```
Last login: Wed Jun 13 12:03:14 on console
Snehas-MacBook-Pro:~ snehamishra$ start-yarn.sh
-bash: start-yarn.sh: command not found
Snehas-MacBook-Pro:~ snehamishra$ ls
Applications     Movies          lab4
Desktop          Music           myApp
Documents        Pictures    node_modules
Downloads        Public          package-lock.json
Library          eclipse         todo
Snehas-MacBook-Pro:~ snehamishra$ cd Documents/
Snehas-MacBook-Pro:Documents snehamishra$ ls
Increment 1 -  Smart Shopping.key iTunes Software License.rtf
bigdata                  workspace
Snehas-MacBook-Pro:Documents snehamishra$ cd bigdata/
Snehas-MacBook-Pro:bigdata snehamishra$ ls
HadoopInstallationSteps.pages hadoop-2.8.1.tar
data             name
hadoop-2.8.1
Snehas-MacBook-Pro:bigdata snehamishra$ cd hadoop-2.8.1
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ ls
LICENSE.txt  bin      lib     sbin
NOTICE.txt   etc      libexec     share
README.txt   include      logs
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ start-yarn.sh
-bash: start-yarn.sh: command not found
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /Users/snehamishra/Documents/
bigdata/hadoop-2.8.1/logs/yarn-snehamishra-resourcemanager-Snehas-
MacBook-Pro.local.out
localhost: starting nodemanager, logging to /Users/snehamishra/
Documents/bigdata/hadoop-2.8.1/logs/yarn-snehamishra-nodemanager-
Snehas-MacBook-Pro.local.out
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ jps
801 NodeManager
838 Jps
716 ResourceManager
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ sbin/startall-yarn.sh
-bash: sbin/startall-yarn.sh: No such file or directory
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ sbin/dfs.sh
-bash: sbin/dfs.sh: No such file or directory
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ sbin/start-dfs.sh
18/06/13 12:14:16 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /Users/snehamishra/Documents/
bigdata/hadoop-2.8.1/logs/hadoop-snehamishra-namenode-Snehas-MacBook-
Pro.local.out
```

```
localhost: starting datanode, logging to /Users/snehamishra/Documents/
bigdata/hadoop-2.8.1/logs/hadoop-snehamishra-datanode-Snehas-MacBook-
Pro.local.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /Users/snehamishra/
Documents/bigdata/hadoop-2.8.1/logs/hadoop-snehamishra-
secondarynamenode-Snehas-MacBook-Pro.local.out
18/06/13 12:14:31 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ jps
928 NameNode
801 NodeManager
1106 SecondaryNameNode
1174 Jps
716 ResourceManager
1006 DataNode
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ sbin/stop-dfs.sh
18/06/13 12:27:34 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
Stopping namenodes on [localhost]
localhost: stopping namenode
localhost: stopping datanode
Stopping secondary namenodes [0.0.0.0]
0.0.0.0: stopping secondarynamenode
18/06/13 12:27:52 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ sbin/stop-yarn.sh
stopping yarn daemons
stopping resourcemanager
localhost: stopping nodemanager
localhost: nodemanager did not stop gracefully after 5 seconds:
killing with kill -9
no proxyserver to stop
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ jps
1591 Jps
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ brew install hbase
Updating Homebrew...
^Z
[1]+  Stopped                 brew install hbase
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ cd ..
Snehas-MacBook-Pro:bigdata snehamishra$ ls
HadoopInstallationSteps.pages hadoop-2.8.1.tar
data             name
```

```
hadoop-2.8.1
Snehas-MacBook-Pro:bigdata snehamishra$ brew install hbase
Error: Another active Homebrew update process is already in progress.
Please wait for it to finish or terminate it to continue.
