

# HBase findings and trials

## Different ways to use HBase:

HBase is an open source NoSQL distributed database, we don't need a specific schema for the storage. From our investigation on materials present on HBase, there are different ways we could use HBase in our project: We can

1. Use it locally, by installing it.
2. We can use the AWS EMR HBase service.
3. We can use the plain EC2 instance and install HBase on top of it.

## **Using Hbase Locally:**

For local installation we have below options:

- Standalone mode installation (No dependency on Hadoop system)
- Pseudo-Distributed mode installation (Single node Hadoop system + HBase installation)
- Fully Distributed mode installation (Multinode Hadoop environment + HBase installation)

We tried the Standalone installation mode, which requires us to follow below steps:

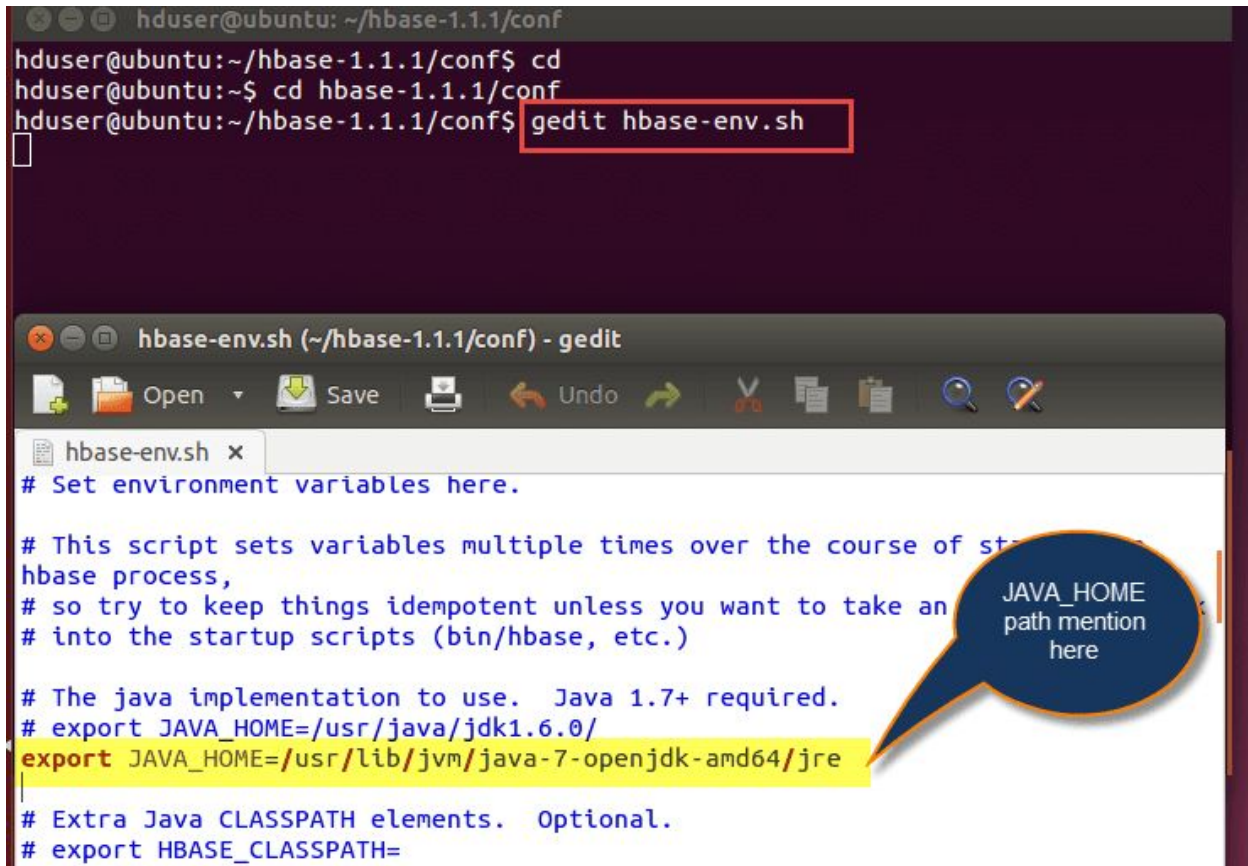
Installation is performed on Ubuntu with Hadoop already installed.

