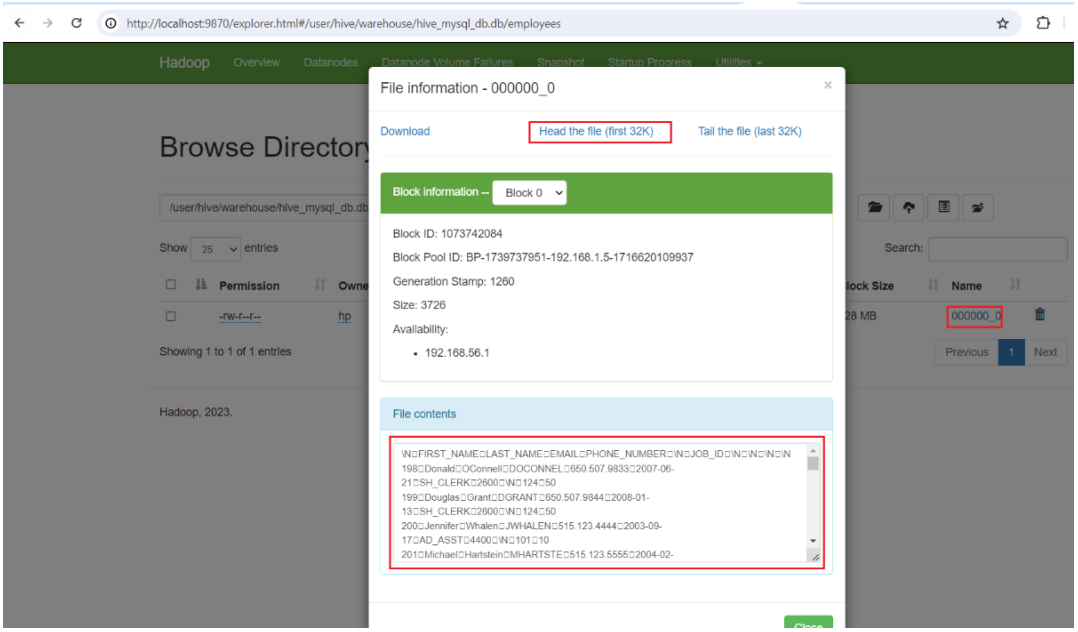
```
select * from employees limit 5;
```

```
0: jdbc:hive2://localhost:10000/> select * from employees limit 5;
INFO  : Compiling command(queryId=hp_20240603085819_84cb37ad-1486-448c-b970-3250fb91b9cb): select * from employees limit 5
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:[FieldSchema(name:employees.employee_id, type:int, comment:null), FieldSchema(name:em
ployees.first_name, type:string, comment:null), FieldSchema(name:employees.last_name, type:string, comment:null), FieldSchema(name:emp
loyees.email, type:string, comment:null), FieldSchema(name:employees.phone_number, type:string, comment:null), FieldSchema(name:employ
ees.hire_date, type:date, comment:null), FieldSchema(name:employees.job_id, type:string, comment:null), FieldSchema(name:employees.sal
ary, type:int, comment:null), FieldSchema(name:employees.commission_pct, type:int, comment:null), FieldSchema(name:employees.manager_i
d, type:int, comment:null), FieldSchema(name:employees.department_id, type:int, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hp_20240603085819_84cb37ad-1486-448c-b970-3250fb91b9cb); Time taken: 0.315 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240603085819_84cb37ad-1486-448c-b970-3250fb91b9cb): select * from employees limit 5
INFO  : Completed executing command(queryId=hp_20240603085819_84cb37ad-1486-448c-b970-3250fb91b9cb); Time taken: 0.001 seconds
+----------------------+----------------------+---------------------+-----------------+-----------------------+-----------------------+
| employees.employee_id  | employees.first_name  | employees.last_name  | employees.email  | employees.phone_number  | employees.hire_date  | employees.job_id  | employees.salary  | employees.commission_pct  | employees.manager_id  | employees.department_id  |
+----------------------+----------------------+---------------------+-----------------+-----------------------+-----------------------+
| 198                  | Donald               | OConnell            | DOCONNEL        | 650.507.9833          | 2007-06-21            | SH_CLERK          | 2600              | NULL                      | 124                   | 50                       |
| 199                  | Douglas              | Grant               | DGRANT          | 650.507.9844          | 2008-01-13            | SH_CLERK          | 2600              | NULL                      | 124                   | 50                       |
| 200                  | Jennifer             | Whalen              | JWHALEN         | 515.123.4444          | 2003-09-17            | AD_ASST           | 4400              | NULL                      | 101                   | 10                       |
| 201                  | Michael              | Hartstein           | MHARTSTE        | 515.123.5555          | 2004-02-17            | MK_MAN            | 13000             | NULL                      | 100                   | 20                       |
| 202                  | Pat                  | Fay                 | PFAY            | 603.123.6666          | 2005-08-17            | MK_REP            | 6000              | NULL                      | 201                   | 20                       |
+----------------------+----------------------+---------------------+-----------------+-----------------------+-----------------------+
5 rows selected (0.403 seconds)
0: jdbc:hive2://localhost:10000/>
```

We can verify the same in HDFS using `hadoop cat` command.

```
hadoop fs -cat /user/hive/warehouse/hive_mysql_db.db/employees/000000_0
```

In the NameNode UI, click on `000000_0` file and go to **Head the file** tab to see first few records.



