# Lab: HBase Shell

## Create and Manipulate HBase Tables

## About this Lab

| | |
|---|---|
| **Objective:** | To become familiar with HBase shell operations |
| **File locations:** | n/a |
| **Successful outcome:** | Explored the structure of and performed some command line HBase operations |
| **Before you begin** | Ensure HBase has been started |
| **Related lesson:** | HBase Architecture |

### Switch to the Appropriate User

The cluster being used is the publicly available HDP Sandbox and this lab will be completed as user `maria_dev`. Ensure you are logged into the sandbox as this user. The following steps assume you started a new Terminal.

```
[root@ip-172-30-0-164 ~]# ssh -p 2222 root@127.0.0.1
root@127.0.0.1's password:
Last login: Thu Jun  1 20:58:25 2017 from 172.17.0.1
[root@sandbox ~]# su - maria_dev
[maria_dev@sandbox ~]$ pwd
/home/maria_dev
[maria_dev@sandbox ~]$
```

## Lab Steps

***Perform the following steps:***

1. Launch the HBase Shell
    a. Issue the following command:

```
$ hbase shell
```

   b. A prompt will appear that looks like:

```
hbase>
```

2. Get the version of hbase

   a. Issue the command:

```
hbase> version
```

3. Get the status of hbase

   a. Issue the command:

```
hbase>

status

1 servers, 0 dead, 4.0000 average load
```

   This shows there is only one server in the hbase "cluster.”

   b. Get detailed status by adding 'detailed' to the status command:

```
hbase> status 'detailed'

version 0.96.0.2.0.6.0-76-hadoop2

0 regionsInTransition
master coprocessors: [
]
1 live servers
    sandbox.hortonworks.com:60020 1390046418833
        requestsPerSecond=0.0,
        numberOfOnlineRegions=4,

usedHeapMB=102, maxHeapMB=1004, numberOfStores=4,
numberOfStorefiles=4, storefileUncompressedSizeMB=14,
storefileSizeMB=14, compressionRatio=1.0000, memstoreSizeMB=0,
storefileIndexSizeMB=0, readRequestsCount=1231,
writeRequestsCount=46, rootIndexSizeKB=16,
totalStaticIndexSizeKB=8, totalStaticBloomSizeKB=64,
totalCompactingKVs=37, currentCompactedKVs=37,
compactionProgressPct=1.0, coprocessors=[ ]
```

```
"ambarismoketest,,1382317477828.8f2af39f4875a4539e3876c7b122ad8d."
            numberOfStores=1, numberOfStorefiles=1,

storefileUncompressedSizeMB=0, storefileSizeMB=0,
memstoreSizeMB=0, storefileIndexSizeMB=0, readRequestsCount=0,
writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0,
currentCompactedKVs=0, compactionProgressPct=NaN

        "hbase:meta,,1"

            numberOfStores=1, numberOfStorefiles=1,
storefileUncompressedSizeMB=0, storefileSizeMB=0,
memstoreSizeMB=0, storefileIndexSizeMB=0, readRequestsCount=1223,
writeRequestsCount=14, rootIndexSizeKB=0,
totalStaticIndexSizeKB=0, totalStaticBloomSizeKB=0,
totalCompactingKVs=37, currentCompactedKVs=37,
compactionProgressPct=1.0

"hbase:namespace,,1382307437528.dd774cb1b0266abb03b555f05f0fc334."
            numberOfStores=1, numberOfStorefiles=1,

storefileUncompressedSizeMB=0, storefileSizeMB=0,
memstoreSizeMB=0, storefileIndexSizeMB=0, readRequestsCount=6,
writeRequestsCount=0, rootIndexSizeKB=0, totalStaticIndexSizeKB=0,
totalStaticBloomSizeKB=0, totalCompactingKVs=0,
currentCompactedKVs=0, compactionProgressPct=NaN

        "users,,1390143562217.f241cb20cfd0eef7a212ca1d8128d103."
            numberOfStores=1, numberOfStorefiles=1,

storefileUncompressedSizeMB=14, storefileSizeMB=14,
compressionRatio=1.0000, memstoreSizeMB=0, storefileIndexSizeMB=0,
readRequestsCount=2, writeRequestsCount=32, rootIndexSizeKB=16,
totalStaticIndexSizeKB=8, totalStaticBloomSizeKB=64,
totalCompactingKVs=0, currentCompactedKVs=0,
compactionProgressPct=NaN

0 dead servers
```

c. Hbase is not a relational database. The operations available for data stored in hbase are the following:

