NAME: Meith Navlakha
SAP: 60004190068
BDI Practical

EXPERIMENT 5

AIM: To execute HBase commands to load, insert, retrieve, update, or delete data.

THEORY:

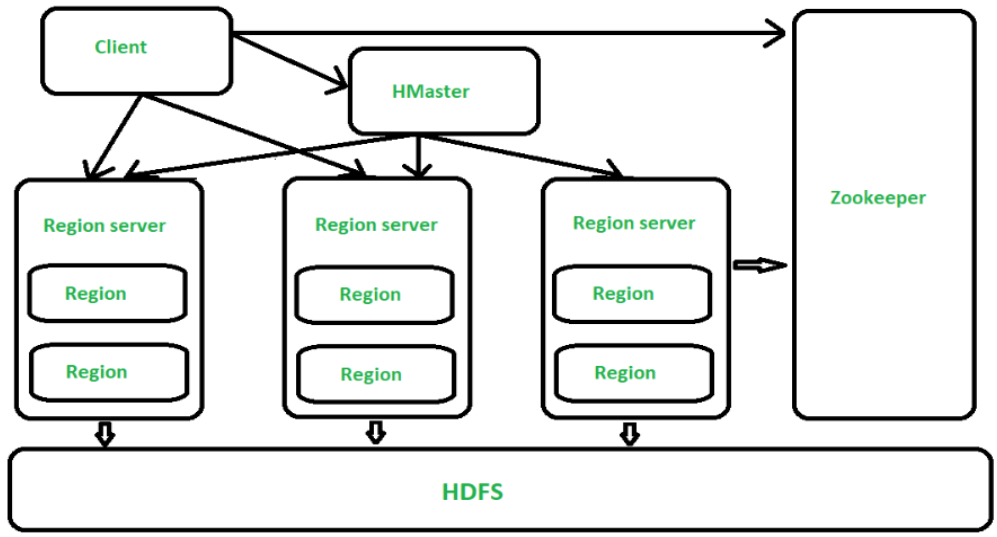
HBase is a data model that is similar to Google's big table. It is an open source, distributed database developed by *Apache* software foundation written in Java. HBase is an essential part of our Hadoop ecosystem. HBase runs on top of HDFS (Hadoop Distributed File System). It can store massive amounts of data from terabytes to petabytes. It is column oriented and horizontally scalable.

Features of HBase:

1. It is linearly scalable across various nodes as well as modularly scalable, as it divided across various nodes.
2. HBase provides consistent read and writes.
3. It provides atomic read and write means during one read or write process, all other processes are prevented from performing any read or write operations.
4. It provides easy to use Java API for client access.
5. It supports Thrift and REST API for non-Java front ends which supports XML, Protobuf and binary data encoding options.
6. It supports a Block Cache and Bloom Filters for real-time queries and for high volume query optimization.
7. HBase provides automatic failure support between Region Servers.
8. It support for exporting metrics with the Hadoop metrics subsystem to files.
9. It doesn't enforce relationship within your data.
10. It is a platform for storing and retrieving data with random access

HBase architecture has 3 main components:

- HMaster
- Region Server
- Zookeeper.



NAME: Meith Navlakha
SAP: 60004190068
BDI Practical

Architecture of HBase :
All the 3 components are described below:

1. **HMaster:**
   The implementation of Master Server in HBase is HMaster. It is a process in which regions are assigned to region server as well as DDL (create, delete table) operations. It monitor all Region Server instances present in the cluster. In a distributed environment, Master runs several background threads. HMaster has many features like controlling load balancing, failover etc.

2. **Region Server:**
   HBase Tables are divided horizontally by row key range into Regions. Regions are the basic building elements of HBase cluster that consists of the distribution of tables and are comprised of Column families. Region Server runs on HDFS DataNode which is present in Hadoop cluster. Regions of Region Server are responsible for several things, like handling, managing, executing as well as reads and writes HBase operations on that set of regions. The default size of a region is 256 MB.

3. **Zookeeper:**
   It is like a coordinator in HBase. It provides services like maintaining configuration information, naming, providing distributed synchronization, server failure notification etc. Clients communicate with region servers via zookeeper.

