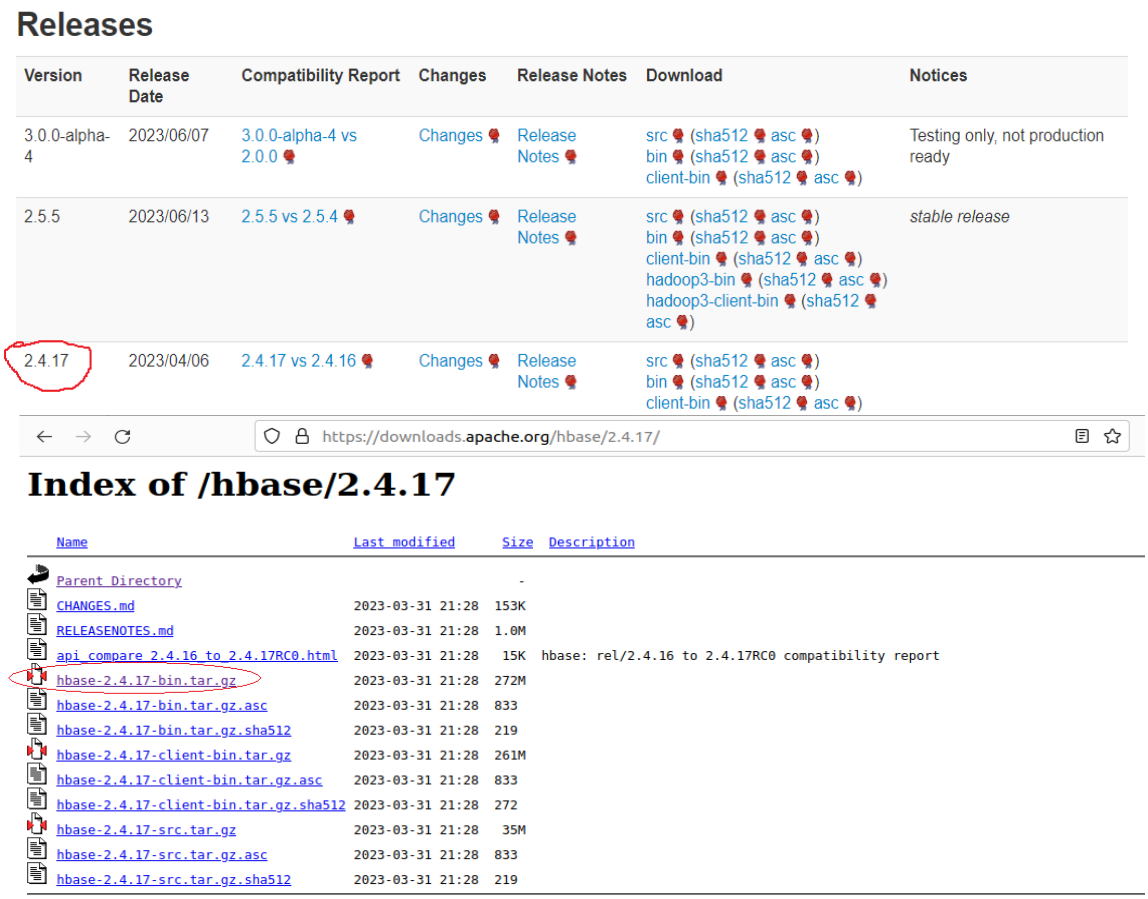
Tutorial 6
Apache HBase

Before starting this tutorial 6, make sure the Hadoop distributed file system is operating perfectly on your virtual machine (VM) and that you have a sound understanding of Tutorials **1**, **2**, and **3** (check the working of all five processes of Hadoop by using the command **$jps**). The Ubuntu operating system's terminal can be used to run these commands, but the screenshots have been provided to aid in understanding some specific steps. Please check the results of each command following execution based on the conceptual understanding presented in the lecture. Following the execution of the commands in the Ubuntu terminal, some information, such as the time, dates, or username, or output messages might be different on your screen. You can get the help of any command on Linux terminal in Ubuntu OS using **$man command.** All these commands are taken from https://hbase.apache.org and you can explore further details from hbase official website.

Part 1: Install & Test HBase

- Start your Oracle VM and login into the system using **`hduser`** and download Apache HBase using Mozilla Firefox browser using the link mentioned in the next step.
- Download the latest stable release of HBase from the Apache HBase release page
- **https://dlcdn.apache.org/hbase/2.4.17/hbase-2.4.17-bin.tar.gz**