==> Installing dependencies for hbase: lzo
==> Installing hbase dependency: lzo
==> Downloading https://homebrew.bintray.com/bottles/
lzo-2.10.high_sierra.bottle
######################################################################
## 100.0%
==> Pouring lzo-2.10.high_sierra.bottle.tar.gz
🍺  /usr/local/Cellar/lzo/2.10: 31 files, 556.5KB
==> Installing hbase
==> Downloading https://homebrew.bintray.com/bottles/
hbase-1.2.6_2.high_sierra.b
######################################################################
## 100.0%
==> Pouring hbase-1.2.6_2.high_sierra.bottle.tar.gz
==> Caveats
To have launchd start hbase now and restart at login:
  brew services start hbase
Or, if you don't want/need a background service you can just run:
  /usr/local/opt/hbase/bin/start-hbase.sh
==> Summary
🍺  /usr/local/Cellar/hbase/1.2.6_2: 9,846 files, 329.8MB
Snehas-MacBook-Pro:bigdata snehamishra$ ls
HadoopInstallationSteps.pages hadoop-2.8.1.tar
data             name
hadoop-2.8.1
Snehas-MacBook-Pro:bigdata snehamishra$ cd hadoop-2.8.1
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ ls
LICENSE.txt  bin      lib      sbin
NOTICE.txt   etc      libexec      share
README.txt   include      logs
Snehas-MacBook-Pro:hadoop-2.8.1 snehamishra$ cd /usr/local/Cellar/
Snehas-MacBook-Pro:Cellar snehamishra$ ls
apr      gettext      lz4      python       utf8proc
apr-util hadoop       lzo      python@2 wget
cassandra    hbase        mongodb      readline xz
cython       libidn2      openssl      sqlite
gdbm     libunistring perl     subversion
Snehas-MacBook-Pro:Cellar snehamishra$ cd hbase/
Snehas-MacBook-Pro:hbase snehamishra$ ls
1.2.6_2
Snehas-MacBook-Pro:hbase snehamishra$ cd 1.2.6_2/
Snehas-MacBook-Pro:1.2.6_2 snehamishra$ ls
CHANGES.txt          README.txt
INSTALL_RECEIPT.json     bin
LICENSE.txt          homebrew.mxcl.hbase.plist
```

```
NOTICE.txt           libexec
Snehas-MacBook-Pro:1.2.6_2 snehamishra$ cd libexec/
Snehas-MacBook-Pro:libexec snehamishra$ ls
bin     conf    docs    hbase-webapps    lib
Snehas-MacBook-Pro:libexec snehamishra$ cd conf/
Snehas-MacBook-Pro:conf snehamishra$ ls
hadoop-metrics2-hbase.properties  hbase-site.xml
hbase-env.sh             log4j.properties
hbase-policy.xml         regionservers
Snehas-MacBook-Pro:conf snehamishra$ vim hbase-site.xml
Snehas-MacBook-Pro:conf snehamishra$ ls
hadoop-metrics2-hbase.properties  hbase-site.xml
hbase-env.sh             log4j.properties
hbase-policy.xml         regionservers
Snehas-MacBook-Pro:conf snehamishra$ vim hbase-env.sh
Snehas-MacBook-Pro:conf snehamishra$ vim hbase-env.sh
Snehas-MacBook-Pro:conf snehamishra$ vim hbase-site.xml
Snehas-MacBook-Pro:conf snehamishra$ ls
hadoop-metrics2-hbase.properties  hbase-site.xml
hbase-env.sh             log4j.properties
hbase-policy.xml         regionservers
Snehas-MacBook-Pro:conf snehamishra$ cd ..
Snehas-MacBook-Pro:libexec snehamishra$ ls
bin     conf    docs    hbase-webapps    lib
Snehas-MacBook-Pro:libexec snehamishra$ bin/start-hbase.sh
localhost: starting zookeeper, logging to /usr/local/var/log/hbase/
hbase-snehamishra-zookeeper-Snehas-MacBook-Pro.local.out
starting master, logging to /usr/local/var/log/hbase/hbase-
snehamishra-master-Snehas-MacBook-Pro.local.out
starting regionserver, logging to /usr/local/var/log/hbase/hbase-
snehamishra-1-regionserver-Snehas-MacBook-Pro.local.out
Snehas-MacBook-Pro:libexec snehamishra$ jps
2629 NameNode
2709 DataNode
3637 Jps
3366 HMaster
3318 HQuorumPeer
2919 ResourceManager
2808 SecondaryNameNode
3003 NodeManager
3518 HRegionServer
Snehas-MacBook-Pro:libexec snehamishra$ bin/hbase shell
2018-06-13 17:00:59,931 WARN  [main] util.NativeCodeLoader: Unable to
load native-hadoop library for your platform... using builtin-java
classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hbase/1.2.6_2/
libexec/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/
StaticLoggerBinder.class]
```

```
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hadoop/3.1.0/
libexec/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/
impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an
explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

hbase(main):001:0> create 'location_table1','country_cf1','state_cf2'

ERROR: Can't get master address from ZooKeeper; znode data == null

Here is some help for this command:
Creates a table. Pass a table name, and a set of column family
specifications (at least one), and, optionally, table configuration.