Advantages of HBase :

1. Can store large data sets
2. Database can be shared
3. Cost-effective from gigabytes to petabytes
4. High availability through failover and replication

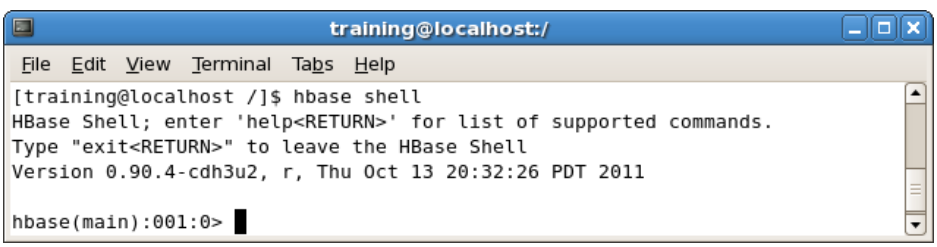
Disadvantages of HBase:

1. No support SQL structure
2. No transaction support
3. Sorted only on key
4. Memory issues on the cluster

IMPLEMENTATION:

1) start

hbase shell

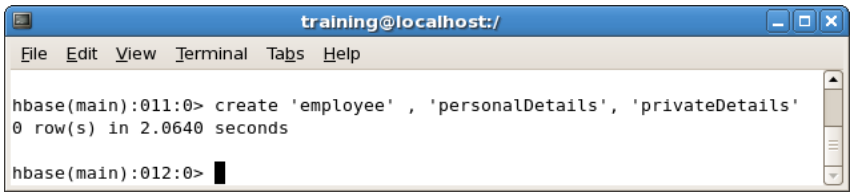


NAME: Meith Navlakha
SAP: 60004190068
BDI Practical

2) Create

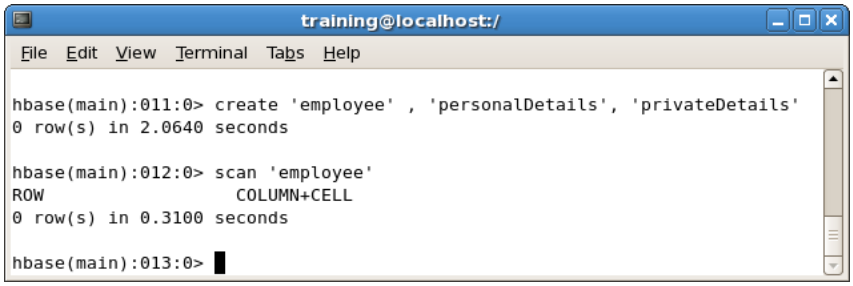
Syntax: create <tablename>, <columnfamilyname>

create 'employee' , 'personalDetails', 'privateDetails'



3) scan

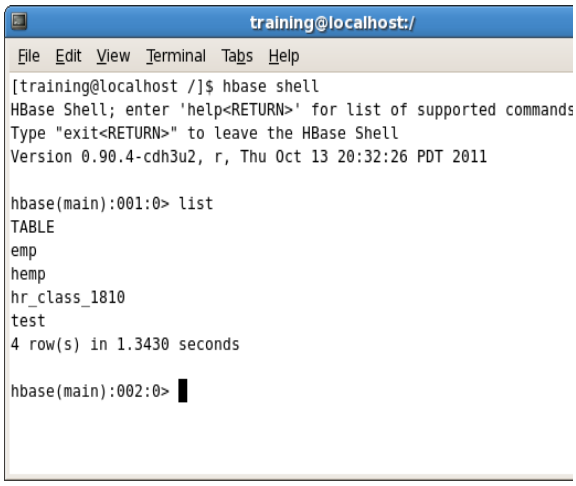
scan 'employee'