Releases

Version	Release Date	Compatibility Report	Changes	Release Notes	Download	Notices
3.0.0-alpha-4	2023/06/07	3.0.0-alpha-4 vs 2.0.0	Changes	Release Notes	src (sha512 asc) bin (sha512 asc) client-bin (sha512 asc)	Testing only, not production ready
2.5.5	2023/06/13	2.5.5 vs 2.5.4	Changes	Release Notes	src (sha512 asc) bin (sha512 asc) client-bin (sha512 asc) hadoop3-bin (sha512 asc) hadoop3-client-bin (sha512 asc)	*stable release*
2.4.17	2023/04/06	2.4.17 vs 2.4.16	Changes	Release Notes	src (sha512 asc) bin (sha512 asc) client-bin (sha512 asc)	

https://downloads.apache.org/hbase/2.4.17/

Index of /hbase/2.4.17

Name	Last modified	Size	Description
Parent Directory		-	
CHANGES.md	2023-03-31 21:28	153K	
RELEASENOTES.md	2023-03-31 21:28	1.0M	
api_compare_2.4.16_to_2.4.17RC0.html	2023-03-31 21:28	15K	hbase: rel/2.4.16 to 2.4.17RC0 compatibility report
hbase-2.4.17-bin.tar.gz	2023-03-31 21:28	272M	
hbase-2.4.17-bin.tar.gz.asc	2023-03-31 21:28	833	
hbase-2.4.17-bin.tar.gz.sha512	2023-03-31 21:28	219	
hbase-2.4.17-client-bin.tar.gz	2023-03-31 21:28	261M	
hbase-2.4.17-client-bin.tar.gz.asc	2023-03-31 21:28	833	
hbase-2.4.17-client-bin.tar.gz.sha512	2023-03-31 21:28	272	
hbase-2.4.17-src.tar.gz	2023-03-31 21:28	35M	
hbase-2.4.17-src.tar.gz.asc	2023-03-31 21:28	833	
hbase-2.4.17-src.tar.gz.sha512	2023-03-31 21:28	219	

1) Unzip the downloaded file using the **`tar`** command (unzip folder) in the **`hduser`** user. By default, the file is downloaded to "Download" folder in Ubuntu operating system. You navigate to the "Download" folder by using the following commands on ubuntu terminal.

```
$cd /home/hduser/Downloads
$sudo tar xvf hbase-2.4.17-bin.tar.gz
```

2) Move the unzipped file (**hbase-2.4.17**) into the **/usr/local** directory by using the following command as mentioned below

```
$sudo mv /home/hduser/Downloads/hbase-2.4.17 /usr/local/
```

Change the folder by using cd command for deployment of HBase.

```
$cd /usr/local
```

3) Create a symbolic link called **hbase** to the **hbase-2.4.17** directory in the **/usr/local** directory. This symbolic link helps to use hbase commands in a simplified way.

```
$sudo ln -sf /usr/local/hbase-2.4.17 /usr/local/hbase
```

If this command does not work, you can get help from the Hadoop installation tutorial as you did this step in the previous tutorial 2.

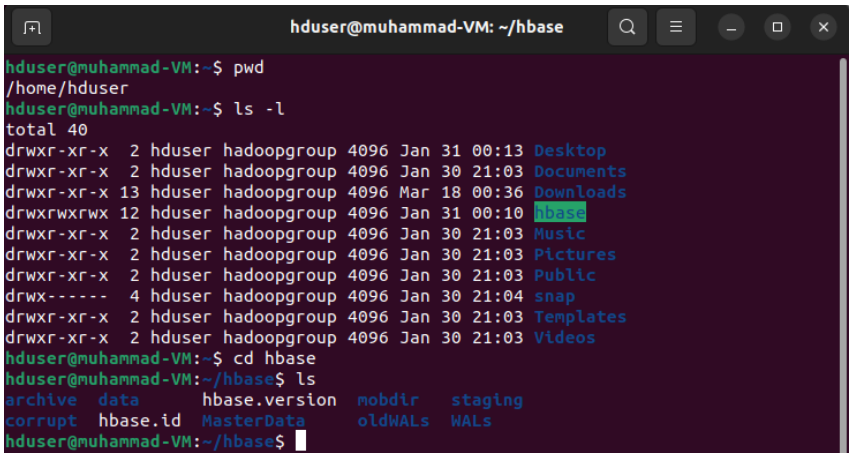
4) Change the ownership of the files in the **hbase** directory so that the group is assigned to **hadoop** and the owner is **hduser**:

```
$sudo chown -R hduser:hadoopgroup hbase
$sudo chown -R hduser:hadoopgroup hbase-2.4.17
```

You are deploying hbase in hduser and it is essential to provide all privileges for proper functioning of the hbase operations.

* The above commands should be executed in the folder **/usr/local**

5) We need to create a folder named "**hbase**" for the data handling as shown in the screenshot below with green color shade. For this purpose, create a new Directory/ folder named as '**hbase**' at location **/home/hbase** for Apache HBase database usage



a) Create the **/home/hduser/hbase** directory by using the following commands.

```
$cd /home/hduser/
$sudo mkdir /home/hduser/hbase
```

This

b) Change permissions of the above **hbase** directory so that the group is assigned to

hadoopgroup and the owner is **hduser**

```
$sudo chown -R hduser:hadoopgroup hbase
$sudo chmod 777 hbase
```

c) Edit the **/usr/local/hbase/conf/hbase-site.xml** file using **nano** editor

```
$nano /usr/local/hbase/conf/hbase-site.xml
```

and update **hbase-site.xml** by adding the property as follows inside the **<configuration> </configuration>** tags at the top along with other properties. Do not change any other property in this file. This folder will be used during the processing of HBase operations and cluster information.

```
<configuration>
    <property>
        <name>hbase.rootdir</name>
        <value>file:///home/hduser/hbase</value>
    </property>
</configuration>
```

We provided information to hbase that the data processing will be performed on the local machine by using this path `file:///home/hduser/hbase`.