**Step 1)** Place hbase-x.x.x-bin.tar.gz in /home/hduser

**Step 2)** Unzip by executing command `$tar -xvf hbase-x.x.x-bin.tar.gz`. It will unzip the contents, and it will create hbase-x.x.x in the location /home/hduser

**Step 3)** Open hbase-env.sh as below and mention JAVA\_HOME path in the location.

```
hduser@ubuntu: ~/hbase-1.1.1/conf
hduser@ubuntu:~/hbase-1.1.1/conf$ cd
hduser@ubuntu:~$ cd hbase-1.1.1/conf
hduser@ubuntu:~/hbase-1.1.1/conf$ gedit hbase-env.sh
```



```
# Set environment variables here.

# This script sets variables multiple times over the course of starting
# hbase process,
# so try to keep things idempotent unless you want to take advantage of it
# into the startup scripts (bin/hbase, etc.)

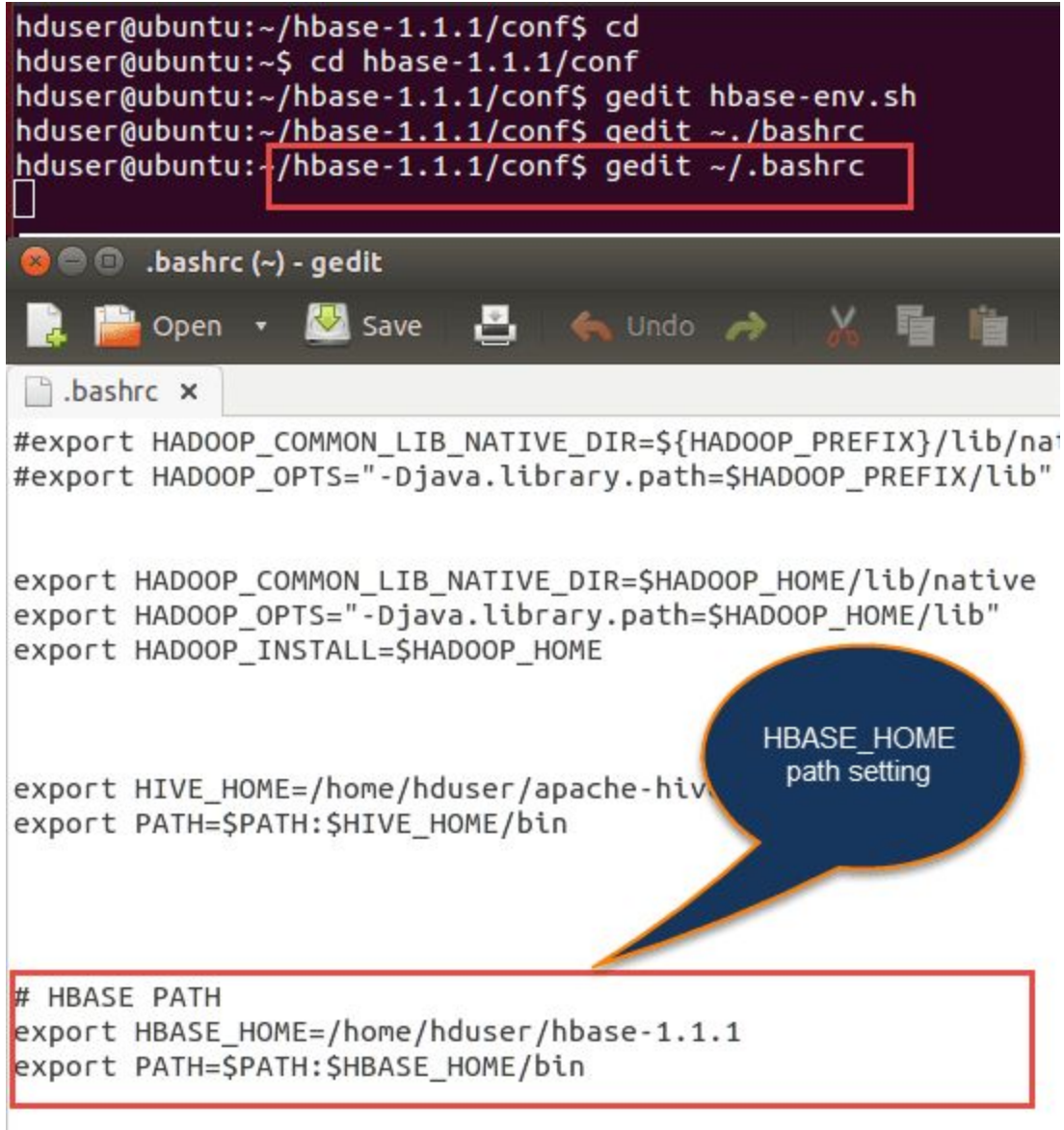
# The java implementation to use.  Java 1.7+ required.
# export JAVA_HOME=/usr/java/jdk1.6.0/
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64/jre

# Extra Java CLASSPATH elements.  Optional.
# export HBASE_CLASSPATH=
```