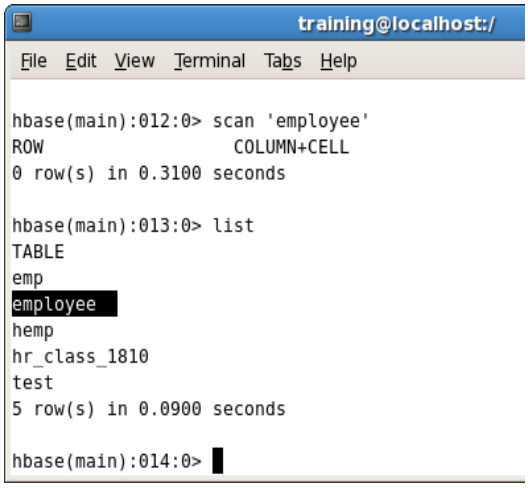
4) check for table

Syntax:list

list



Before

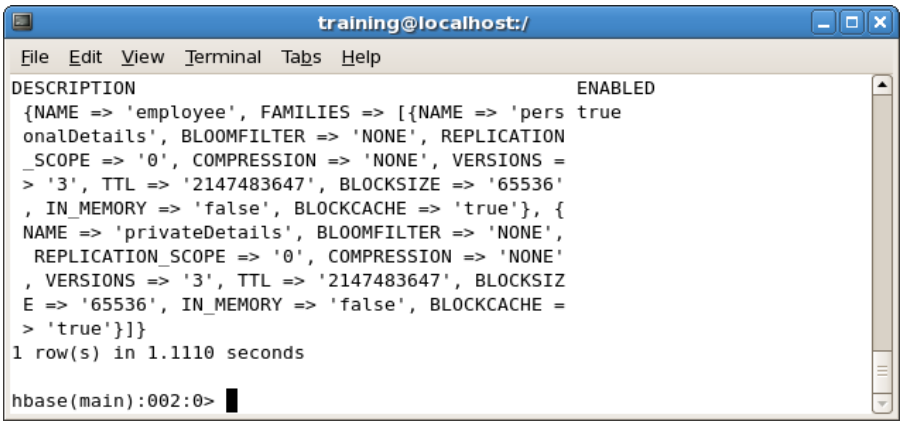


After

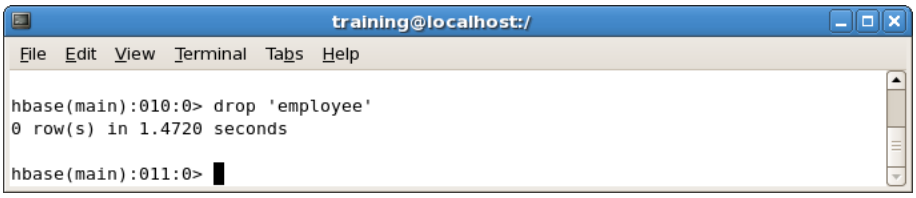
NAME: Meith Navlakha
SAP: 60004190068
BDI Practical

5) Syntax:describe <table name>

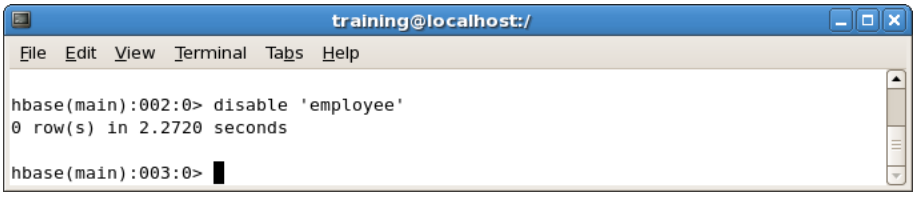
describe 'employee'



6) Syntax:drop <table name>



7) disable 'employee'



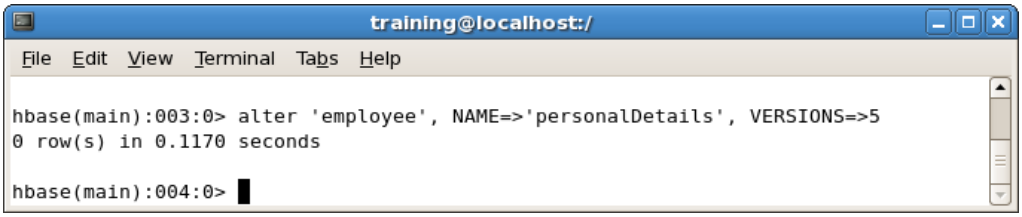
8) Alter

Syntax: alter <tablename>, NAME=><column familyname>, VERSIONS=>5

I] Change the max cell number.