   `Get`: retrieves a row or a subset of a row

   `Put`: Add or update a row or a subset of a row

   `Scan`: retrieve a range of sequential rows

   `Delete`: remove a row or a subset of a row

4. Determine what user you are connected as

   a. Use the `whoami` command:

```
hbase> whoami
```

```
maria_dev (auth:SIMPLE)
```

5. Ask HBase for help

   a. Run the `help` command and view the output

   ```
   hbase(main):027:0> help
   HBase Shell, version 0.96.0.2.0.6.0-76-hadoop2,
   re6d7a56f72914d01e55c0478d74e5cfd3778f231, Thu Oct 17 18:15:20 PDT
   2013
   ```

   b. Type `'help "COMMAND"'`, (e.g. `'help "get"'` - the quotes are necessary) for help on a specific command

   Commands are grouped. Type 'help "COMMAND_GROUP"', (e.g. 'help "general"') for help on a command group.

   c. `ddl` stands for Data Definition Language. So help on creating tables defining column family attributes will be displayed by typing:

   ```
   hbase> help 'ddl'


   hbase> help 'get'
   hbase> help 'put'
   hbase> help 'scan'
   hbase> help
   'delete'
   ```

6. Create a table

   a. Use the `create` command

   ```
   hbase> create 't1','cf1'
   0 row(s) in 0.5070 seconds
   ```

   b. List the table you just created using the `list` command

   ```
   hbase> list
   ```

7. Put some data in the table using the `put` command

```
hbase> put 't1', '1','cf1:name','yourname'
0 row(s) in 0.1480 seconds
```

8. Scan the table using the `scan` command

```
hbase> scan 't1'
ROW              COLUMN+CELL
1                column=cf1:name, timestamp=1390166020067, value=yourname
```

9. Add another row

```
hbase> put 't1', '2','cf1:name','rainInSpain'

hbase> scan 't1'
```

10. Change your name

```
hbase> put 't1', '1','cf1:name','your_new_name'
0 row(s) in 0.0070 seconds


hbase > scan ‘t1’
```

11. Drop the table

```
hbase> disable 't1'
0 row(s) in 1.2840 seconds


hbase> drop 't1'
0 row(s) in 0.1630 seconds
```

12. Create a table that stores more than one version of a column

```
hbase> create 't1', {NAME => 'f1', VERSIONS => 2}
```

This creates table t1, with column family f1 and any data stored in column family f1 will be permitted to have up to 2 versions. Versions beyond 2 will be deleted oldest first.

13. Insert multiple versions of a column.

```
hbase> put 't1','1', 'f1:name','name1'


hbase> put 't1','1', 'f1:name','name2'
```

14. Scan the table requesting multiple versions (note different timestamps)

```
hbase> scan 't1',{VERSIONS => 2}

ROW               COLUMN+CELL
 1                column=f1:name, timestamp=1390167231632,
 value=name2
 1                column=f1:name, timestamp=1390167226238,
 value=name1
```

15. Add a third value for the column identifier

```
f1:name hbase> put 't1','1',

'f1:name','name3'
0 row(s) in 0.0080 seconds
```

16. Scan again

```
hbase> scan 't1',{VERSIONS =>
2}
ROW               COLUMN+CELL
 1                column=f1:name, timestamp=1390167445021, value=name3
 1                column=f1:name, timestamp=1390167231632, value=name2
1 row(s) in 0.0110 seconds
```

Try to scan for three versions

```
hbase> scan 't1',{VERSIONS =>
3}

ROW              COLUMN+CELL

 1               column=f1:name, timestamp=1390167445021, value=name3

 1               column=f1:name, timestamp=1390167231632, value=name2

1 row(s) in 0.0170 seconds
```

**Gets vs. Scans:** If the table is large, the scan operation uses a lot of resources. HBase was designed for the optimal lookup to be a single row get.