Column specification can be a simple string (name), or a dictionary
(dictionaries are described below in main help output), necessarily
including NAME attribute.
Examples:

Create a table with namespace=ns1 and table qualifier=t1
  hbase> create 'ns1:t1', {NAME => 'f1', VERSIONS => 5}

Create a table with namespace=default and table qualifier=t1
  hbase> create 't1', {NAME => 'f1'}, {NAME => 'f2'}, {NAME => 'f3'}
  hbase> # The above in shorthand would be the following:
  hbase> create 't1', 'f1', 'f2', 'f3'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 1, TTL => 2592000,
BLOCKCACHE => true}
  hbase> create 't1', {NAME => 'f1', CONFIGURATION =>
{'hbase.hstore.blockingStoreFiles' => '10'}}

Table configuration options can be put at the end.
Examples:

  hbase> create 'ns1:t1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS => ['10', '20', '30', '40']
  hbase> create 't1', 'f1', SPLITS_FILE => 'splits.txt', OWNER =>
'johndoe'
  hbase> create 't1', {NAME => 'f1', VERSIONS => 5}, METADATA =>
{ 'mykey' => 'myvalue' }
  hbase> # Optionally pre-split the table into NUMREGIONS, using
  hbase> # SPLITALGO ("HexStringSplit", "UniformSplit" or classname)
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO =>
'HexStringSplit'}
```

```
  hbase> create 't1', 'f1', {NUMREGIONS => 15, SPLITALGO =>
'HexStringSplit', REGION_REPLICATION => 2, CONFIGURATION =>
{'hbase.hregion.scan.loadColumnFamiliesOnDemand' => 'true'}}
  hbase> create 't1', {NAME => 'f1', DFS_REPLICATION => 1}

You can also keep around a reference to the created table:

  hbase> t1 = create 't1', 'f1'

Which gives you a reference to the table named 't1', on which you can
then
call methods.


hbase(main):002:0> list
TABLE

ERROR: Can't get master address from ZooKeeper; znode data == null

Here is some help for this command:
List all tables in hbase. Optional regular expression parameter could
be used to filter the output. Examples:

  hbase> list
  hbase> list 'abc.*'
  hbase> list 'ns:abc.*'
  hbase> list 'ns:.*'


hbase(main):003:0> Snehas-MacBook-Pro:libexec snehamishra$ exit
logout
There are stopped jobs.
Snehas-MacBook-Pro:libexec snehamishra$ jps
2629 NameNode
2709 DataNode
3941 Jps
3318 HQuorumPeer
2919 ResourceManager
2808 SecondaryNameNode
3003 NodeManager
Snehas-MacBook-Pro:libexec snehamishra$ bin/start-hbase.sh
localhost: zookeeper running as process 3318. Stop it first.
starting master, logging to /usr/local/var/log/hbase/hbase-
snehamishra-master-Snehas-MBP.kc.umkc.edu.out
starting regionserver, logging to /usr/local/var/log/hbase/hbase-
snehamishra-1-regionserver-Snehas-MBP.kc.umkc.edu.out
Snehas-MacBook-Pro:libexec snehamishra$ bin/stop-hbase.sh
stopping hbase...............
localhost: stopping zookeeper.
```

```
Snehas-MacBook-Pro:libexec snehamishra$ jps
2629 NameNode
2709 DataNode
2919 ResourceManager
2808 SecondaryNameNode
3003 NodeManager
4895 Jps
Snehas-MacBook-Pro:libexec snehamishra$ bin/start-hbase.sh
localhost: starting zookeeper, logging to /usr/local/var/log/hbase/
hbase-snehamishra-zookeeper-Snehas-MBP.kc.umkc.edu.out
starting master, logging to /usr/local/var/log/hbase/hbase-
snehamishra-master-Snehas-MBP.kc.umkc.edu.out
starting regionserver, logging to /usr/local/var/log/hbase/hbase-
snehamishra-1-regionserver-Snehas-MBP.kc.umkc.edu.out
Snehas-MacBook-Pro:libexec snehamishra$ jps
5491 Jps
2629 NameNode
2709 DataNode
2919 ResourceManager
5160 HQuorumPeer
2808 SecondaryNameNode
5370 HRegionServer
3003 NodeManager
5213 HMaster
Snehas-MacBook-Pro:libexec snehamishra$ bin/hbase shell
2018-06-13 18:10:01,913 WARN  [main] util.NativeCodeLoader: Unable to
load native-hadoop library for your platform... using builtin-java
classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hbase/1.2.6_2/
libexec/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/
StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hadoop/3.1.0/
libexec/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/
impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an
explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

hbase(main):001:0> create 'location_table1','country_cf1','state_cf2'
0 row(s) in 1.5530 seconds

=> Hbase::Table - location_table1
hbase(main):002:0> list
TABLE
location_table1
```

```
1 row(s) in 0.0270 seconds

=> ["location_table1"]
hbase(main):003:0> describe c
callcc                   caller                   case
catalogjanitor_enabled   catalogjanitor_run
catalogjanitor_switch    catch                    cb
chomp                    chomp!