- Verify if the count of records between `employees_tmp` and `employees` tables are matching.

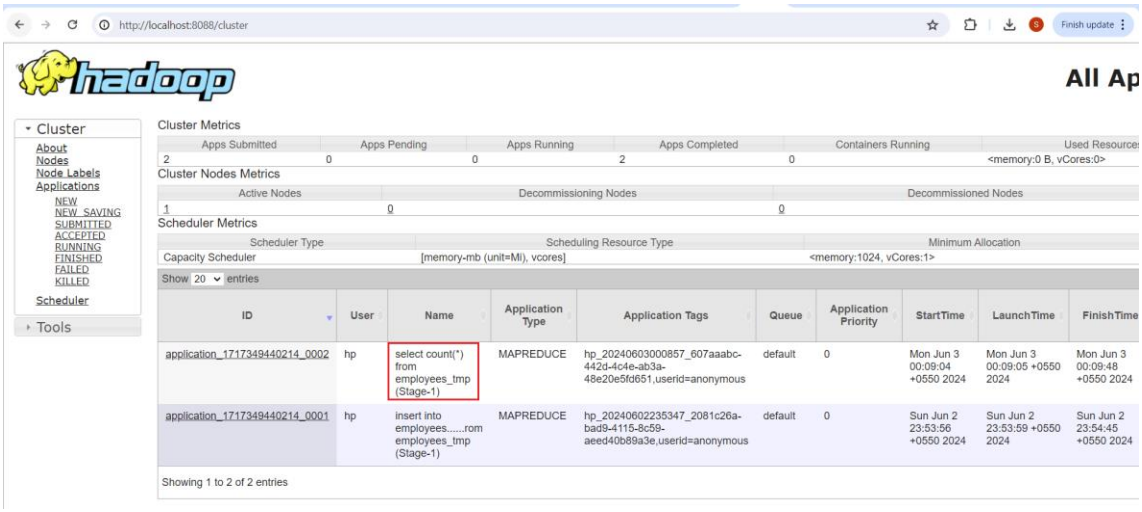
Count records in employees_tmp table:

```
select count(*) from employees_tmp;
```

```
0: jdbc:hive2://localhost:10000/> select count(*) from employees_tmp;
INFO  : Compiling command(queryId=hp_20240603000857_607aaabc-442d-4c4e-ab3a-48e20e5fd651): select count(*) from employees_tmp
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Created Hive schema: Schema(fieldSchemas:[FieldSchema(name:_c0, type:bigint, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hp_20240603000857_607aaabc-442d-4c4e-ab3a-48e20e5fd651); Time taken: 0.332 seconds
INFO  : Concurrency mode is disabled, not creating a lock manager
INFO  : Executing command(queryId=hp_20240603000857_607aaabc-442d-4c4e-ab3a-48e20e5fd651): select count(*) from employees_tmp
WARN  : Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engin
e (i.e. tez) or using Hive 1.X releases.
INFO  : Query ID = hp_20240603000857_607aaabc-442d-4c4e-ab3a-48e20e5fd651
INFO  : Total jobs = 1
INFO  : Launching Job 1 out of 1
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Number of reduce tasks determined at compile time: 1
INFO  : In order to change the average load for a reducer (in bytes):
INFO  :   set hive.exec.reducers.bytes.per.reducer=<number>
INFO  : In order to limit the maximum number of reducers:
INFO  :   set hive.exec.reducers.max=<number>
INFO  : In order to set a constant number of reducers:
INFO  :   set mapreduce.job.reduces=<number>
INFO  : number of splits:1
INFO  : Submitting tokens for job: job_1717349440214_0002
INFO  : Executing with tokens: []
INFO  : The url to track the job: http://DESKTOP-KGH2E2G:8088/proxy/application_1717349440214_0002/
INFO  : Starting Job = job_1717349440214_0002, Tracking URL = http://DESKTOP-KGH2E2G:8088/proxy/application_1717349440214_0002/
INFO  : Kill Command = D:\ProgramFiles\Hadoop\hadoop-3.3.6\bin\mapred job  -kill job_1717349440214_0002
INFO  : Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
INFO  : 2024-06-03 00:09:21,198 Stage-1 map = 0%,  reduce = 0%
INFO  : 2024-06-03 00:09:32,931 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 6.247 sec
INFO  : 2024-06-03 00:09:47,571 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 13.244 sec
INFO  : MapReduce Total cumulative CPU time: 13 seconds 244 msec
INFO  : Ended Job = job_1717349440214_0002
INFO  : MapReduce Jobs Launched:
INFO  : Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 13.244 sec   HDFS Read: 20486 HDFS Write: 102 HDFS EC Read: 0 SUCCESS
INFO  : Total MapReduce CPU Time Spent: 13 seconds 244 msec
+------+
| _c0  |
+------+
| 51   |
+------+
1 row selected (52.793 seconds)
INFO  : Completed executing command(queryId=hp_20240603000857_607aaabc-442d-4c4e-ab3a-48e20e5fd651); Time taken: 52.374 seconds
0: jdbc:hive2://localhost:10000/>
```

