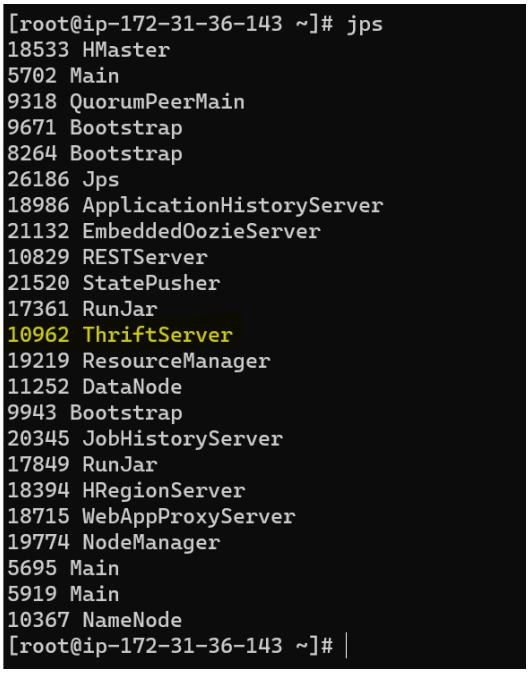
Running **HappyBase**

1. Login to your EMR instance using PuTTY in Windows or Shell command in Linux/Mac OS and switch to root user using the following command-

```
sudo -i
```

2. Make sure the thrift server is running.
    a. Run **jps** to list all the running daemons. By default, EMR should have the ThriftServer up and running already when you login to your EMR instance.



If the thrift server is not working by default, run the following command to start it

```
hbase thrift start
```

```
[root@ip-10-0-0-91 ~]# hbase thrift start
21/03/03 07:51:43 INFO util.VersionInfo: HBase 1.2.0-cdh5.15.1
21/03/03 07:51:43 INFO util.VersionInfo: Source code repository file:///data/jenkins/workspace/generic-package-centos64-7-0/topdir/B
UILD/hbase-1.2.0-cdh5.15.1 revision=Unknown
21/03/03 07:51:43 INFO util.VersionInfo: Compiled by jenkins on Thu Aug  9 09:07:41 PDT 2018
21/03/03 07:51:43 INFO util.VersionInfo: From source with checksum 1309177973f3ca5da342a75775f3edc5
21/03/03 07:51:43 INFO thrift.ThriftServerRunner: Using default thrift server type
21/03/03 07:51:43 INFO thrift.ThriftServerRunner: Using thrift server type threadpool
21/03/03 07:51:45 WARN impl.MetricsConfig: Cannot locate configuration: tried hadoop-metrics2-hbase.properties,hadoop-metrics2.prope
rties
21/03/03 07:51:45 INFO impl.MetricsSystemImpl: Scheduled snapshot period at 10 second(s).
21/03/03 07:51:45 INFO impl.MetricsSystemImpl: HBase metrics system started
21/03/03 07:51:45 INFO mortbay.log: Logging to org.slf4j.impl.Log4jLoggerAdapter(org.mortbay.log) via org.mortbay.log.Slf4jLog
21/03/03 07:51:45 INFO http.HttpRequestLog: Http request log for http.requests.thrift is not defined
21/03/03 07:51:45 INFO http.HttpServer: Added global filter 'safety' (class=org.apache.hadoop.hbase.http.HttpServer$QuotingInputFilt
er)
21/03/03 07:51:45 INFO http.HttpServer: Added global filter 'clickjackingprevention' (class=org.apache.hadoop.hbase.http.Clickjackin
gPreventionFilter)
21/03/03 07:51:45 INFO http.HttpServer: Added filter static_user_filter (class=org.apache.hadoop.hbase.http.lib.StaticUserWebFilter$
StaticUserFilter) to context thrift
21/03/03 07:51:45 INFO http.HttpServer: Added filter static_user_filter (class=org.apache.hadoop.hbase.http.lib.StaticUserWebFilter$
StaticUserFilter) to context static
21/03/03 07:51:45 INFO http.HttpServer: Added filter static_user_filter (class=org.apache.hadoop.hbase.http.lib.StaticUserWebFilter$
StaticUserFilter) to context logs
21/03/03 07:51:45 INFO http.HttpServer: Jetty bound to port 9095
21/03/03 07:51:45 INFO mortbay.log: jetty-6.1.26.cloudera.4
21/03/03 07:51:46 INFO mortbay.log: Started SelectChannelConnector@0.0.0.0:9095
21/03/03 07:51:46 INFO thrift.ThriftServerRunner: starting TBoundedThreadPoolServer on /0.0.0.0:9090 with readTimeout 60000ms; min w
orker threads=16, max worker threads=1000, max queued requests=1000
```

Verify if the thrift server has started. Run the **jps** command again

```
[root@ip-10-0-0-91 ~]# jps
8230 ThriftServer
27417 -- process information unavailable
8638 Jps
1086 Main
[root@ip-10-0-0-91 ~]#
```

Run the below command to test if the happyBase package is present or not

```
python -c "import happybase"
```

If you get an import error saying "**No module named happybase**".

```
    import happybase
ImportError: No module named happybase
```

Then you need to install **happybase** by running the command below.

```
pip install happybase
```

Running HappyBase using the following python scripts.

