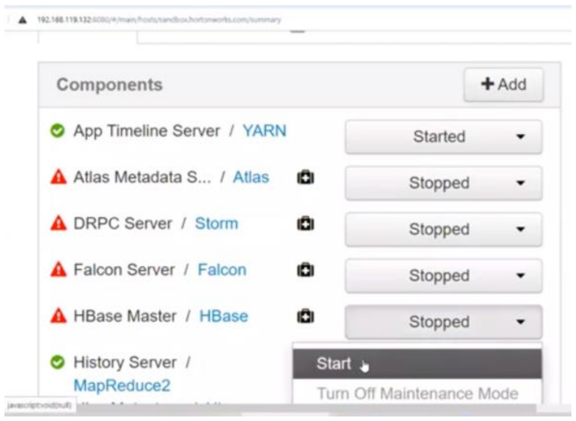
Practical No. 6

Aim:- Implement an application that stores big data in Hbase/ Python

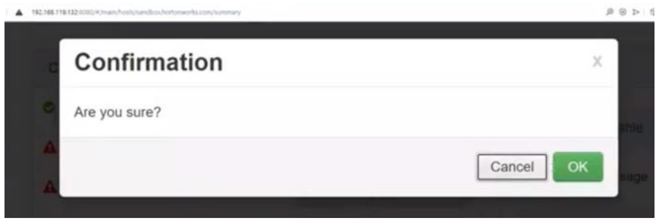
What is HBase?

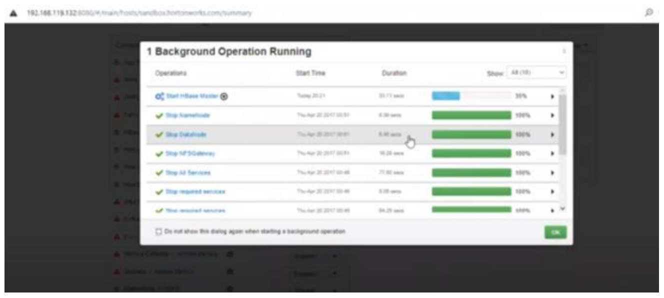
HBase is a distributed column-oriented database built on top of the Hadoop file system. It is an open-source project and is horizontally scalable. It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System.

Go to GUI page and start the hbase service.

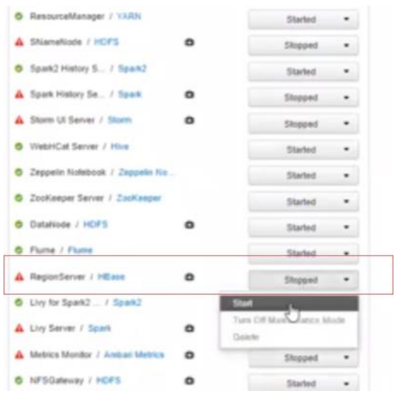


Click on OK to start the service.

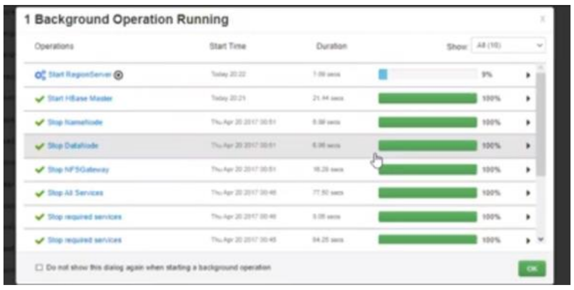




Now we must start region server.



ResourceManager / YARN — Started
SNameNode / HDFS — Stopped
Spark2 History S... / Spark2 — Started
Spark History Se... / Spark — Stopped
Storm UI Server / Storm — Stopped
WebHCat Server / Hive — Started
Zeppelin Notebook / Zeppelin No... — Started
ZooKeeper Server / ZooKeeper — Started
DataNode / HDFS — Started
Flume / Flume — Started
RegionServer / HBase — Stopped
Livy for Spark2 ... / Spark2
Livy Server / Spark
Metrics Monitor / Ambari Metrics — Stopped
NFSGateway / HDFS — Started