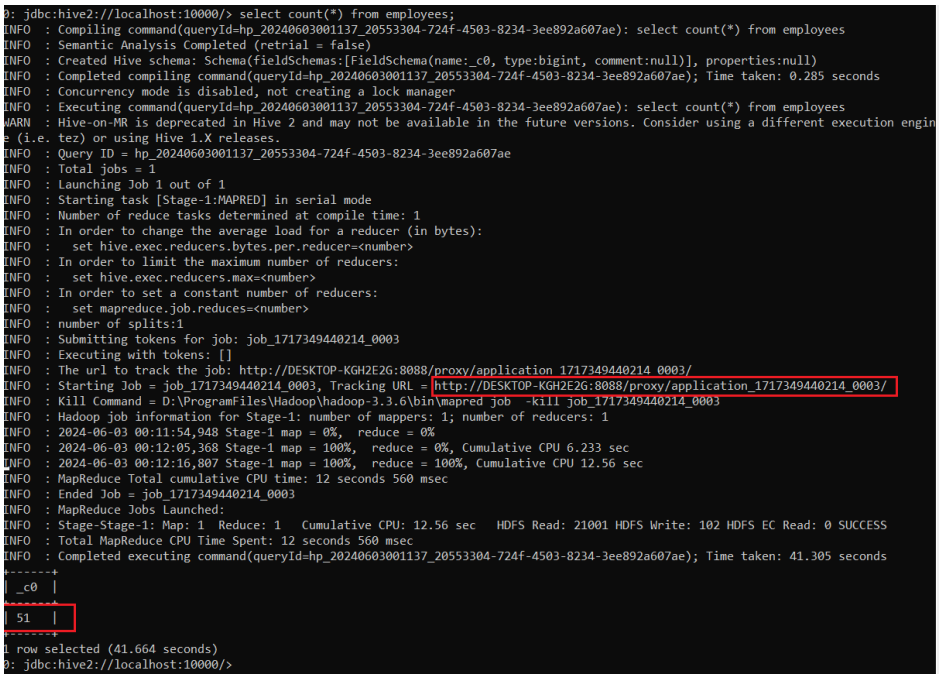
It displayed the count as `51`. Note that the actual records in the table are 50 but it is counting header also as one record and displaying 51.

The above query submitted another job to Hadoop YARN and provided us a tracking URL which we can see in YARN UI.



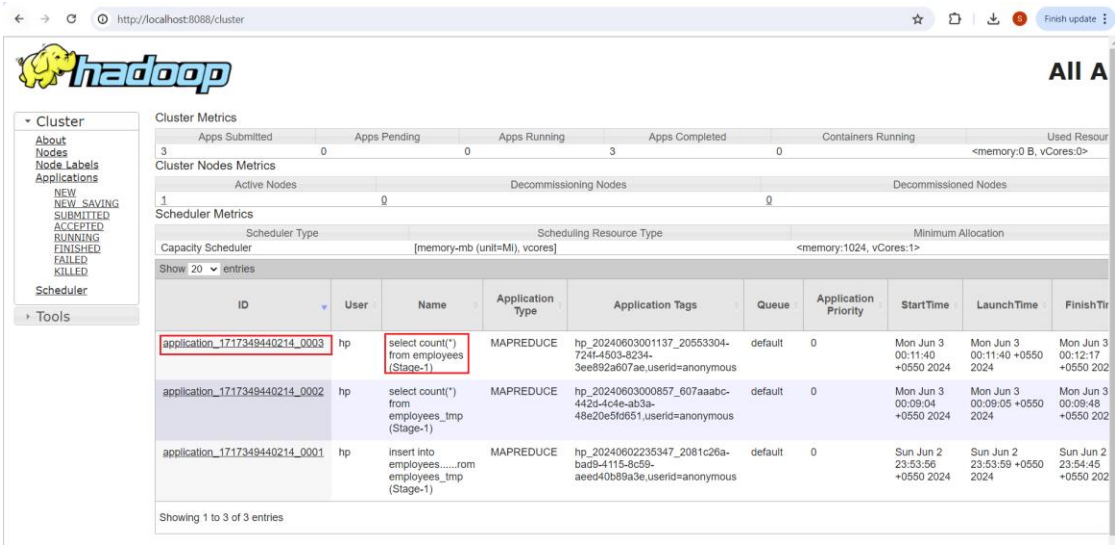
Count records in employees table:

```
select count(*) from employees;
```



It displayed the count as `51`. Note that the actual records in the table are 50 but it is counting header also as one record and displaying 51.

The above query submitted another job to Hadoop YARN and provided us a tracking URL which we can see in YARN UI. It displayed the count as `51`.



9.10. Verify Metadata in MySQL:

Let us connect to MySQL server and verify Hive metadata of database and tables created in MySQL.