Note- Create the python script using the command `vi file_name.py`. To run the script use command `python file_name.py`.

1. List all the tables present in HBase.

```
#Listing table
import happybase

print("connecting to HBase")
con = happybase.Connection('localhost')


con.open()
print("Connected")

print("Listing tables...")
print(con.tables())

print("Closing the connection")
con.close()
```

Output:

```
[root@ip-10-0-0-28 basics]# python list_tables.py
connecting to HBase
Connected
Listing tables...
['Customer', 'employee', 'events', 'happyTest', 'products', 'sample', 'user']
Closing the connection
[root@ip-10-0-0-28 basics]#
```

2. Create tables in HBase.

```
#Creating tables

import happybase

print("connecting to HBase")
con = happybase.Connection('localhost')


con.open()
print("Connected")

print("Listing tables...")
print(con.tables())

print("Creating new table...")
con.create_table(
    'mytable',
    {'cf1': dict(),
     'cf2': dict(),
     'cf3': dict(),
    }
)
print("Listing tables...")
print(con.tables())

print("Closing the connection")
con.close()
```

Output:

```
[root@ip-10-0-0-28 basics]# python create_tables.py
connecting to HBase
Connected
Listing tables...
['Customer', 'employee', 'events', 'happyTest', 'products', 'sample', 'user']
Creating new table...
Listing tables...
['Customer', 'employee', 'events', 'happyTest', 'mytable', 'products', 'sample
', 'user']
Closing the connection
[root@ip-10-0-0-28 basics]#
```

3. Insert data into tables.

```
# Inserting data
import happybase

print("connecting to HBase")
con = happybase.Connection('localhost')

con.open()
print("Connected")

print("Listing tables...")
print(con.tables())

mtable = con.table('mytable')

print("Putting data into rows...")
mtable.put(b'rawat1', {b'cf1:name': b'Abhinav', b'cf2:age': b'22',
b'cf3:gender':b'M'})
mtable.put(b'rawat2', {b'cf1:name': b'Vishal', b'cf2:age': b'27',
b'cf3:gender':b'M'})

print("Closing the connection")
con.close()
```

Output:

```
[root@ip-10-0-0-28 basics]# python insert_data.py
connecting to HBase
Connected
Listing tables...
['Customer', 'employee', 'events', 'happyTest', 'mytable', 'products', 'sample
', 'user']
Putting data into rows...
Closing the connection
[root@ip-10-0-0-28 basics]#
```

4. Read data from tables.

```
#Reading data

import happybase

print("Connecting to HBase")
con = happybase.Connection('localhost')

con.open()
print("Connected")

print("Listing tables...")
print(con.tables())

print("Connecting to -> mytable...")
mtable = con.table('mytable')

print("Getting the data...")
row1 = mtable.row(b'rawat1', columns=[b'cf1:name', b'cf2:age',
b'cf3:gender'])
row2 = mtable.row(b'rawat2')

print("Printing the data...")
print("----------------------------------------")
print(str(row1[b'cf1:name']) +" "+str(row1[b'cf2:age'])+"
```

```
"+str(row1[b'cf3:gender']))
print(str(row2[b'cf1:name']) +" "+str(row2[b'cf2:age'])+"
"+str(row2[b'cf3:gender']))
print("-----------------------------------------")

print("Closing the Connection...")
con.close()
```