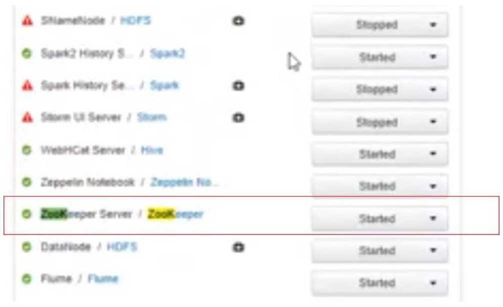
1 Background Operation Running

Show: All (10)

Operations	Start Time	Duration		
Start RegionServer	Today 20:22	1.09 secs		9%
✓ Start HBase Master	Today 20:21	21.44 secs		100%
✓ Stop NameNode	Thu Apr 20 2017 00:51	6.98 secs		100%
✓ Stop DataNode	Thu Apr 20 2017 00:51	6.06 secs		100%
✓ Stop NFSGateway	Thu Apr 20 2017 00:51	16.28 secs		100%
✓ Stop All Services	Thu Apr 20 2017 00:46	77.50 secs		100%
✓ Stop required services	Thu Apr 20 2017 00:46	5.05 secs		100%
✓ Stop required services	Thu Apr 20 2017 00:45	64.25 secs		100%

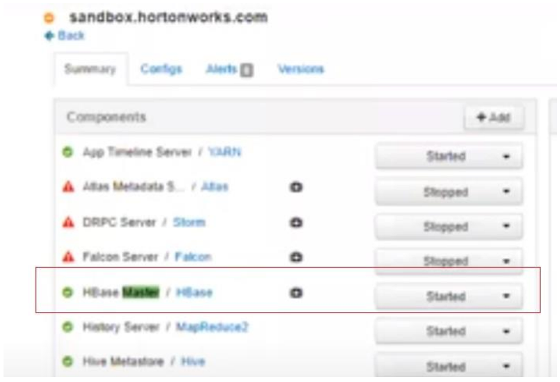
☐ Do not show this dialog again when starting a background operation

Check zooperkeeper server is started.



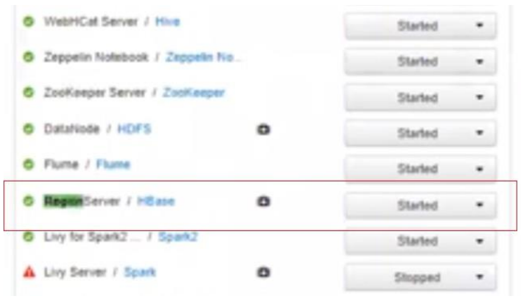
SNameNode / HDFS — Stopped
Spark2 History S... / Spark2 — Started
Spark History Se... / Spark — Stopped
Storm UI Server / Storm — Stopped
WebHCat Server / Hive — Started
Zeppelin Notebook / Zeppelin No... — Started
ZooKeeper Server / ZooKeeper — Started
DataNode / HDFS — Started
Flume / Flume — Started

Check hbase and region server are started.



sandbox.hortonworks.com

App Timeline Server / YARN — Started
Atlas Metadata S... / Atlas — Stopped
DRPC Server / Storm — Stopped
Falcon Server / Falcon — Stopped
HBase Master / HBase — Started
History Server / MapReduce2 — Started
Hive Metastore / Hive — Started

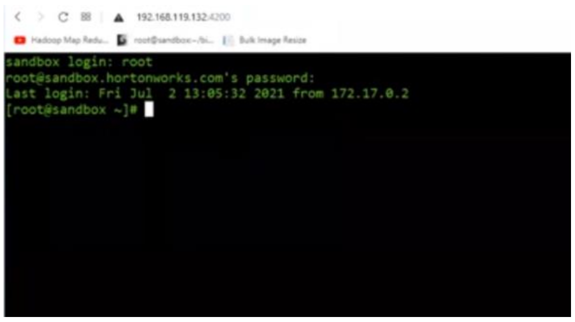


```
WebHCat Server / Hive                 Started
Zeppelin Notebook / Zeppelin No...    Started
ZooKeeper Server / ZooKeeper          Started
DataNode / HDFS                       Started
Flume / Flume                         Started
RegionServer / HBase                  Started
Livy for Spark2 ... / Spark2          Started
Livy Server / Spark                   Stopped
```

Command: **which application-name** gives directory in which application-name is installed.