6) Modify the **/usr/local/hbase/conf/hbase-env.sh** file to update the **JAVA_HOME** environment variable by opening the file '**hbase-env.sh**' using **nano** editor.

```
$nano /usr/local/hbase/conf/hbase-env.sh
```

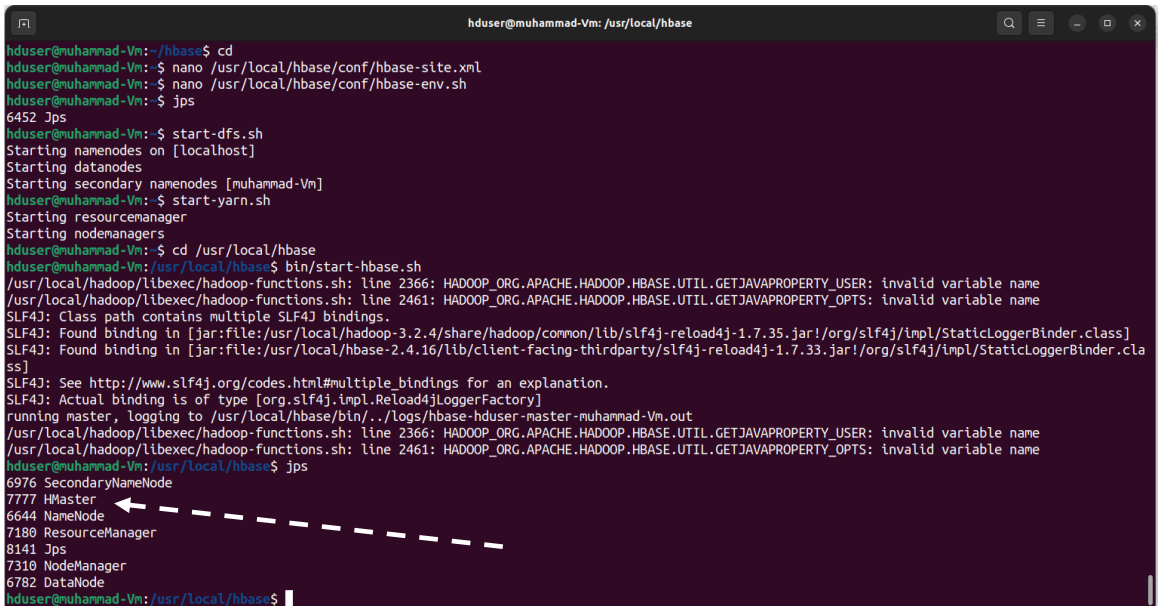
Add the following lines at the end of the file

```
# The java implementation to use.  Java 1.8+ required.
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
# Native library
export LD_LIBRARY_PATH=/usr/local/hadoop/lib/native
```

HBase requires Java to run, and the line (JAVA_HOME) ensures that the correct version of Java is used. The next command sets the LD_LIBRARY_PATH environment variable to include the directory where the native libraries required by HBase are located.

7) Start hadoop (use the commands, such as **start-dfs.sh** and **start-yarn.sh**) before launching **Hbase**. Start **HBase** with the command: **/usr/local/hbase/bin/start-hbase.sh** and the following screenshot will show the execution of all steps.

If HBase started correctly, you can find an additional process '**HMaster**' using jps command as shown by dashed arrow.



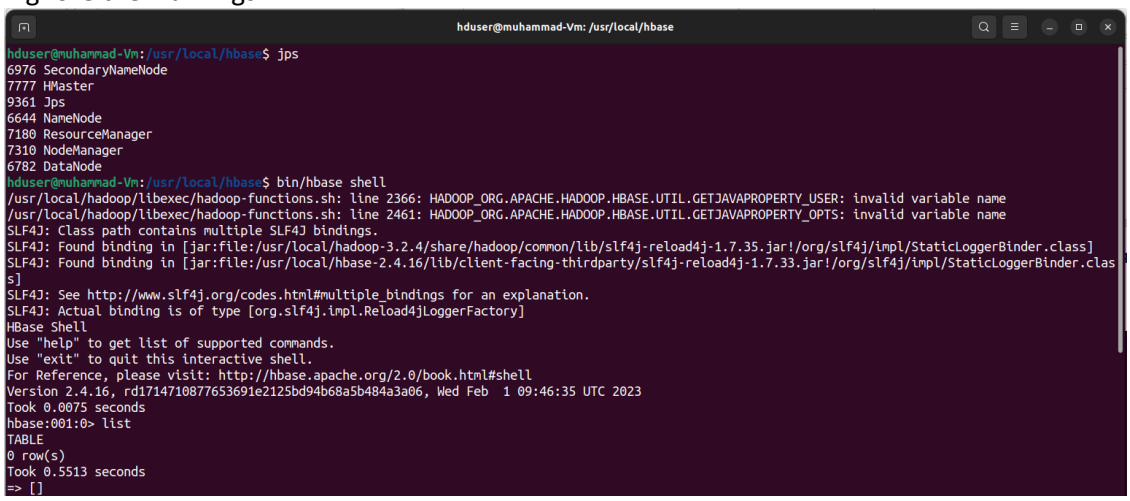
8) Launch **HBase** shell and check the status using the status command.