i) single column family

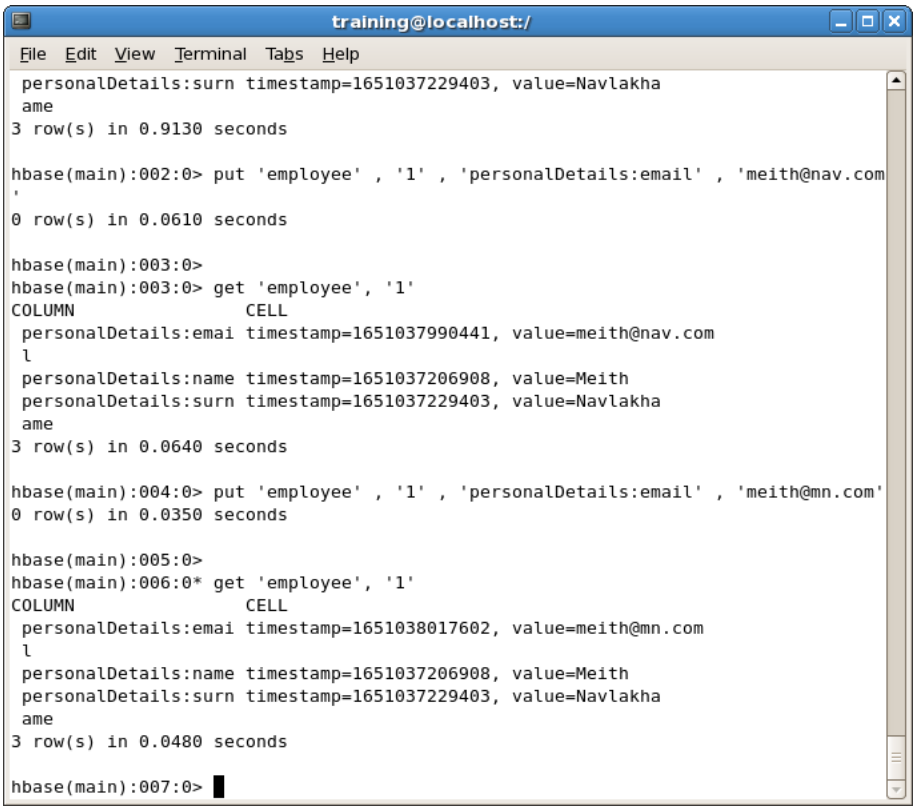
alter 'employee', NAME=>'personalDetails', VERSIONS=>5



NAME: Meith Navlakha
SAP: 60004190068
BDI Practical

II] Update readonly

alter 'employee', READONLY



III] Delete Col Family

alter 'employee', 'delete'=>'personalDetails'

9) Insert

put 'employee' , '1' , 'personalDetails:name' , 'Meith'

put 'employee' , '2' , 'personalDetails:name' , 'Mukesh'

put 'employee' , '3' , 'personalDetails:name' , 'Gautum'

put 'employee' , '1' , 'personalDetails:surname' , 'Navlakha'

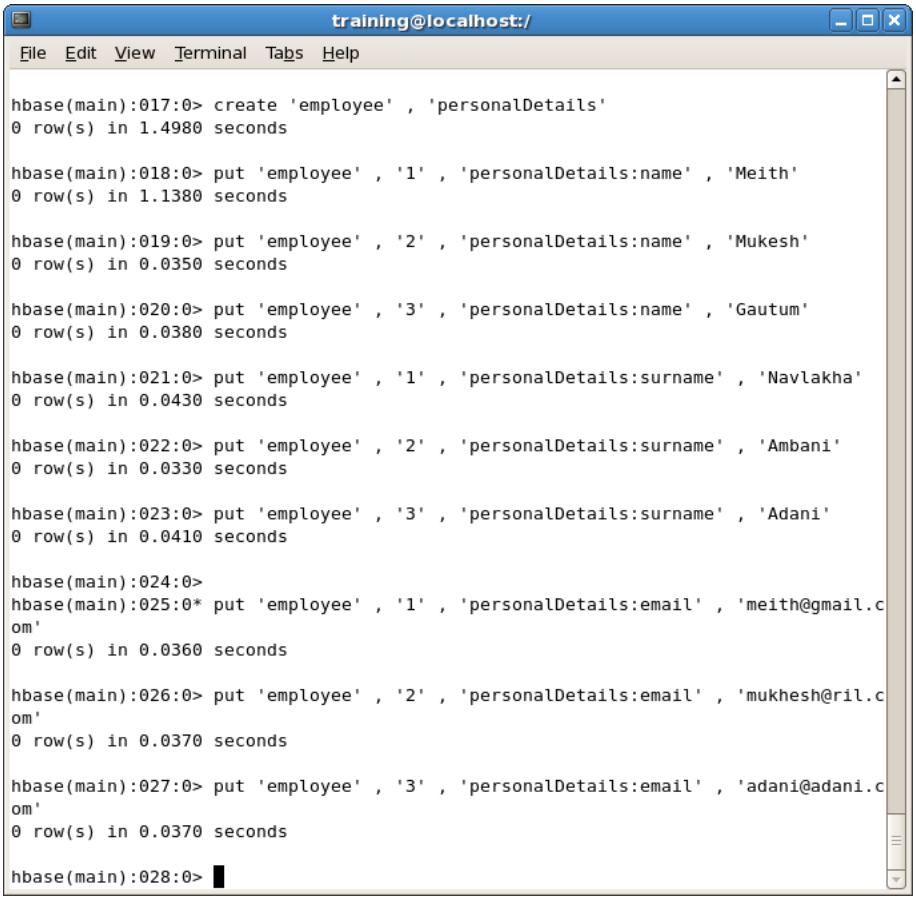
put 'employee' , '2' , 'personalDetails:surname' , 'Ambani'