17. Exit from the HBase shell

```
hbase> exit
```

## Summary

You have now performed some command-line HBase operations.

# Lab: HBase Column Families

## Create and Manipulate HBase Tables

## About this Lab

| | |
|---|---|
| **Objective:** | To create a table with different column families and explore the physical layout of the table in HDFS |
| **File locations:** | n/a |
| **Successful outcome:** | Created an HBase table with multiple column families |
| **Before you begin** | n/a |
| **Related lesson:** | HBase Architecture |

### Switch to the Appropriate User

The cluster being used is the publicly available HDP Sandbox and this lab will be completed as user `maria_dev`. Ensure you are logged into the sandbox as this user. The following steps assume you started a new Terminal.

```
[root@ip-172-30-0-164 ~]# ssh -p 2222 root@127.0.0.1
root@127.0.0.1's password:
Last login: Thu Jun  1 20:58:25 2017 from 172.17.0.1
[root@sandbox ~]# su - maria_dev
[maria_dev@sandbox ~]$ pwd
/home/maria_dev
[maria_dev@sandbox ~]$
```

## Lab Steps

***Perform the following steps:***

1. Launch the HBase shell

   ```
   $ hbase shell
   ```

2. Create a table called cf with two column families, column family a will store a single version of each cell, column family b will store up to 3 versions of each cell

```
hbase> create 'cf',{NAME=>'a', VERSIONS =>1},{NAME=>'b', VERSIONS=>2}
```

3. Insert some cells into each column family

    a. Put a name into column family 'a'

    ```
    hbase> put 'cf','1','a:name','yourname'
    0 row(s) in 0.0200 seconds
    ```

4. Retrieve all cells for rowkey 1

```
 hbase> get 'cf', '1'
 COLUMN                        CELL
a:name                       timestamp=1396035080378, value=yourname
```

5. Put salary into column family 'b'

```
hbase> put 'cf','1','b:salary','50,000'
```

6. Retrieve all cells for rowkey 1

```
hbase> get 'cf', '1'
COLUMN
                             CELL
 a:name                      timestamp=1396035080378, value=yourname
                            CELL
 b:salary                    timestamp=1396035310760, value=50,000
```

7. Enter a new salary for rowkey '1' into column family b

```
hbase> put 'cf','1','b:salary','77,000'
```

8. Retrieve multiple versions of the cells for rowkey 1

```
hbase> get 'cf', '1',{COLUMN =>
['a','b'],VERSIONS=>2}
COLUMN                       CELL
```

```
 a:name                        timestamp=1396035080378, value=yourname
b:salary                      timestamp=1396035310760, value=77,000
b:salary                      timestamp=1396035206632, value=50,000
```

9. Flush the table

    a. The rows just inserted are in a memory cache. Flush them to hdfs:

    ```
    hbase> flush 'cf'
    0 row(s) in 0.1460 seconds
    ```

10. Find the data directories for each column family.

    a. Exit the HBase shell and run this hdfs command:

    ```
    $ hdfs dfs -ls
    /apps/hbase/data/data/default/cf
    Found 3 items
    drwxr-xr-x   - hbase hdfs          0 2014-03-28 12:31
    /apps/hbase/data/data/default/cf/.tabledesc
    drwxr-xr-x   - hbase hdfs          0 2014-03-28 12:31
    /apps/hbase/data/data/default/cf/.tmp
    drwxr-xr-x   - hbase hdfs          0 2014-03-28 12:43
    /apps/hbase/data/data/default/cf/98491258b8d8b1dd5e7d84478a6f3290
    ```

    This shows that the data for table cf is stored in hdfs under `/apps/hbase/data/data/default/cf`