$cd /usr/local/hbase

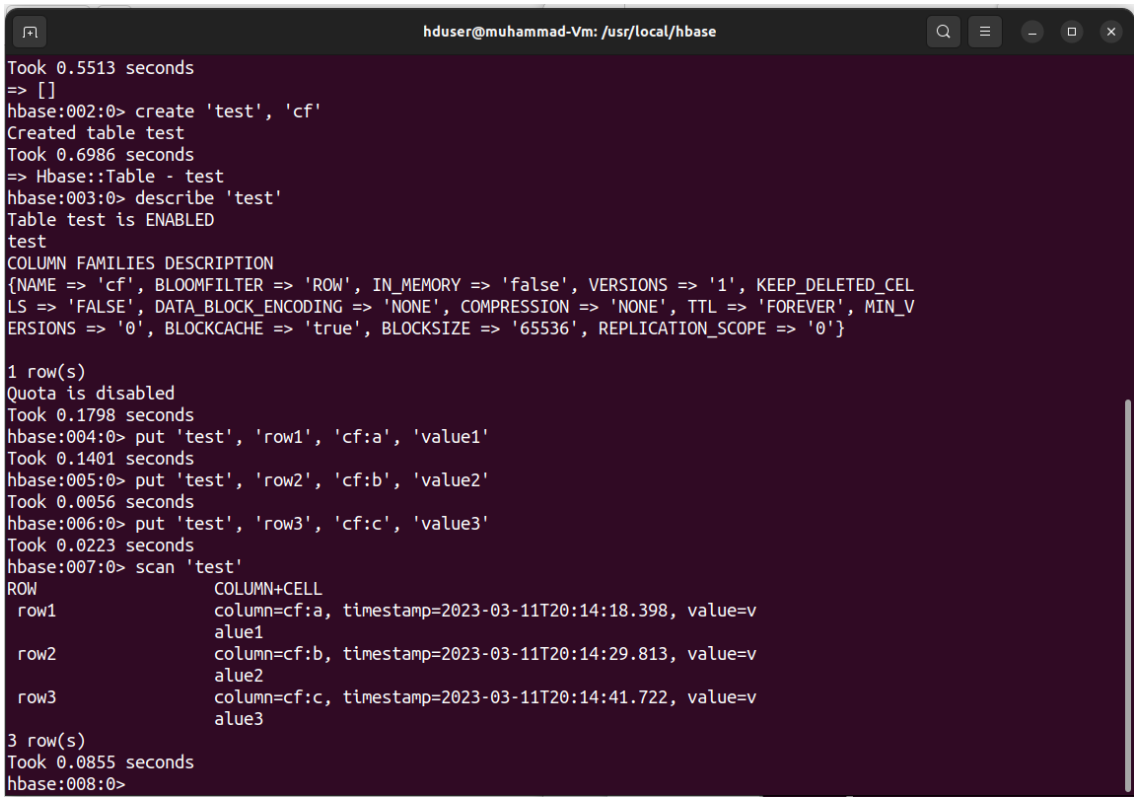
$bin/start-hbase.sh

$bin/hbase shell

The following screenshot will show the execution of all commands in the sequence and ignore the warnings.



This screen shot is the continuing part of the previous screen shot.



9) How to create a table in Hbase? Explanation of the above screen shot commands.

Use the create command to create a new table. You must specify the table name and the ColumnFamily name.

```
hbase:001:0> list
TABLE
0 row(s)
Took 0.2719 seconds
=> []
hbase:002:0> create 'test', 'cf'
Created table test
Took 0.6784 seconds
=> Hbase::Table - test
hbase:003:0>
```

10) List Information about your Table. Use the list command to confirm your table exists

```
hbase:003:0> list 'test'
TABLE
test
1 row(s)
Took 0.0268 seconds
=> ["test"]
hbase:004:0>
```

11) Now use the describe command to see the details, including configuration defaults

```
hbase:004:0> describe 'test'
Table test is ENABLED
test
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf', BLOOMFILTER => 'ROW', IN_MEMORY => 'false', VERSIONS => '1', KEEP_DELETED_CELLS => 'F
ALSE', DATA_BLOCK_ENCODING => 'NONE', COMPRESSION => 'NONE', TTL => 'FOREVER', MIN_VERSIONS => '0',
BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}

1 row(s)
Quota is disabled
Took 0.1282 seconds
hbase:005:0>
```