Launch MySQL command prompt using this command. Note that we are connecting with `hive` user credentials that we created earlier.

```
mysql -h localhost -u hive -p
```

Provide the password of `hive` user when asked.

In `mysql>` prompt, run the following queries:

- Get the metadata of `hive_mysql_db` database.

```
use hive_metastore;
select * from dbs;
```

```
mysql> use hive_metastore;
Database changed
mysql> select * from dbs;
+-------+-----------------------+-----------------------------------------------------+---------------+------------+------------+
| DB_ID | DESC                  | DB_LOCATION_URI                                     | NAME          | OWNER_NAME | OWNER_TYPE |
+-------+-----------------------+-----------------------------------------------------+---------------+------------+------------+
|     1 | Default Hive database | hdfs://localhost:9820/user/hive/warehouse           | default       | public     | ROLE       |
|    31 | NULL                  | hdfs://localhost:9820/user/hive/warehouse/hive_mysql_db.db | hive_mysql_db | hp         | USER       |
+-------+-----------------------+-----------------------------------------------------+---------------+------------+------------+
2 rows in set (0.00 sec)

mysql>
```

It shows the DB location URI as `hdfs://localhost:9820/user/hive/warehouse/hive_mysql_db.db`.

- Get the metadata of `employees_tmp` and `employees` tables and its columns.

```
select * from tbls;
select * from columns_v2;
```

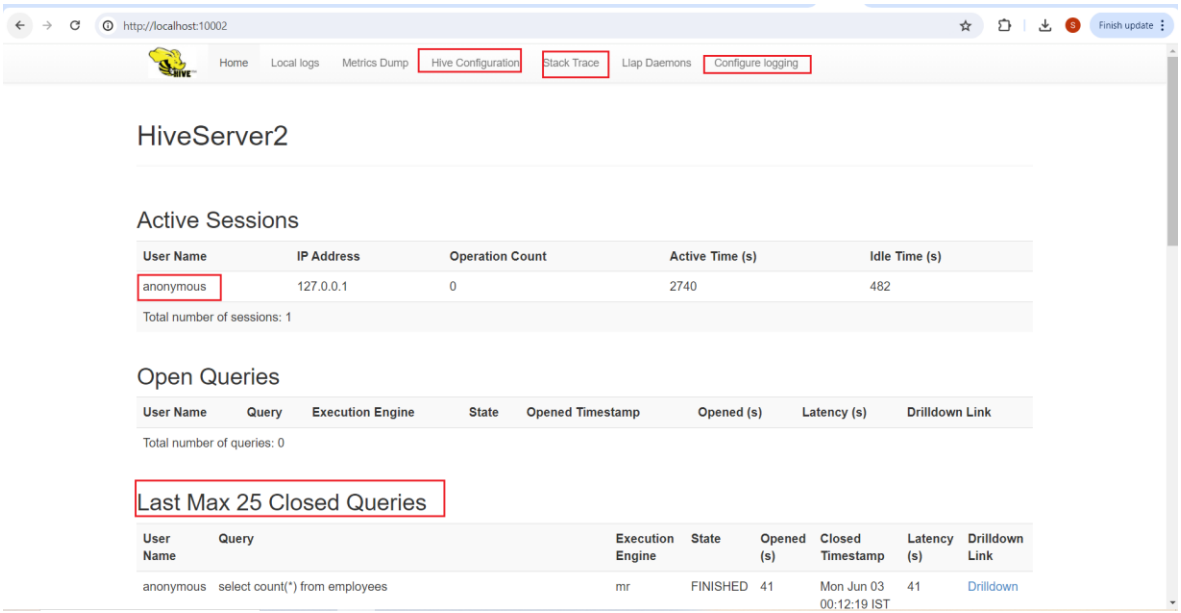
```
mysql> select * from tbls;
+--------+-------------+-------+------------------+-------+------------+-----------+-------+---------------+---------------+--------------------+--------------------+----------+
| TBL_ID | CREATE_TIME | DB_ID | LAST_ACCESS_TIME | OWNER | OWNER_TYPE | RETENTION | SD_ID | TBL_NAME      | TBL_TYPE      | VIEW_EXPANDED_TEXT | VIEW_ORIGINAL_TEXT | IS_REWRITE_ENABLED | WRITE_ID |
+--------+-------------+-------+------------------+-------+------------+-----------+-------+---------------+---------------+--------------------+--------------------+--------------------+----------+
|      1 |  1717351754 |    31 |                0 | hp    | USER       |         0 |     1 | employees_tmp | EXTERNAL_TABLE | NULL               | NULL               |                    |        0 |
|      2 |  1717352413 |    31 |                0 | hp    | USER       |         0 |     2 | employees     | EXTERNAL_TABLE | NULL               | NULL               |                    |        0 |
+--------+-------------+-------+------------------+-------+------------+-----------+-------+---------------+---------------+--------------------+--------------------+--------------------+----------+
2 rows in set (0.00 sec)

mysql> select * from columns_v2;
+-------+---------+---------------+-----------+-------------+
| CD_ID | COMMENT | COLUMN_NAME   | TYPE_NAME | INTEGER_IDX |
+-------+---------+---------------+-----------+-------------+
|     1 | NULL    | commission_pct | int       |           8 |
|     1 | NULL    | department_id | int       |          10 |
|     1 | NULL    | email         | string    |           3 |
|     1 | NULL    | employee_id   | int       |           0 |
|     1 | NULL    | first_name    | string    |           1 |
|     1 | NULL    | hire_date     | string    |           5 |
|     1 | NULL    | job_id        | string    |           6 |
|     1 | NULL    | last_name     | string    |           2 |
|     1 | NULL    | manager_id    | int       |           9 |
|     1 | NULL    | phone_number  | string    |           4 |
|     1 | NULL    | salary        | int       |           7 |
|     2 | NULL    | commission_pct | int       |           8 |
|     2 | NULL    | department_id | int       |          10 |
|     2 | NULL    | email         | string    |           3 |
|     2 | NULL    | employee_id   | int       |           0 |
|     2 | NULL    | first_name    | string    |           1 |
|     2 | NULL    | hire_date     | date      |           5 |
|     2 | NULL    | job_id        | string    |           6 |
|     2 | NULL    | last_name     | string    |           2 |
|     2 | NULL    | manager_id    | int       |           9 |
|     2 | NULL    | phone_number  | string    |           4 |
|     2 | NULL    | salary        | int       |           7 |
+-------+---------+---------------+-----------+-------------+
22 rows in set (0.00 sec)

mysql>
```