chop                     chop!                    chws
class                    clear_auths
clone                    clone_snapshot           close_region
com                      compact
compact_rs               conf                     context
count                    create
create_namespace         cws                      cwws
hbase(main):003:0> describe location_table1
NameError: undefined local variable or method `location_table1' for
#<Object:0x234c5e41>

hbase(main):004:0> scan location_table1
NameError: undefined local variable or method `location_table1' for
#<Object:0x234c5e41>

hbase(main):005:0> describe 'location_table1
hbase(main):006:0'
hbase(main):007:0' "
hbase(main):008:0' '

ERROR: Illegal character code:10, <
> at 15. User-space table qualifiers can only contain 'alphanumeric
characters': i.e. [a-zA-Z_0-9-.]: location_table1

"

Here is some help for this command:
Describe the named table. For example:
  hbase> describe 't1'
  hbase> describe 'ns1:t1'

Alternatively, you can use the abbreviated 'desc' for the same thing.
  hbase> desc 't1'
  hbase> desc 'ns1:t1'


hbase(main):009:0> describe 'location_table1'
Table location_table1 is ENABLED
location_table1
COLUMN FAMILIES DESCRIPTION
```

```
{NAME => 'country_cf1', BLOOMFILTER => 'ROW', VERSIONS => '1',
IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCO
DING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS
=> '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATI
ON_SCOPE => '0'}
{NAME => 'state_cf2', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODI
NG => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION
_SCOPE => '0'}
2 row(s) in 0.1180 seconds

hbase(main):010:0> scan 'location_table1'
ROW                               COLUMN+CELL
0 row(s) in 0.0550 seconds

hbase(main):011:0> put
'location_table1','row1','country_cf1:country_code','india'
0 row(s) in 0.0830 seconds

hbase(main):012:0> scan 'location_table1'
ROW                               COLUMN+CELL
 row1                             column=country_cf1:country_code,
timestamp=1528931742562, value=india
1 row(s) in 0.0230 seconds

hbase(main):013:0> describe 'location_table1'
Table location_table1 is ENABLED
location_table1
COLUMN FAMILIES DESCRIPTION
{NAME => 'country_cf1', BLOOMFILTER => 'ROW', VERSIONS => '1',
IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCO
DING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS
=> '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATI
ON_SCOPE => '0'}
{NAME => 'state_cf2', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODI
NG => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION
_SCOPE => '0'}
2 row(s) in 0.0240 seconds

hbase(main):014:0> alter 'location_table1','city_cf3'
Updating all regions with the new schema...
0/1 regions updated.
1/1 regions updated.
Done.