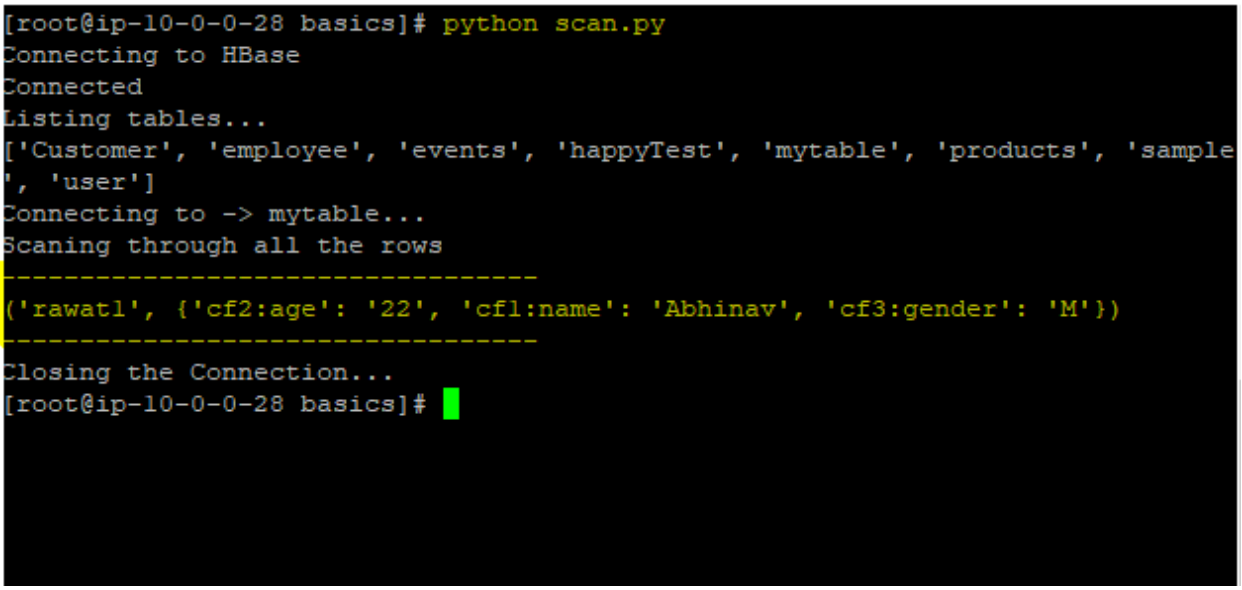
Output:

```
[root@ip-10-0-0-28 basics]# python read_data.py
Connecting to HBase
Connected
Listing tables...
['Customer', 'employee', 'events', 'happyTest', 'mytable', 'products', 'sample
', 'user']
Connecting to -> mytable...
Getting the data...
Printing the data...
-----------------------------------------
Abhinav 22 M
Vishal 27 M
-----------------------------------------
Closing the Connection...
[root@ip-10-0-0-28 basics]#
```

5. Scan rows from tables.

```
#Scan the the rows
import happybase

print("Connecting to HBase")
con = happybase.Connection('localhost')

con.open()
print("Connected")

print("Listing tables...")
print(con.tables())

print("Connecting to -> mytable...")
mtable = con.table('mytable')


print("Scaning through all the rows")

print("----------------------------------")

# The stop and start row points are optional
for key, data in mtable.scan(row_start=b'rawat1', row_stop=b'rawat2'):
    print(key, data)

print("----------------------------------")

print("Closing the Connection...")
con.close()
```

Output:



6. Delete data from tables.

```
#Delete Data

import happybase

print("Connecting to HBase")
con = happybase.Connection('localhost')

con.open()
print("Connected")

print("Listing tables...")
print(con.tables())

print("Connecting to -> mytable...")
mtable = con.table('mytable')
print("Deteting the row-> rawat2...")
mtable.delete(b'rawat2')

print("Deteting a value-> rawat1-cf2:age...")
mtable.delete(b'rawat1', columns=[b'cf2:age'])
```

```
print("Closing the Connection...")
con.close()
```

Output:

```
[root@ip-10-0-0-28 basics]# python delete_data.py
Connecting to HBase
Connected
Listing tables...
['Customer', 'employee', 'events', 'happyTest', 'mytable', 'products', 'sample
', 'user']
Connecting to -> mytable...
Deteting the row-> rawat2...
Deteting a value-> rawat1-cf2:age...
Closing the Connection...
[root@ip-10-0-0-28 basics]#
```

7. Delete HBase tables.

```
# Delete Table
import happybase

print("connecting to HBase")
con = happybase.Connection('localhost')

con.open()
print("Connected")

print("----------------------------------")
print("Listing tables...")
print(con.tables())
print("----------------------------------")
print("Disabling and Deleting the table...")
con.delete_table("mytable",disable=True)

print("Table Deleted")

print("----------------------------------")
```

```
print("Listing remaining tables...")
print(con.tables())
print("-----------------------------------")

print("Closing the Connection...")
con.close()
```

Output:

```
[root@ip-10-0-0-28 basics]# python delete_table.py
connecting to HBase
Connected
-----------------------------------
Listing tables...
['Customer', 'employee', 'events', 'happyTest', 'mytable', 'products', 'sample
', 'user']
-----------------------------------
Disabling and Deleting the table...
Table Deleted
-----------------------------------
Listing remaining tables...
['Customer', 'employee', 'events', 'happyTest', 'products', 'sample', 'user']
-----------------------------------
Closing the Connection...
[root@ip-10-0-0-28 basics]#
```