It shows `employees_tmp` and `employees` tables and their respective columns.

10. Hive Web UI:

Hive provides the following web interface to monitor HiveServer2 service.

HiveServer2 UI: http://localhost:10002/



Since we last connected to Beeline as `anonymous` user, it shows the **User Name** as `anonymous`.
It also displays the last 25 queries executed on HiveServer2 after the latest restart.
We can also view Hive Configuration, Stack Trace, Configure logging and much more in this UI.

11. HCatalog and WebHCat:

HCatalog is a table and storage management layer for Hadoop that allows **MapReduce**, **Pig** and **Hive** users to easily read and write data on HDFS. HCatalog provides a relational view of data stored in HDFS. It is built on top of the Hive metastore and incorporates Hive's DDL. It provides read and write interfaces for Pig and MapReduce and uses Hive's command line interface for issuing data definition and metadata exploration commands.

WebHCat (previously called as **Templeton**) is the REST API provided to access HCatalog service. Unlike HCatalog, which executed the command directly, WebHCat keeps the Hive, PIG, and MapReduce jobs in queues. The jobs can then be monitored and stopped as needed.

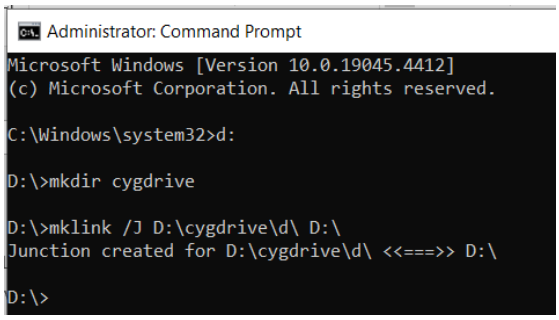
Note: We cannot start **HCatalog** and **WebHCat** server from Windows since they are not compatible for Windows and so, we will use Cygwin utility which allows executing Linux commands from Windows.

11.1. Create Symbolic Link for Cygwin:

Since Java cannot understand Cygwin paths properly, we will first create symbolic links for `cygdrive` to use Cygwin utility.

Open command prompt in Administrator mode and run the following commands:

```
d:
mkdir cygdrive
mklink /J D:\cygdrive\d\ D:\
```



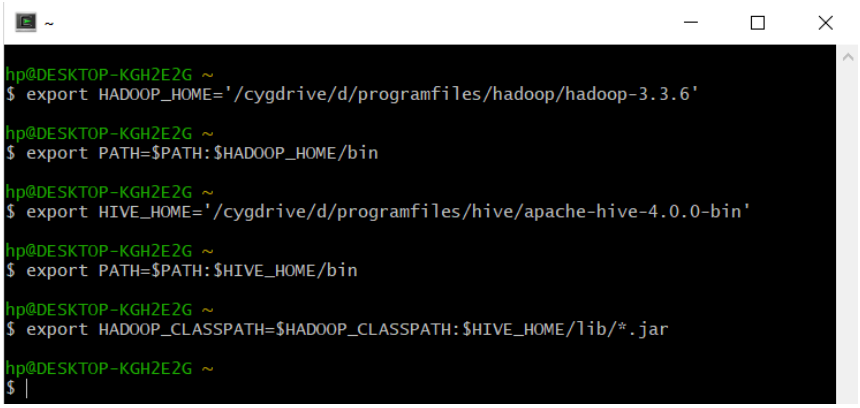
Note that we created `cygdrive` directory in `D` drive since we installed Hive in this drive. If you have installed Hive in different drive then create symbolic link to that drive.