0 row(s) in 3.2010 seconds
```

```
hbase(main):015:0> describe 'location_table1'
Table location_table1 is ENABLED
location_table1
COLUMN FAMILIES DESCRIPTION
{NAME => 'city_cf3', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODIN
G => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_
SCOPE => '0'}
{NAME => 'country_cf1', BLOOMFILTER => 'ROW', VERSIONS => '1',
IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCO
DING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS
=> '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATI
ON_SCOPE => '0'}
{NAME => 'state_cf2', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODI
NG => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION
_SCOPE => '0'}
3 row(s) in 0.0170 seconds

hbase(main):016:0> put 'location_table1','row1','country','india'

ERROR: Unknown column family! Valid column names: city_cf3:*,
country_cf1:*, state_cf2:*

Here is some help for this command:
Put a cell 'value' at specified table/row/column and optionally
timestamp coordinates.  To put a cell value into table 'ns1:t1' or
't1'
at row 'r1' under column 'c1' marked with the time 'ts1', do:

  hbase> put 'ns1:t1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value', ts1
  hbase> put 't1', 'r1', 'c1', 'value',
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1,
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {VISIBILITY=>'PRIVATE|
SECRET'}

The same commands also can be run on a table reference. Suppose you
had a reference
t to table 't1', the corresponding command would be:

  hbase> t.put 'r1', 'c1', 'value', ts1,
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
```

```
hbase(main):017:0> put 'location_table1','row1','country_cf1','india'
0 row(s) in 0.0090 seconds

hbase(main):018:0> describe 'location_table1'
Table location_table1 is ENABLED
location_table1
COLUMN FAMILIES DESCRIPTION
{NAME => 'city_cf3', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODIN
G => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_
SCOPE => '0'}
{NAME => 'country_cf1', BLOOMFILTER => 'ROW', VERSIONS => '1',
IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCO
DING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS
=> '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATI
ON_SCOPE => '0'}
{NAME => 'state_cf2', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODI
NG => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION
_SCOPE => '0'}
3 row(s) in 0.0220 seconds

hbase(main):019:0> scan 'location_table1'
ROW                               COLUMN+CELL
 row1                             column=country_cf1:,
timestamp=1528933503784, value=india
 row1                             column=country_cf1:country_code,
timestamp=1528931742562, value=india
1 row(s) in 0.0240 seconds

hbase(main):020:0> diable 'location_table1'
NoMethodError: undefined method `diable' for #<Object:0x234c5e41>

hbase(main):021:0> drop 'location_table1'

ERROR: Table location_table1 is enabled. Disable it first.

Here is some help for this command:
Drop the named table. Table must first be disabled:
  hbase> drop 't1'
  hbase> drop 'ns1:t1'


hbase(main):022:0> scan 'location_table1'
ROW                               COLUMN+CELL
```

```
 row1                             column=country_cf1:,
timestamp=1528933503784, value=india
 row1                             column=country_cf1:country_code,
timestamp=1528931742562, value=india
1 row(s) in 0.0190 seconds

hbase(main):023:0> disable 'location_table1'
0 row(s) in 2.2860 seconds

hbase(main):024:0> drop 'location_table1'
0 row(s) in 1.2830 seconds

hbase(main):025:0> scan 'location_table1'
ROW                               COLUMN+CELL

ERROR: Unknown table location_table1!

Here is some help for this command:
Scan a table; pass table name and optionally a dictionary of scanner
specifications.  Scanner specifications may include one or more of:
TIMERANGE, FILTER, LIMIT, STARTROW, STOPROW, ROWPREFIXFILTER,
TIMESTAMP,
MAXLENGTH or COLUMNS, CACHE or RAW, VERSIONS, ALL_METRICS or METRICS

If no columns are specified, all columns will be scanned.
To scan all members of a column family, leave the qualifier empty as
in
'col_family'.

The filter can be specified in two ways:
1. Using a filterString - more information on this is available in the
Filter Language document attached to the HBASE-4176 JIRA
2. Using the entire package name of the filter.

If you wish to see metrics regarding the execution of the scan, the
ALL_METRICS boolean should be set to true. Alternatively, if you would
prefer to see only a subset of the metrics, the METRICS array can be
defined to include the names of only the metrics you care about.

Some examples:

  hbase> scan 'hbase:meta'
  hbase> scan 'hbase:meta', {COLUMNS => 'info:regioninfo'}
  hbase> scan 'ns1:t1', {COLUMNS => ['c1', 'c2'], LIMIT => 10,
STARTROW => 'xyz'}
  hbase> scan 't1', {COLUMNS => ['c1', 'c2'], LIMIT => 10, STARTROW =>
'xyz'}
  hbase> scan 't1', {COLUMNS => 'c1', TIMERANGE => [1303668804,
1303668904]}
```

```
  hbase> scan 't1', {REVERSED => true}
  hbase> scan 't1', {ALL_METRICS => true}
  hbase> scan 't1', {METRICS => ['RPC_RETRIES', 'ROWS_FILTERED']}
  hbase> scan 't1', {ROWPREFIXFILTER => 'row2', FILTER => "
    (QualifierFilter (>=, 'binary:xyz')) AND (TimestampsFilter ( 123,
456))"}
  hbase> scan 't1', {FILTER =>
    org.apache.hadoop.hbase.filter.ColumnPaginationFilter.new(1, 0)}
  hbase> scan 't1', {CONSISTENCY => 'TIMELINE'}
For setting the Operation Attributes
  hbase> scan 't1', { COLUMNS => ['c1', 'c2'], ATTRIBUTES => {'mykey'
=> 'myvalue'}}
  hbase> scan 't1', { COLUMNS => ['c1', 'c2'], AUTHORIZATIONS =>
['PRIVATE','SECRET']}
For experts, there is an additional option -- CACHE_BLOCKS -- which
switches block caching for the scanner on (true) or off (false).  By
default it is enabled.  Examples:

  hbase> scan 't1', {COLUMNS => ['c1', 'c2'], CACHE_BLOCKS => false}

Also for experts, there is an advanced option -- RAW -- which
instructs the
scanner to return all cells (including delete markers and uncollected
deleted
cells). This option cannot be combined with requesting specific
COLUMNS.