put 'employee' , '3' , 'personalDetails:surname' , 'Adani'

put 'employee' , '1' , 'personalDetails:email' , 'meith@gmail.com'

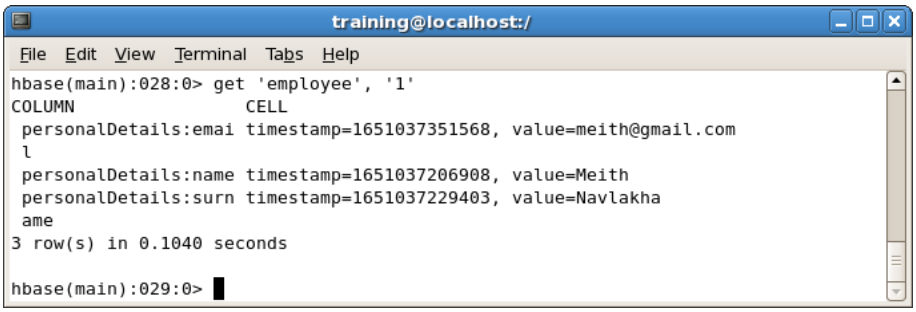
put 'employee' , '2' , 'personalDetails:email' , 'mukhesh@ril.com'

put 'employee' , '3' , 'personalDetails:email' , 'adani@adani.com'

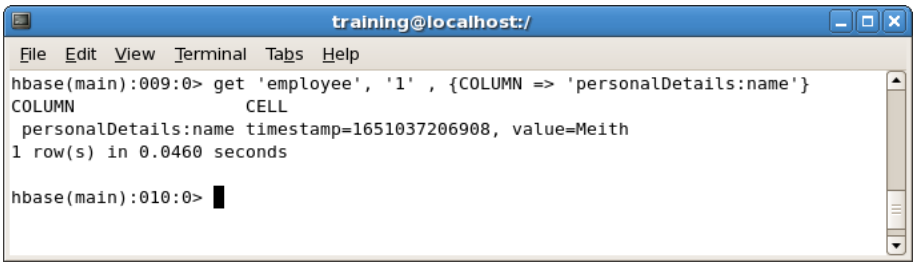


10) get data

get 'employee', '1'