**Step 4)** Open ~/.bashrc file and mention HBASE\_HOME path as shown in below

```
export HBASE_HOME=/home/hduser/hbase-x.x.x export PATH=
$PATH:$HBASE_HOME/bin
```

```
hduser@ubuntu:~/hbase-1.1.1/conf$ cd
hduser@ubuntu:~$ cd hbase-1.1.1/conf
hduser@ubuntu:~/hbase-1.1.1/conf$ gedit hbase-env.sh
hduser@ubuntu:~/hbase-1.1.1/conf$ gedit ~/.bashrc
hduser@ubuntu:~/hbase-1.1.1/conf$ gedit ~/.bashrc
```



```
#export HADOOP_COMMON_LIB_NATIVE_DIR=${HADOOP_PREFIX}/lib/native
#export HADOOP_OPTS="-Djava.library.path=$HADOOP_PREFIX/lib"

export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
export HADOOP_INSTALL=$HADOOP_HOME

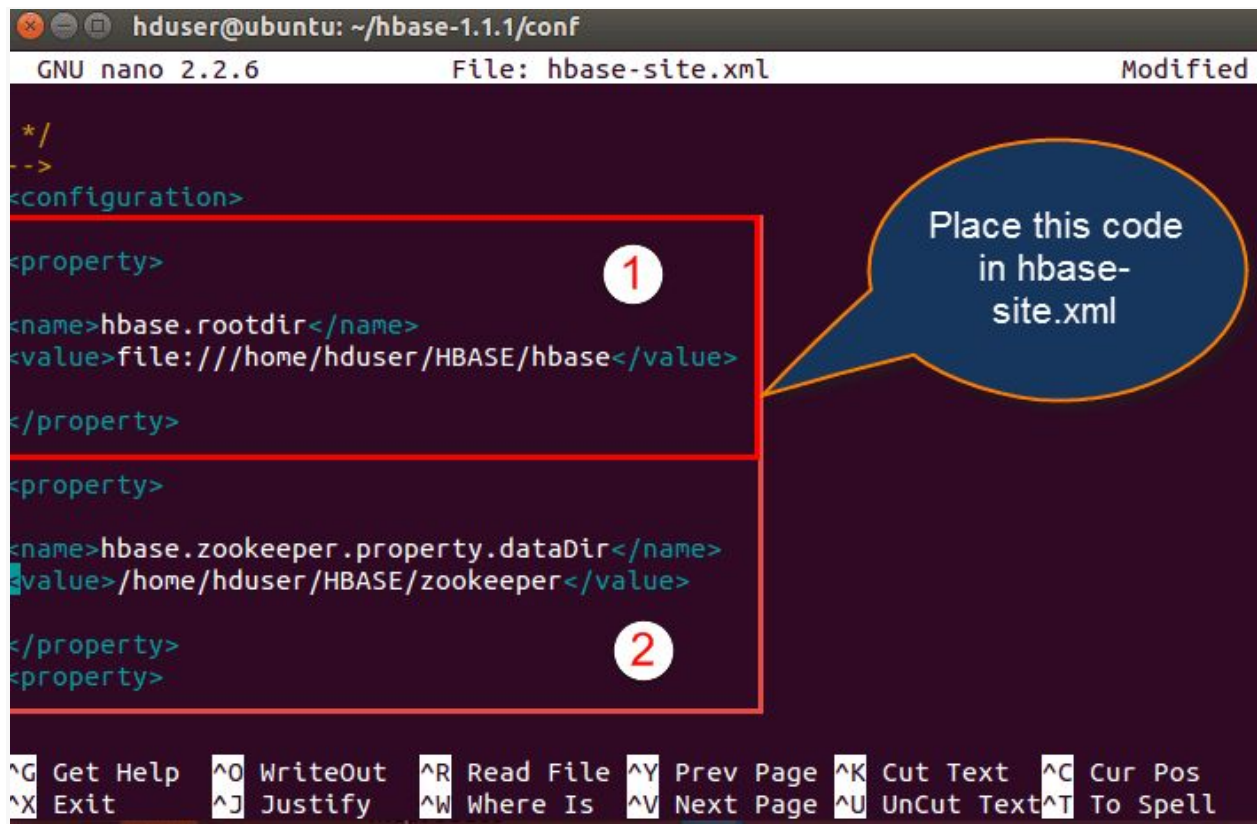
export HIVE_HOME=/home/hduser/apache-hive
export PATH=$PATH:$HIVE_HOME/bin

# HBASE PATH
export HBASE_HOME=/home/hduser/hbase-1.1.1
export PATH=$PATH:$HBASE_HOME/bin
```