12) Insert the data into your table by using put command.

```
hbase:005:0> put 'test', 'row1', 'cf:a', 'value1'
Took 0.0882 seconds
hbase:006:0> put 'test', 'row2', 'cf:b', 'value2'
Took 0.0048 seconds
hbase:007:0> put 'test', 'row3', 'cf:c', 'value3'
Took 0.0072 seconds
hbase:008:0>
```

We inserted three values, one at a time. The first insert is at **row1**, column **cf:a**, with a value of **value1**. Columns in HBase are comprised of a column family prefix, **cf** in this example, followed by a colon and then a column qualifier suffix, **a** in this case.

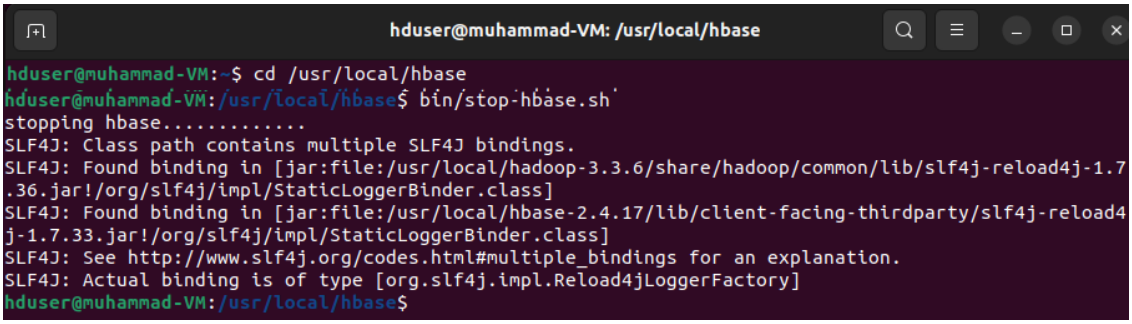
13) Scan the table for all data at once. One of the ways to get data from HBase is to scan. Use the scan command to scan the table for data. You can limit your scan, but for now, all data is fetched.

```
hbase:008:0> scan 'test'
ROW                     COLUMN+CELL
 row1                   column=cf:a, timestamp=2024-03-18T21:56:52.628, value=value1
 row2                   column=cf:b, timestamp=2024-03-18T21:57:03.915, value=value2
 row3                   column=cf:c, timestamp=2024-03-18T21:57:15.601, value=value3
3 row(s)
Took 0.0324 seconds
hbase:009:0>
```

14) To drop (delete) a table, use the disable command first and then drop command to delete the table from hbase database.

```
hbase:009:0> disable 'test'
Took 0.3523 seconds
hbase:010:0> drop 'test'
Took 0.1370 seconds
hbase:011:0> list
TABLE
0 row(s)
Took 0.0049 seconds
=> []
hbase:012:0>
```

Stop hbase using the command as mentioned below



This process will a little while depending on your system speed.

Part 2: Load a data from csv file to HBASE table

First, create the following **employees.csv** file on ubuntu operating system and enter the three lines of data (csv format) as mentioned below

Move to the Desktop folder/ directory by using the following command in step 1

1) **`$cd /home/hduser/Desktop`**

The open a new file named as **employee.csv** and add the three lines of data as shown in step 2

2) **`$nano employees.csv`**

Add the below mentioned three lines into 'employees.csv' file. Save the text and close the nano editor.

1,Lucy,Engineering
2,Milton,Engineering
3,Edith,Support

3) Save the data into **employees.csv** file using **nano/gedit** editor on Ubuntu Desktop. We copy this file to Hadoop distributed file system by using the following commands. We load (employees.csv) file into **hdfs** using **put** command as you have used already. Open a new terminal

```
$cd /home/hduser/Desktop
$hadoop fs -mkdir /user1
$hadoop fs -ls /
$hadoop fs -put ./employees.csv /user1
$hadoop fs -ls /user1
```

```
hduser@muhammad-VM:~/Desktop$ nano employees.csv
hduser@muhammad-VM:~/Desktop$ hadoop fs -put ./employees.csv /user1
hduser@muhammad-VM:~/Desktop$ hadoop fs -ls /user1
-rw-r--r--   1 hduser supergroup         56 2022-03-05 01:12 /user1/employees.csv
hduser@muhammad-VM:~/Desktop$
```