Open the shell

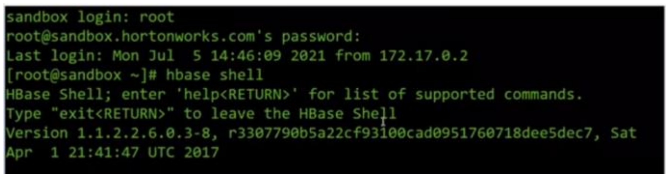
192.168.119.132:4200



```
sandbox login: root
root@sandbox.hortonworks.com's password:
Last login: Fri Jul  2 13:05:32 2021 from 172.17.0.2
[root@sandbox ~]# 
```

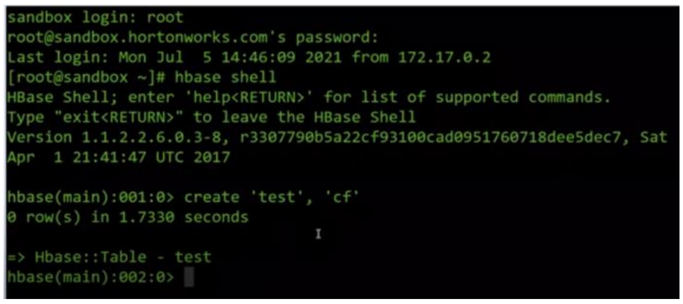
Command: hbase shell

It will start the server



```
sandbox login: root
root@sandbox.hortonworks.com's password:
Last login: Mon Jul  5 14:46:09 2021 from 172.17.0.2
[root@sandbox ~]# hbase shell
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.1.2.2.6.0.3-8, r3307790b5a22cf93100cad0951760718dee5dec7, Sat
Apr  1 21:41:47 UTC 2017
```

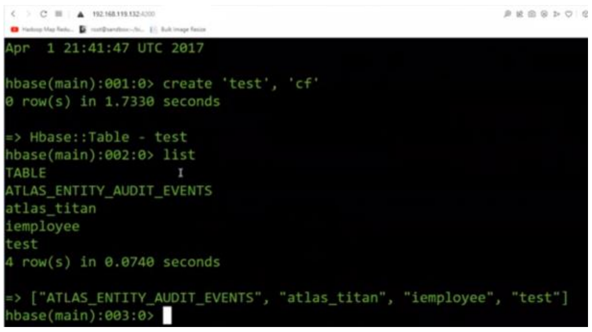
Enter the command create ‘test’, ‘cf’ and it will create the table



```
sandbox login: root
root@sandbox.hortonworks.com's password:
Last login: Mon Jul  5 14:46:09 2021 from 172.17.0.2
[root@sandbox ~]# hbase shell
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.1.2.2.6.0.3-8, r3307790b5a22cf93100cad0951760718dee5dec7, Sat
Apr  1 21:41:47 UTC 2017

hbase(main):001:0> create 'test', 'cf'
0 row(s) in 1.7330 seconds

=> Hbase::Table - test
hbase(main):002:0> 
```

Check the table is created with command

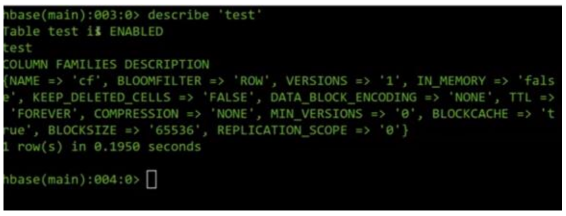
List- It will list all the tables created.



```
Apr  1 21:41:47 UTC 2017

hbase(main):001:0> create 'test', 'cf'
0 row(s) in 1.7330 seconds

=> Hbase::Table - test
hbase(main):002:0> list
TABLE
ATLAS_ENTITY_AUDIT_EVENTS
atlas_titan
iemployee
test
4 row(s) in 0.0740 seconds

=> ["ATLAS_ENTITY_AUDIT_EVENTS", "atlas_titan", "iemployee", "test"]
hbase(main):003:0>
```

If we want to see column description of a table.