**Step 5)** Open hbase-site.xml and place the following properties inside the file

ubuntu\$ gedit hbase-site.xml(code as below)

```
<property>
<name>hbase.rootdir</name>
<value>file:///home/hduser/HBASE/hbase</value>
</property>
<property>
<name>hbase.zookeeper.property.dataDir</name>
<value>/home/hduser/HBASE/zookeeper</value>
</property>
```



```
hduser@ubuntu: ~/hbase-1.1.1/conf
GNU nano 2.2.6 File: hbase-site.xml Modified

*/
-->
<configuration>

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

<property>
<name>hbase.zookeeper.property.dataDir</name>
<value>/home/hduser/HBASE/zookeeper</value>
</property>
<property>

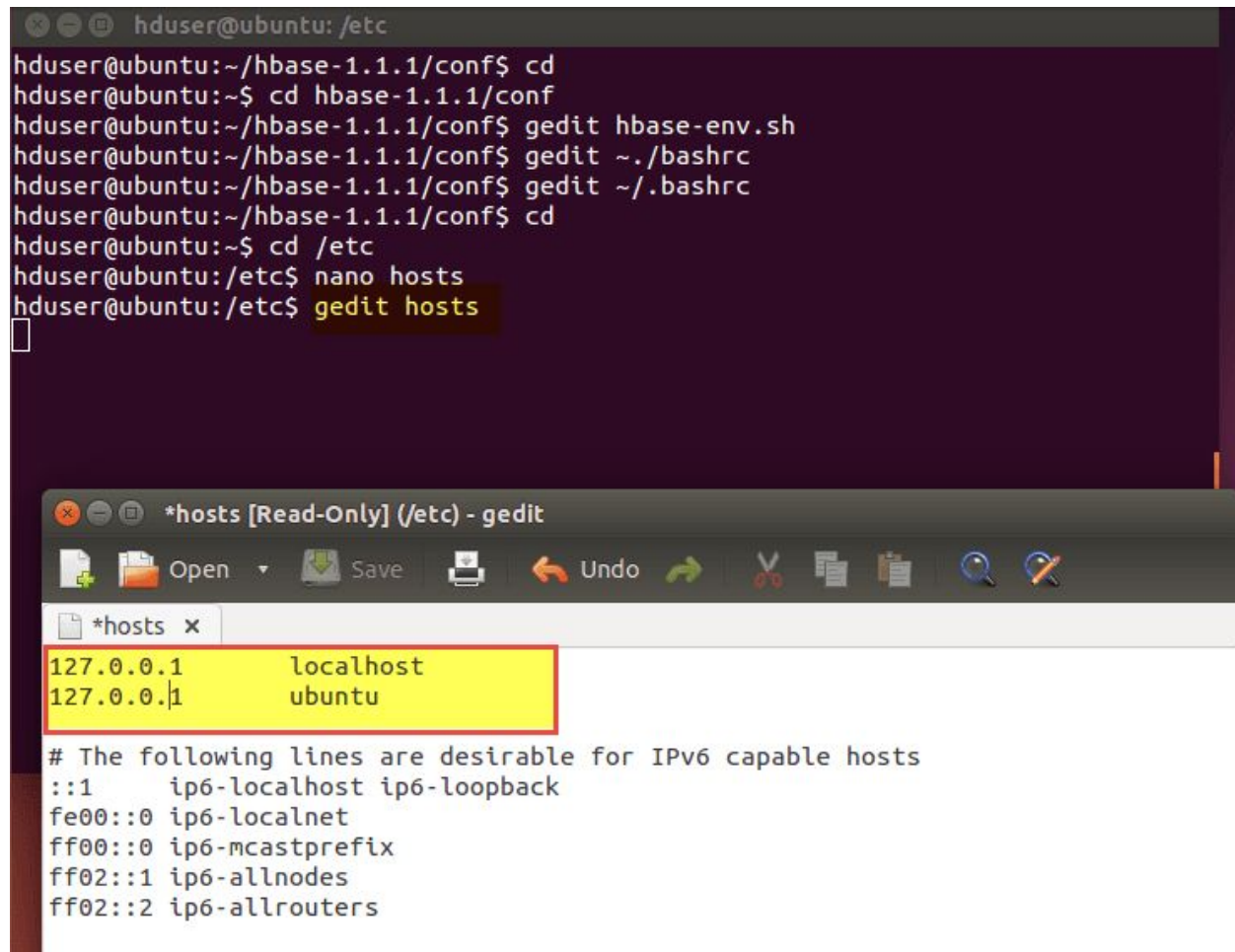
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Here we are placing two properties

- One for HBase root directory and
- Second one for the data directory corresponds to ZooKeeper.

All HMaster and ZooKeeper activities point out to this hbase-site.xml.

**Step 6)** Open hosts file present in /etc. location and mention the IPs as shown in below.



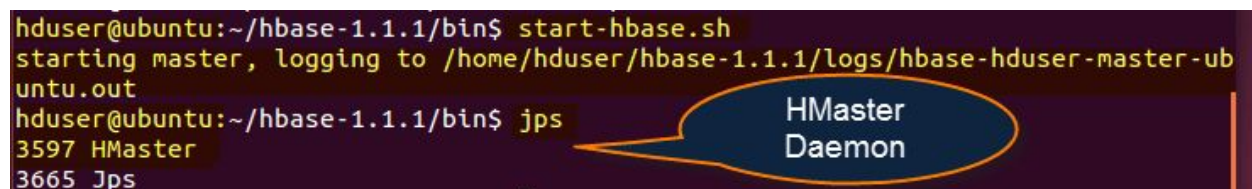
The image shows a terminal window and a nano editor window. The terminal window shows the user navigating to the /etc directory and opening the hosts file with nano. The nano editor window shows the contents of the /etc/hosts file, which includes the following lines:

```
127.0.0.1    localhost
127.0.0.1    ubuntu

# The following lines are desirable for IPv6 capable hosts
::1         ip6-localhost ip6-loopback
fe00::0     ip6-localnet
ff00::0     ip6-mcastprefix
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters
```

**Step 7)** Now Run Start-hbase.sh in hbase-x.x.x/bin location as shown below.

And we can check by jps command to see whether HMaster is running or not.



The image shows a terminal window with the following commands and output:

```
hduser@ubuntu:~/hbase-1.1.1/bin$ start-hbase.sh
starting master, logging to /home/hduser/hbase-1.1.1/logs/hbase-hduser-master-ubuntu.out
hduser@ubuntu:~/hbase-1.1.1/bin$ jps
3597 HMaster
3665 Jps
```

A blue speech bubble points to the HMaster process, indicating it is the HMaster Daemon.



**Step8)** HBase shell can start by using "hbase shell" and it will enter into interactive shell mode as shown in the screenshot below. Once it enters shell mode, we can perform all types of commands.

```
rahul@rahul-inspiron-7586:~/hduser/hbase-2.2.6/bin$ ./start-hbase.sh
Running master, logging to /home/rahul/hduser/hbase-2.2.6/bin/./logs/hbase-rahul-master-rahul-inspiron-7586.out
rahul@rahul-inspiron-7586:~/hduser/hbase-2.2.6/bin$ jps
5296 RemoteMavenServer36
14536 Jps
3689 Main
rahul@rahul-inspiron-7586:~/hduser/hbase-2.2.6/bin$
```

```
ubuntu: ~/hbase-1.1.1/bin
hduser@ubuntu:~/hbase-1.1.1/bin$ hbase shell
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/home/hduser/hbase-1.1.1/
SLF4J: Found binding in [jar:file:/home/hduser/hadoop-2.2.0
SLF4J: See http://www.slf4j.org/codes.html#multiple_binding
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLogge
2015-09-11 17:01:42,907 WARN [main] util.NativeCodeLoader:
HBase Shell; enter 'help<RETURN>' for list of supported com
Type "exit<RETURN>" to leave the HBase Shell
Version 1.1.1, rd0a115a7267f54e01c72c603ec53e91ec418292f, T
hbase(main):001:0> status
```

As it is seen above the standalone mode works fine, but when we try to access it at: <http://localhost:60010/> we can't see the page. We also checked the zookeeper on the port 2181, which is the default mode for the zookeeper installation but it didn't work.

So we assumed it may work in pseudo distributed mode:


We follow the steps similar to we did in case of standalone installation but changing the distributed property in HBase-site.xml as **true**.

```
<property>
<name>hbase.cluster.distributed</name>
<value>true</value>
</property>
```

Now we point to our hadoop installation and start Hadoop daemons first and after that start HBase daemons as shown below:

Here first you have to start Hadoop daemons by using `./start-all.sh` command as below.

```
hduser@ubuntu:~/hadoop-1.2.1/bin$ ./start-all.sh
hduser@ubuntu:~/hadoop-1.2.1/bin$ jps
3368 Jps
2871 DataNode
3151 JobTracker
2723 NameNode
3061 SecondaryNameNode
3321 TaskTracker
```



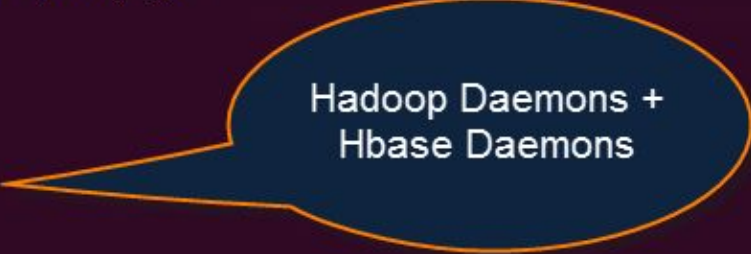
Hadoop Daemons Running here

After starting Hbase daemons by `hbase-start.sh`