4) Start the **HBase** shell on another terminal and create a new blank table called **employees**.

```
$cd /usr/local/hbase
$jps
```

jps must show the presence of HMaster

```
hduser@muhammad-VM:~$ cd /usr/local/hbase
hduser@muhammad-VM:/usr/local/hbase$ bin/start-hbase.sh
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop-3.3.6/share/hadoop/common/lib/slf4j-reload4j-1.7.3
6.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hbase-2.4.17/lib/client-facing-thirdparty/slf4j-reload4j-
1.7.33.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Reload4jLoggerFactory]
running master, logging to /usr/local/hbase/bin/../logs/hbase-hduser-master-muhammad-VM.out
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop-3.3.6/share/hadoop/common/lib/slf4j-reload4j-1.7.3
6.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hbase-2.4.17/lib/client-facing-thirdparty/slf4j-reload4j-
1.7.33.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Reload4jLoggerFactory]
hduser@muhammad-VM:/usr/local/hbase$ jps
4613 SecondaryNameNode
4790 ResourceManager
4392 DataNode
4264 NameNode
4920 NodeManager
7705 Jps
7322 HMaster
hduser@muhammad-VM:/usr/local/hbase$ bin/hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop-3.3.6/share/hadoop/common/lib/slf4j-reload4j-1.7.3
6.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hbase-2.4.17/lib/client-facing-thirdparty/slf4j-reload4j-
1.7.33.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Reload4jLoggerFactory]
HBase Shell
Use "help" to get list of supported commands.
Use "exit" to quit this interactive shell.
For Reference, please visit: http://hbase.apache.org/2.0/book.html#shell
Version 2.4.17, r7fd096f39b4284da9a71da3ce67c48d259ffa79a, Fri Mar 31 18:10:45 UTC 2023
Took 0.0032 seconds
hbase:001:0>
```

Now we create a Table named as **employees** in hbase.

```
hbase:001:0>create 'employees', 'cf'
hbase:001:0>list
```

```
hbase:001:0> list
TABLE
0 row(s)
Took 0.2731 seconds
=> []
hbase:002:0> create 'employees', 'cf'
Created table employees
Took 0.6911 seconds
=> Hbase::Table - employees
hbase:003:0> list
TABLE
employees
1 row(s)
Took 0.0053 seconds
=> ["employees"]
hbase:004:0>
```

5) Use ImportTsv to load data from **HDFS (/user1/employees.csv)** into the HBase table created in the previous step. [**Type this command** and do not copy – paste to terminal please]

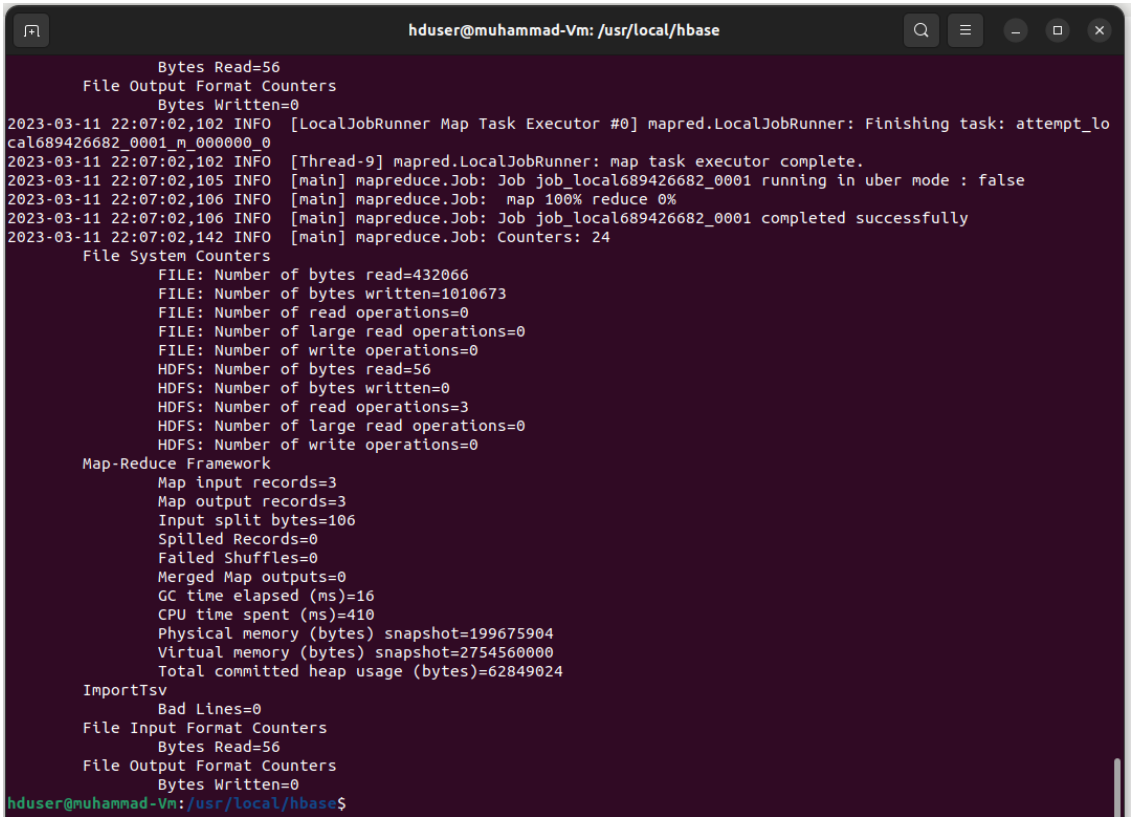
```
$cd /usr/local/hbase
```

```
$bin/hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator=',' -Dimporttsv.columns=HBASE_ROW_KEY,cf:name,cf:department employees hdfs:///user1/employees.csv
```