Command- describe tablename



```
hbase(main):003:0> describe 'test'
Table test is ENABLED
test
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'fals
e', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 't
rue', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.1950 seconds

hbase(main):004:0>
```

Now, we have to put the values in table

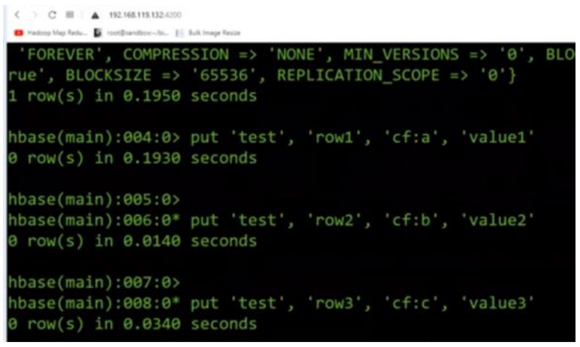
Values:

put ‘test’, ‘row1’, ‘cf:a’, ‘value1’

put ‘test’, ‘row2’, ‘cf:b’, ‘value2’

put ‘test’, ‘row3’, ‘cf:c’, ‘value3’

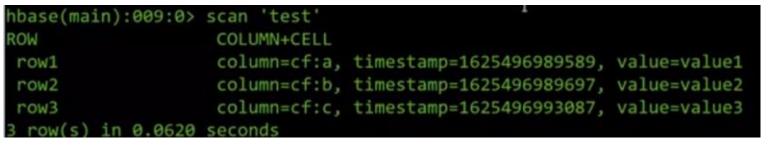
copy paste the data in shell.



```
 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLO
rue', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.1950 seconds

hbase(main):004:0> put 'test', 'row1', 'cf:a', 'value1'
0 row(s) in 0.1930 seconds

hbase(main):005:0>
hbase(main):006:0* put 'test', 'row2', 'cf:b', 'value2'
0 row(s) in 0.0140 seconds

hbase(main):007:0>
hbase(main):008:0* put 'test', 'row3', 'cf:c', 'value3'
0 row(s) in 0.0340 seconds
```

We to display the records of table

Command: scan ‘test’

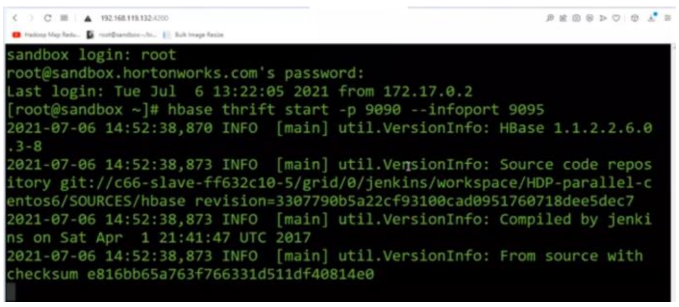


```
hbase(main):009:0> scan 'test'
ROW                  COLUMN+CELL
 row1                column=cf:a, timestamp=1625496989589, value=value1
 row2                column=cf:b, timestamp=1625496989697, value=value2
 row3                column=cf:c, timestamp=1625496993087, value=value3
3 row(s) in 0.0620 seconds
```