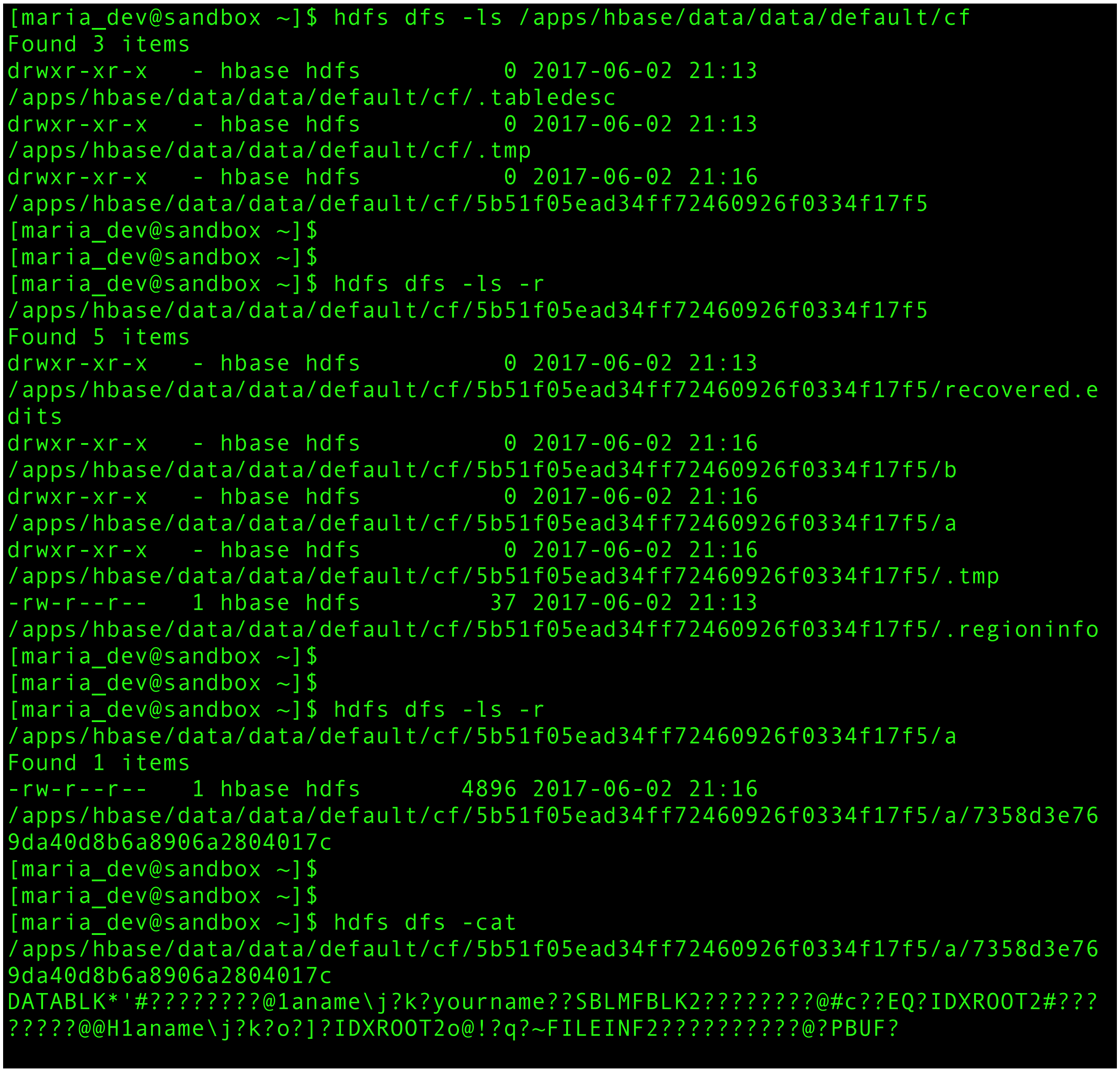
    b. The directory with the hex number for a name is the version number for this table; this number will be different. Use the following command (replacing “HX” with your unique hex number) to see the content:

```
$ hdfs dfs -ls -r /apps/hbase/data/data/default/cf/HX
Found 1 items
-rw-r--r--   3 hbase hdfs        569 2014-03-28 12:31
/apps/hbase/data/data/default/cf/.tabledesc/.tableinfo.0000000001
Found 4 items
-rwxr-xr-x   3 hbase hdfs         35 2014-03-28 12:31
/apps/hbase/data/data/default/cf/98491258b8d8b1dd5e7d84478a6f3290/.regioninfo
drwxr-xr-x   - hbase hdfs          0 2014-03-28 12:43
```

```
/apps/hbase/data/data/default/cf/98491258b8d8b1dd5e7d84478a6f3290/.tmp
drwxr-xr-x   - hbase hdfs          0 2014-03-28 12:43
/apps/hbase/data/data/default/cf/98491258b8d8b1dd5e7d84478a6f3290/a
drwxr-xr-x   - hbase hdfs          0 2014-03-28 12:43
/apps/hbase/data/data/default/cf/98491258b8d8b1dd5e7d84478a6f3290/b
```