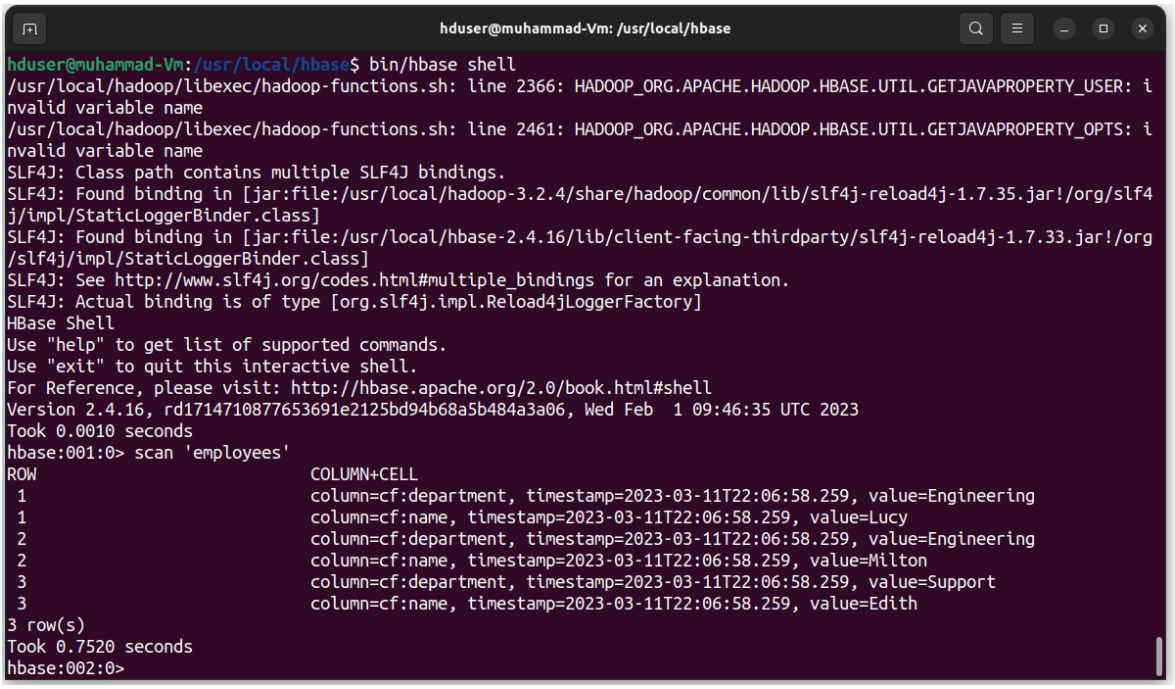
Please check the screenshot below before the execution of the above command as there should not be any page break in writing the above command. For your implementation in the case of continuous assessment, you can use any csv file and

```
hduser@muhammad-Vm: /usr/local/hbase
hduser@muhammad-Vm:/usr/local/hbase$ bin/hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator=','
-Dimporttsv.columns=HBASE_ROW_KEY,cf:name,cf:department employees hdfs:///user1/employees.csv
/usr/local/hadoop/libexec/hadoop-functions.sh: line 2366: HADOOP_ORG.APACHE.HADOOP.HBASE.UTIL.GETJAVAPROPERTY_USER: i
nvalid variable name
/usr/local/hadoop/libexec/hadoop-functions.sh: line 2461: HADOOP_ORG.APACHE.HADOOP.HBASE.UTIL.GETJAVAPROPERTY_OPTS: i
nvalid variable name
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/hadoop-3.2.4/share/hadoop/common/lib/slf4j-reload4j-1.7.35.jar!/org/slf4
j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/hbase-2.4.16/lib/client-facing-thirdparty/slf4j-reload4j-1.7.33.jar!/org
/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Reload4jLoggerFactory]
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:zo
okeeper.version=3.5.7-f0fdd52973d373ffd9c86b81d99842dc2c7f660e, built on 02/10/2020 11:30 GMT
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:ho
st.name=muhammad-Vm
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:ja
va.version=1.8.0_352
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:ja
va.vendor=Private Build
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:ja
va.home=/usr/lib/jvm/java-8-openjdk-amd64/jre
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: p-yarn-server-timelin
e-pluginstorage-3.2.4.jar:/usr/local/hadoop/share/hadoop/yarn/hadoop-yarn-client-3.2.4.jar:/usr/local/hadoop/share/ha
doop/yarn/hadoop-yarn-submarine-3.2.4.jar:/usr/local/hadoop/share/hadoop/yarn/hadoop-yarn-server-web-proxy-3.2.4.jar:
/usr/local/hadoop/share/hadoop/yarn/hadoop-yarn-registry-3.2.4.jar:/usr/local/hadoop/share/hadoop/yarn/hadoop-yarn-se
rver-router-3.2.4.jar:/usr/local/hadoop/share/hadoop/yarn/hadoop-yarn-server-applicationhistoryservice-3.2.4.jar:/usr
/local/hadoop/share/hadoop/yarn/hadoop-yarn-server-resourcemanager-3.2.4.jar:/usr/local/hadoop/share/hadoop/yarn/hado
op-yarn-server-common-3.2.4.jar:/usr/local/hadoop/share/hadoop/yarn/hadoop-yarn-services-api-3.2.4.jar:/usr/local/had
oop/share/hadoop/yarn/hadoop-yarn-server-tests-3.2.4.jar:/usr/local/hadoop/share/hadoop/yarn/hadoop-yarn-common-3.2.4
.jar:/usr/local/hadoop/share/hadoop/yarn/hadoop-yarn-server-nodemanager-3.2.4.jar:/usr/local/hbase/bin/../lib/client-
facing-thirdparty/slf4j-reload4j-1.7.33.jar
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:ja
va.library.path=/usr/local/hadoop/lib/native
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:ja
va.io.tmpdir=/tmp
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:ja
va.compiler=<NA>
2023-03-11 22:06:58,610 INFO  [ReadOnlyZKClient-127.0.0.1:2181@0x3cd3e762] zookeeper.ZooKeeper: Client environment:os
```