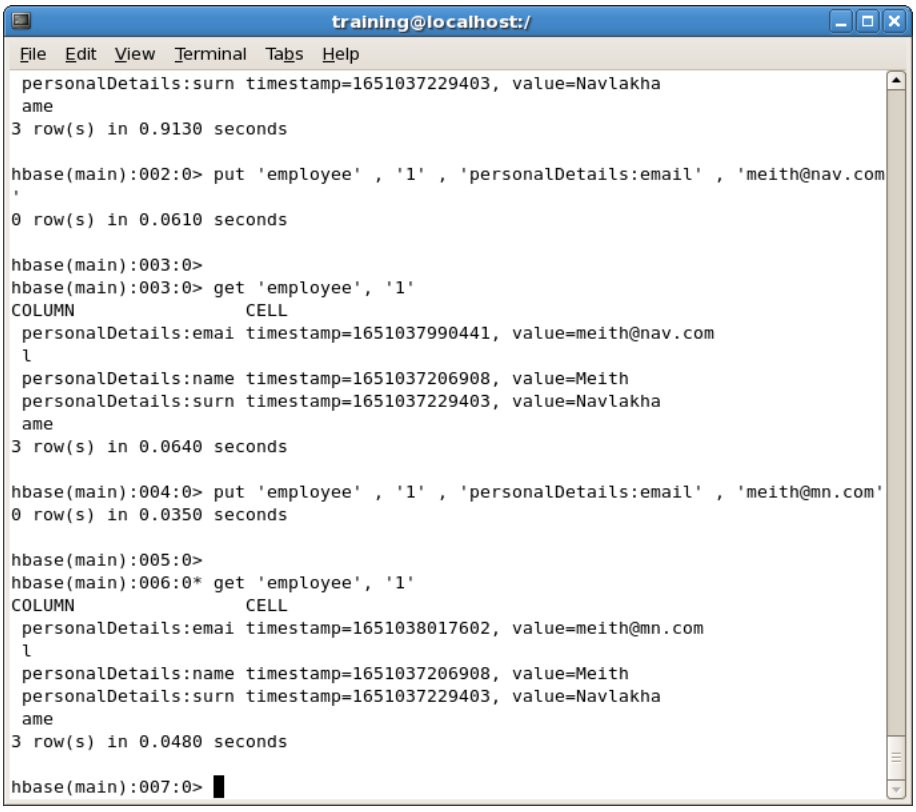


get 'employee', '1' , {COLUMN => 'personalDetails:name'}



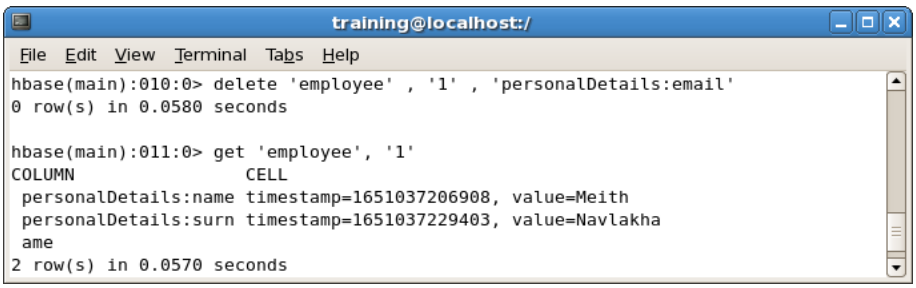
11) update

put 'employee' , '1' , 'personalDetails:email' , 'meith@mn.com'

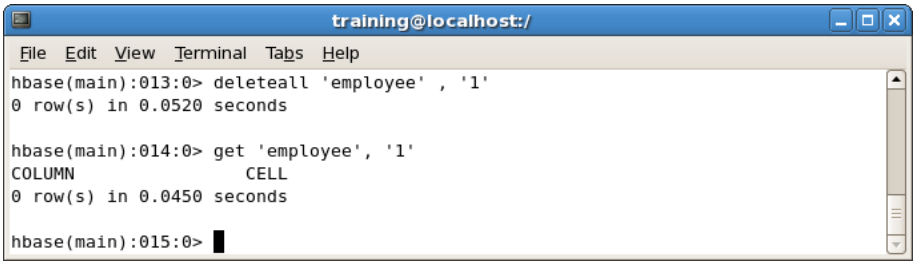


12) Delete

delete 'employee' , '1' , 'personalDetails:email'



deleteall 'employee' , '1'



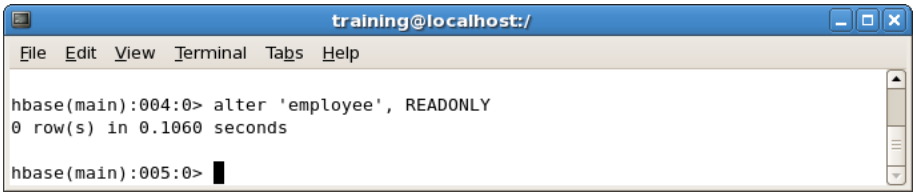
13) Count No. of Rows

count <'tablename'>, CACHE =>1000

count 'employee', CACHE =>1000

14) disable

15) readonly



CONCLUSION:

In this experiment, I explored and learnt about HBase, a database build on top of HDFS for faster execution of data over distributed system. We first learnt about what is HBase and it's application followed by executing it's queries on Cloudera platform. The output is observed and attached. Thus, by the virtue of the experiment, we have successfully understood and implemented the HBase commands.