Python: storage/retrieval

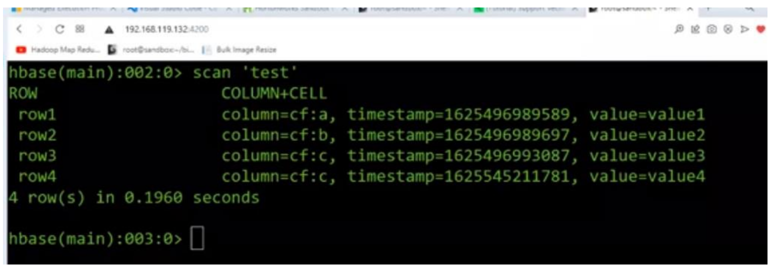
Start the service with command

Hbase thrift start -p 9090 –inforport 9095



```
sandbox login: root
root@sandbox.hortonworks.com's password:
Last login: Tue Jul  6 13:22:05 2021 from 172.17.0.2
[root@sandbox ~]# hbase thrift start -p 9090 --infoport 9095
2021-07-06 14:52:38,870 INFO  [main] util.VersionInfo: HBase 1.1.2.2.6.0
.3-8
2021-07-06 14:52:38,873 INFO  [main] util.VersionInfo: Source code repos
itory git://c66-slave-ff632c10-5/grid/0/jenkins/workspace/HDP-parallel-c
entos6/SOURCES/hbase revision=3307790b5a22cf93100cad0951760718dee5dec7
2021-07-06 14:52:38,873 INFO  [main] util.VersionInfo: Compiled by jenki
ns on Sat Apr  1 21:41:47 UTC 2017
2021-07-06 14:52:38,873 INFO  [main] util.VersionInfo: From source with
checksum e816bb65a763f766331d511df40814e0
```

Create the table the way we did it in hbase and see the records using scan command



```
hbase(main):002:0> scan 'test'
ROW                    COLUMN+CELL
 row1                  column=cf:a, timestamp=1625496989589, value=value1
 row2                  column=cf:b, timestamp=1625496989697, value=value2
 row3                  column=cf:c, timestamp=1625496993087, value=value3
 row4                  column=cf:c, timestamp=1625545211781, value=value4
4 row(s) in 0.1960 seconds

hbase(main):003:0>
```

Create a program file

Import happybase as hb

conn=hb.connection(‘192.168.119.132’, 9090)

print(conn.table(‘test’).row(‘row1’)

print(conn.table(‘test’).row(‘row2’)

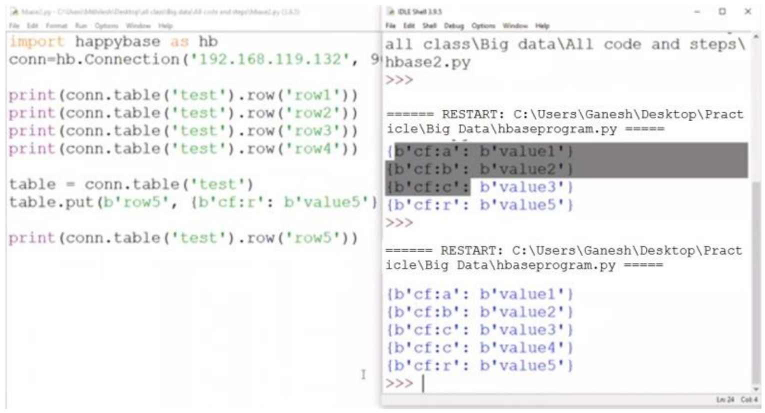
print(conn.table(‘test’).row(‘row3’)

print(conn.table(‘test’).row(‘row4’)

table = conn.table(‘test’)

table.put(b’row5’, {b’cf:r’: b’value5’})

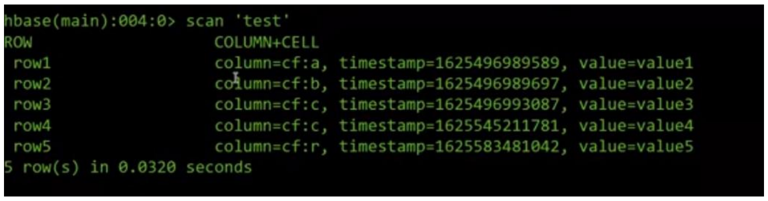
print(conn.table(‘test’).row(‘row5’)



```
import happybase as hb
conn=hb.Connection('192.168.119.132', 9

print(conn.table('test').row('row1'))
print(conn.table('test').row('row2'))
print(conn.table('test').row('row3'))
print(conn.table('test').row('row4'))

table = conn.table('test')
table.put(b'row5', {b'cf:r': b'value5'}

print(conn.table('test').row('row5'))
```