11.2. Setup Env variables for Cygwn:

Now, open **Cygwin64 Terminal** and run the following commands to define environment variables. We can add the below lines to `~/.bashrc` file so we don't need to execute every time we open Cygwin.

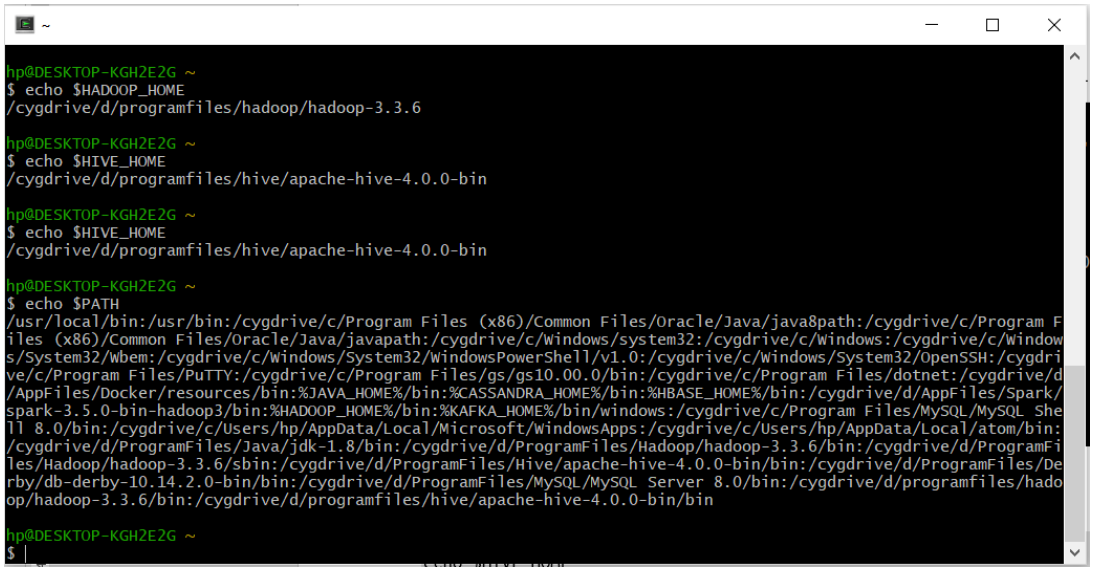
```
export HADOOP_HOME='/cygdrive/d/programfiles/hadoop/hadoop-3.3.6'
export PATH=$PATH:$HADOOP_HOME/bin
export HIVE_HOME='/cygdrive/d/programfiles/hive/apache-hive-4.0.0-bin'
export PATH=$PATH:$HIVE_HOME/bin
export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$HIVE_HOME/lib/*.jar
```

Make sure the above `HADOOP_HOME` and `HIVE_HOME` variables refer to the exact location where Hadoop and Hive were installed.



Verify if the above variables are properly set with these commands

```
echo $HADOOP_HOME
echo $HIVE_HOME
echo $HADOOP_CLASSPATH
echo $PATH
```

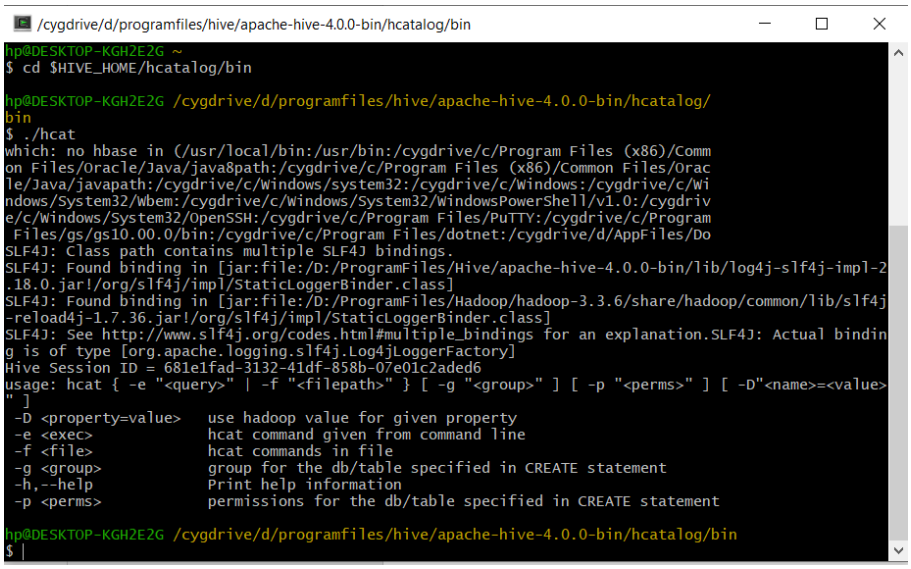


11.3. Start HCatalog CLI:

We can start the HCatalog command line interface by using `hcat` file available in `HIVE_HOME\hcatalog\bin`

In Cygwin, run the following commands to start HCatalog

```
cd $HIVE_HOME/hcatalog/bin
./hcat
```



Note that `hcat` commands can be issued as hive commands and HCatalog CLI supports the following command line options:

`-g` : Tells HCatalog that the table which needs to be created must have group `mygroup`.
For example:
```
hcat -g mygroup ...
```

`-p` : Tells HCatalog that the table which needs to be created must have permissions `"rwxr-xr-x"`.