```
hduser@ubuntu:~/hbase-1.1.1/bin$ start-hbase.sh
starting master, logging to /home/hduser/hbase-1.1.1/logs/hbase-hduser-master-ubuntu.out
```

Checking `jps`

```
hduser@ubuntu:~/hbase-1.1.1/bin$ jps
3776 HQuorumPeer
2871 DataNode
4067 HRegionServer
3151 JobTracker
2723 NameNode
3061 SecondaryNameNode
4142 Jps
3840 HMaster
3321 TaskTracker
```



Hadoop Daemons + Hbase Daemons

As we see the all the above steps worked perfectly fine but still we were not able to expose the hbase via zookeeper on port 2181, we followed different articles on stack overflow, blogs but didn't find something meaningful that could solve this issue. While following the above steps we encountered many errors but were able to solve them and reach the above steps at the end.

We also tried following steps mentioned in the blog:

[https://www.tutorialspoint.com/hbase/hbase\\_installation.htm](https://www.tutorialspoint.com/hbase/hbase_installation.htm)

But we encountered many errors and were not able to expose the port and access it from our java application.

After not being able to make the above steps work, we thought of leveraging the managed HBase service from AWS which comes with AWS EMR:

We followed the blog: <https://aws.amazon.com/emr/features/hbase/> and tried following the configuration from the UI as mentioned in the post:

<https://docs.aws.amazon.com/emr/latest/ReleaseGuide/emr-hbase-configure.html>

We created an EMR cluster with HBase installed as seen below:

The screenshot displays the AWS Management Console interface for an EMR cluster. At the top, there are buttons for 'Clone', 'Terminate', and 'AWS CLI export'. Below these, the cluster name 'Cluster: EMR HBASE Cluster' is shown, along with its status 'Terminated' and the reason 'Terminated by user request'. A navigation bar contains tabs for 'Summary', 'Application user interfaces', 'Monitoring', 'Hardware', 'Configurations', 'Events', 'Steps', and 'Bootstrap actions'. The 'Summary' tab is selected, showing the following details:

- ID: j-RUIEA0QJ2010
- Creation date: 2020-12-09 12:22 (UTC-5)
- End date: 2020-12-09 12:36 (UTC-5)
- Elapsed time: 13 minutes
- After last step completes: Cluster waits
- Termination protection: Off
- Tags: --
- Master public DNS: ec2-3-85-90-194.compute-1.amazonaws.com (with a copy icon)
- Connect to the Master Node Using SSH (with a link icon)

Below the summary, the 'Configuration details' section is expanded, showing:

- Release label: emr-5.17.0
- Hadoop distribution: Amazon
- Applications: Ganglia 3.7.2, HBase 1.4.6, Hive 2.3.3, Hue 4.2.0, Phoenix 4.14.0
- Log URI: s3://lspdpbase/ (with a folder icon)
- EMRFS consistent view: Disabled

And tried accessing the cluster on <http://master-public-dns-name:16010/> for the HBase service. <https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-web-interfaces.html>

But here as well we were unable to access the HBase via port 16010 or zookeeper 2181. Working on this has already consumed a significant portion of time hence we dropped the plan to further get hbase working, we could have installed it on a plain EC2 instance and get it working but as the local installation didn't work and we were not able to expose the port **2181** or **16010**.

Hence as a suggestion from the professor we opted for broadcasting the HBase data during the MapReduce jobs.