There is some information present between these screenshots, and we provided the screenshot just after the execution of command and when the processing is finished.

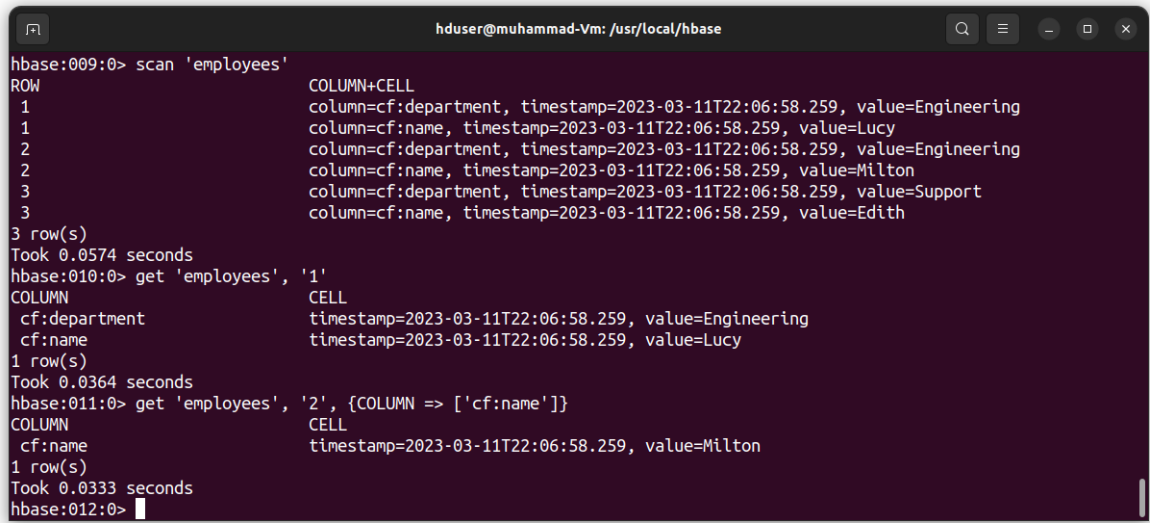


6) Return to the **HBase shell (bin/hbase shell** command**)** as discussed in the previous steps and run the following command to make sure the data in **employees.csv** file was loaded into the HBase table or not. The data from hadoop distributed file system is transferred to HBase table named as “employees”.



7) How the get the data from the table, ‘employees’ created in HBase? Follow the commands as mentioned below to get the first row of the table and second row with a specific column.

```
hbase>get 'employees', '1'
hbase>get 'employees', '2', {COLUMN => ['cf:name']}
```



You can create a table based on your own data file and load into NoSQL HBase database for your project. Further practice for table creation using HBase is available at the web link: **https://hbase.apache.org/book.html#_table**. Further understanding about HBase can be obtained from the following links mentioned in the references.

8) In the same way that the **bin/start-hbase.sh** script is provided to conveniently start all HBase daemons (A daemon is a program that runs continuously and exists for the purpose of handling periodic service requests that a computer system expects to receive.), the bin/stop-hbase.sh script stops them.

Now we follow a procedure to load data from a datafile from the local VM drive to Hadoop (hdfs). Further we load the data into HBASE table.

Note: You must stop HBase state before shutting down of your VM, otherwise, you will face problems next time when you start HBase.

References:

- https://hbase.apache.org/
- https://docs.cloudera.com/cdsw/1.9.0/import-data/topics/cdsw-load-data-into-hbase-table.html
- https://www.cloudduggu.com/hbase/data-model/
- https://www.guru99.com/create-read-data-in-hbase.html