For example:
```
hcat -p rwxr-xr-x ...
```

`-f` : Tells HCatalog that `myscript.hcatalog` is a file containing DDL commands to execute.
For example:
```
hcat -f myscript.hcatalog ...
```

`-e` : Tells HCatalog to treat the following string as a DDL command and execute it
For example:
```
hcat -e 'create table mytable(a int);' ...
```

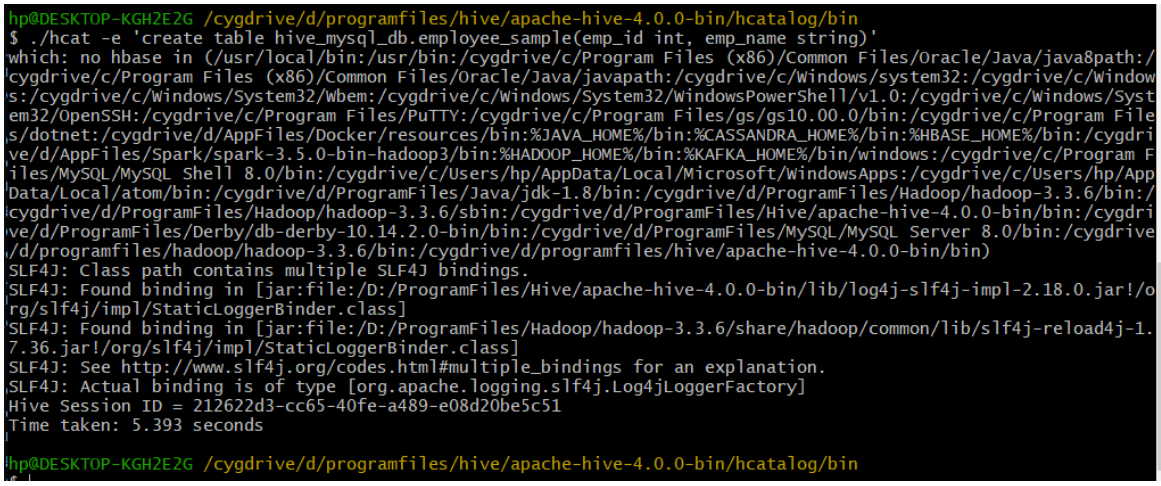
`-D` : Passes the key-value pair to HCatalog as a Java System Property.
For example:
```
hcat -Dkey=value ...
```

`hcat`: Prints a usage message.

For example:

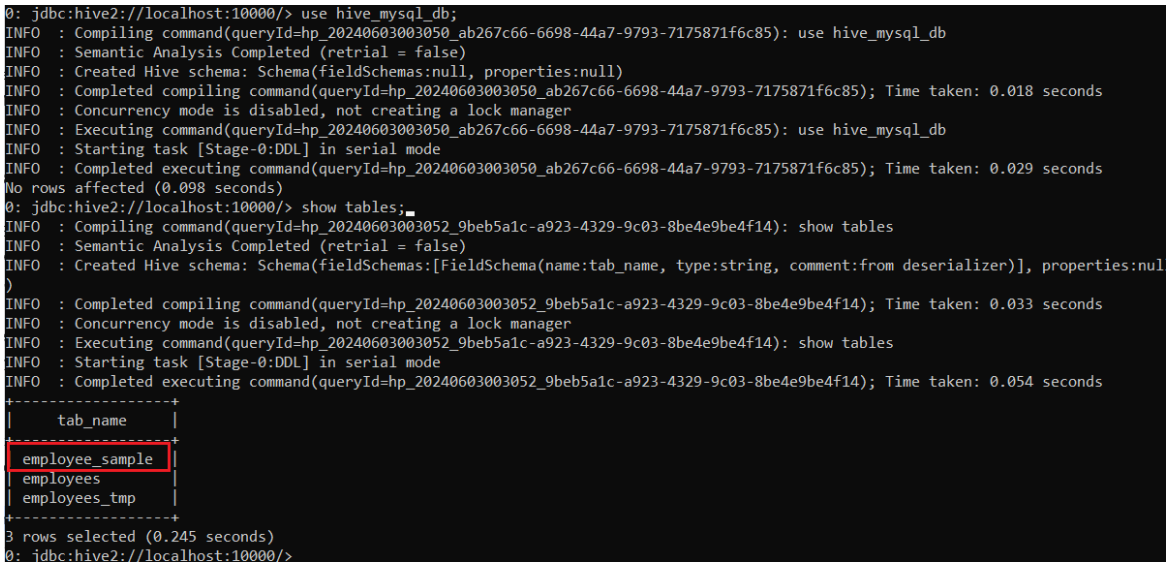
Run the following `hcat` command to create `employee_sample` table in `hive_mysql_db` database:

```
./hcat -e 'create table hive_mysql_db.employee_sample(emp_id int, emp_name string)'
```



We can see the newly created table in Beeline CLI by executing these queries.