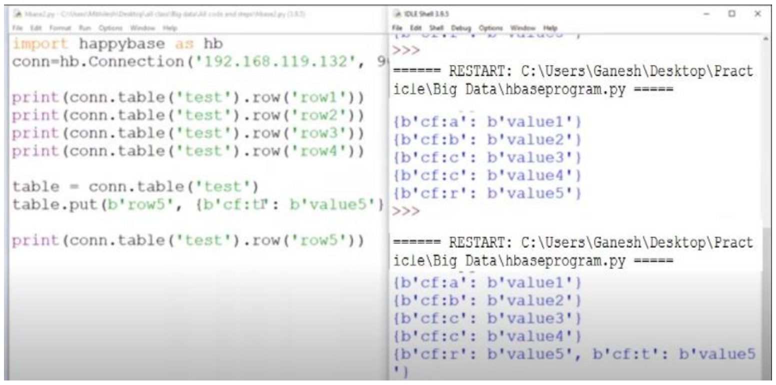
```
all class\Big data\All code and steps\
hbase2.py
>>>

====== RESTART: C:\Users\Ganesh\Desktop\Pract
icle\Big Data\hbaseprogram.py =====
{b'cf:a': b'value1'}
{b'cf:b': b'value2'}
{b'cf:c': b'value3'}
{b'cf:r': b'value5'}
>>>

====== RESTART: C:\Users\Ganesh\Desktop\Pract
icle\Big Data\hbaseprogram.py =====

{b'cf:a': b'value1'}
{b'cf:b': b'value2'}
{b'cf:c': b'value3'}
{b'cf:c': b'value4'}
{b'cf:r': b'value5'}
>>> 
```

Run a scan command on shell to display the values



```
hbase(main):004:0> scan 'test'
ROW                    COLUMN+CELL
 row1                  column=cf:a, timestamp=1625496989589, value=value1
 row2                  column=cf:b, timestamp=1625496989697, value=value2
 row3                  column=cf:c, timestamp=1625496993087, value=value3
 row4                  column=cf:c, timestamp=1625545211781, value=value4
 row5                  column=cf:r, timestamp=1625583481042, value=value5
5 row(s) in 0.0320 seconds
```

Now, try with duplicate value at row 5 say value t

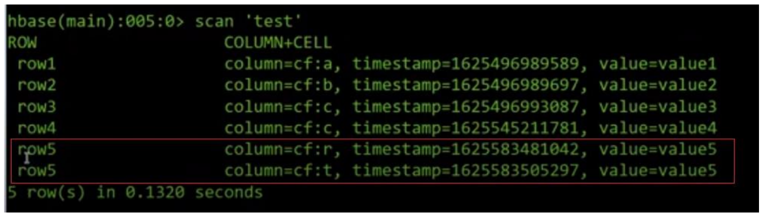


```
import happybase as hb
conn=hb.Connection('192.168.119.132', 9

print(conn.table('test').row('row1'))
print(conn.table('test').row('row2'))
print(conn.table('test').row('row3'))
print(conn.table('test').row('row4'))

table = conn.table('test')
table.put(b'row5', {b'cf:t': b'value5'}

print(conn.table('test').row('row5'))
```

```
>>>
====== RESTART: C:\Users\Ganesh\Desktop\Pract
icle\Big Data\hbaseprogram.py =====

{b'cf:a': b'value1'}
{b'cf:b': b'value2'}
{b'cf:c': b'value3'}
{b'cf:c': b'value4'}
{b'cf:r': b'value5'}
>>>

====== RESTART: C:\Users\Ganesh\Desktop\Pract
icle\Big Data\hbaseprogram.py =====
{b'cf:a': b'value1'}
{b'cf:b': b'value2'}
{b'cf:c': b'value3'}
{b'cf:c': b'value4'}
{b'cf:r': b'value5', b'cf:t': b'value5
'}
```

Run a scan command on shell to display the values

When there is unique value, it will create a record. If duplicate value it will not create arecord



```
hbase(main):005:0> scan 'test'
ROW                    COLUMN+CELL
 row1                  column=cf:a, timestamp=1625496989589, value=value1
 row2                  column=cf:b, timestamp=1625496989697, value=value2
 row3                  column=cf:c, timestamp=1625496993087, value=value3
 row4                  column=cf:c, timestamp=1625545211781, value=value4
 row5                  column=cf:r, timestamp=1625583481042, value=value5
 row5                  column=cf:t, timestamp=1625583505297, value=value5
5 row(s) in 0.1320 seconds
```