There is a directory a for data in column family a, and a directory b for data in column family b.

This specific contents of the underlying HFiles for both column families can be seen by digging a bit deeper. The example below walks down until column family ‘a’ can viewed.



```
[maria_dev@sandbox ~]$ hdfs dfs -ls /apps/hbase/data/data/default/cf
Found 3 items
drwxr-xr-x   - hbase hdfs          0 2017-06-02 21:13
/apps/hbase/data/data/default/cf/.tabledesc
drwxr-xr-x   - hbase hdfs          0 2017-06-02 21:13
/apps/hbase/data/data/default/cf/.tmp
drwxr-xr-x   - hbase hdfs          0 2017-06-02 21:16
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5
[maria_dev@sandbox ~]$
[maria_dev@sandbox ~]$
[maria_dev@sandbox ~]$ hdfs dfs -ls -r
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5
Found 5 items
drwxr-xr-x   - hbase hdfs          0 2017-06-02 21:13
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5/recovered.e
dits
drwxr-xr-x   - hbase hdfs          0 2017-06-02 21:16
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5/b
drwxr-xr-x   - hbase hdfs          0 2017-06-02 21:16
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5/a
drwxr-xr-x   - hbase hdfs          0 2017-06-02 21:16
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5/.tmp
-rw-r--r--   1 hbase hdfs         37 2017-06-02 21:13
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5/.regioninfo
[maria_dev@sandbox ~]$
[maria_dev@sandbox ~]$
[maria_dev@sandbox ~]$ hdfs dfs -ls -r
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5/a
Found 1 items
-rw-r--r--   1 hbase hdfs       4896 2017-06-02 21:16
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5/a/7358d3e76
9da40d8b6a8906a2804017c
[maria_dev@sandbox ~]$
[maria_dev@sandbox ~]$
[maria_dev@sandbox ~]$ hdfs dfs -cat
/apps/hbase/data/data/default/cf/5b51f05ead34ff72460926f0334f17f5/a/7358d3e76
9da40d8b6a8906a2804017c
DATABLK*'#????????@1aname\j?k?yourname??SBLMFBLK2????????@#c??EQ?IDXROOT2#???
?????@@H1aname\j?k?o?]?IDXROOT2o@!?q?~FILEINF2??????????@?PBUF?
```

```
BLOOM_FILTER_TYPEROW

DELETE_FAMILY_COUNT


ARLIEST_PUT_T\j?k?

KEY_VALUE_VERSION

LAST_BLOOM_KEY1

MAJOR_COMPACTION_KEY

MAX_MEMSTORE_TS_KEY

MAX_SEQ_ID_KEY

-
      TIMERANGE \j?k?\j?k?

hfile.AVG_KEY_LEN

file.AVG_VALUE_LEN

hfile.CREATE_TIME_T\j??{
#
hfile.LASTKEY1aname\j?k?Nc
                          BLMFMET2nj????????@?3org.apache.hadoop.hbase.KeyVal
ue$RawBytesComparatorH'1?;?.TRABLK"$?o ?%(08@HPZ.org.apache.hadoop.hbase.KeyV
alue$KeyComparator`[maria_dev@sandbox ~]$
```

## Summary

You have now created an HBase table with multiple column families.