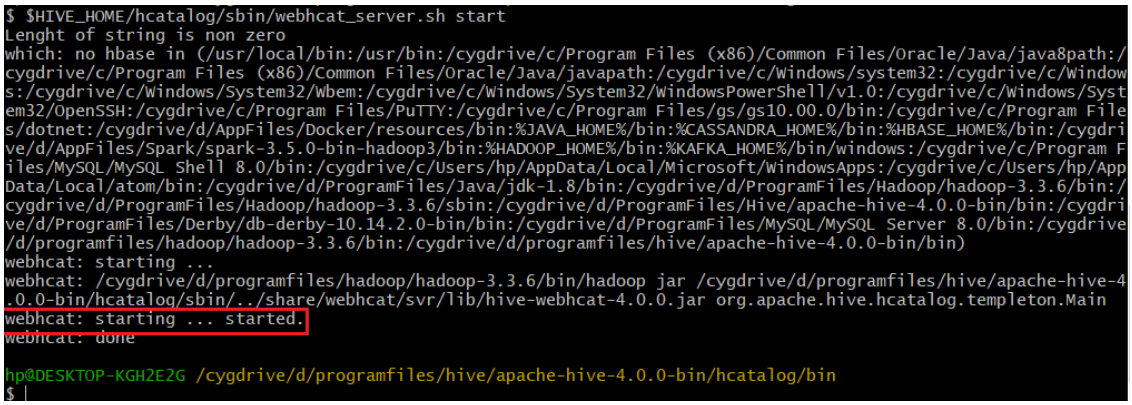
```
use hive_mysql_db;
show tables;
```



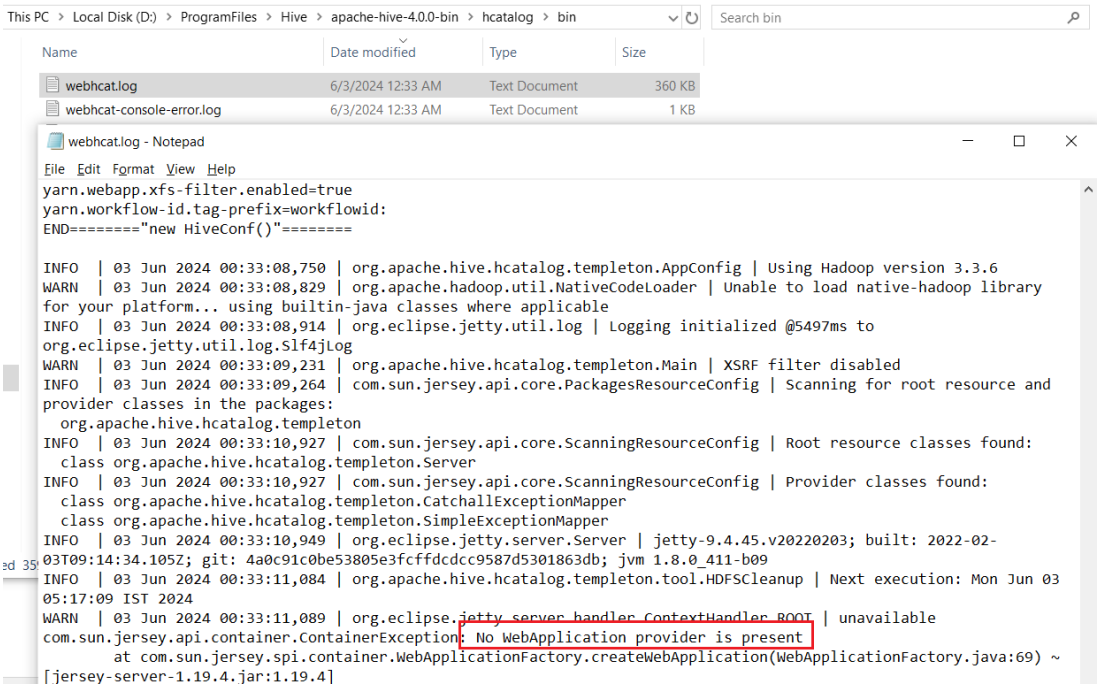
11.4. Start WebHCat server:

Now, start the **WebHCat** server using the following command in Cygwin:

```
$HIVE_HOME/hcatalog/sbin/webhcat_server.sh start
```



Though it says “webchat started” on console, but in `webchat.log` (available in the location from where ever `webhcat_server.sh` script was executed), there is an error “Server failed to start: No WebApplication provider is present”.



This is because some configuration is needed before starting WebHCat server. Follow the Apache Hive documentation on how to setup WebHCat.

Once WebHCat service is successfully running, HCatlog resources can be accessed by REST APIs using the URI format: http://localhost:50111/templeton/v1/<resource> *(For example: http://localhost:50111/templeton/v1/status)*

Congratulations!! You have now successfully installed and configured Hive 4.x with 3 metastore modes in Windows operating system. You also got a glimpse of HCatlog and WebHCat components.