Disabled by default.  Example:

  hbase> scan 't1', {RAW => true, VERSIONS => 10}

Besides the default 'toStringBinary' format, 'scan' supports custom
formatting
by column.  A user can define a FORMATTER by adding it to the column
name in
the scan specification.  The FORMATTER can be stipulated:

 1. either as a org.apache.hadoop.hbase.util.Bytes method name (e.g,
toInt, toString)
 2. or as a custom class followed by method name: e.g.
'c(MyFormatterClass).format'.

Example formatting cf:qualifier1 and cf:qualifier2 both as Integers:
  hbase> scan 't1', {COLUMNS => ['cf:qualifier1:toInt',
    'cf:qualifier2:c(org.apache.hadoop.hbase.util.Bytes).toInt'] }

Note that you can specify a FORMATTER by column only (cf:qualifier).
You cannot
specify a FORMATTER for all columns of a column family.
```

```
Scan can also be used directly from a table, by first getting a
reference to a
table, like such:

  hbase> t = get_table 't'
  hbase> t.scan

Note in the above situation, you can still provide all the filtering,
columns,
options, etc as described above.



hbase(main):026:0> create 'location','country','state','city'
0 row(s) in 1.2560 seconds

=> Hbase::Table - location
hbase(main):027:0> describe 'location'
Table location is ENABLED
location
COLUMN FAMILIES DESCRIPTION
{NAME => 'city', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING =>
 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0',
BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOP
E => '0'}
{NAME => 'country', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING
 => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_S
COPE => '0'}
{NAME => 'state', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING =
> 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCO
PE => '0'}
3 row(s) in 0.0270 seconds

hbase(main):028:0> put
'location','row1','country:USA','state:KS','city:Mission'

ERROR: no method 'add' for arguments
(org.jruby.java.proxies.ArrayJavaProxy,org.jruby.java.proxies.ArrayJav
aProxy,org.jruby.RubyString,org.jruby.java.proxies.ArrayJavaProxy) on
Java::OrgApacheHadoopHbaseClient::Put
  available overloads:
    (byte[],java.nio.ByteBuffer,long,java.nio.ByteBuffer)
    (byte[],byte[],long,byte[])
```

```
Here is some help for this command:
Put a cell 'value' at specified table/row/column and optionally
timestamp coordinates.  To put a cell value into table 'ns1:t1' or
't1'
at row 'r1' under column 'c1' marked with the time 'ts1', do:

  hbase> put 'ns1:t1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value', ts1
  hbase> put 't1', 'r1', 'c1', 'value',
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1,
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {VISIBILITY=>'PRIVATE|
SECRET'}

The same commands also can be run on a table reference. Suppose you
had a reference
t to table 't1', the corresponding command would be:

  hbase> t.put 'r1', 'c1', 'value', ts1,
{ATTRIBUTES=>{'mykey'=>'myvalue'}}


hbase(main):029:0> put 'location','row1','country','USA'
0 row(s) in 0.0110 seconds

hbase(main):030:0> put 'location','row1','city','Mission'
0 row(s) in 0.0040 seconds

hbase(main):031:0> put 'location','row1','state','KS'
0 row(s) in 0.0100 seconds

hbase(main):032:0> scan 'location'
ROW                               COLUMN+CELL
 row1                             column=city:,
timestamp=1528934092020, value=Mission
 row1                             column=country:,
timestamp=1528934074089, value=USA
 row1                             column=state:,
timestamp=1528934113061, value=KS
1 row(s) in 0.0180 seconds

hbase(main):033:0> put 'location','row2','state','KS'
0 row(s) in 0.0040 seconds

hbase(main):034:0> put 'location','row2','country','USA'
0 row(s) in 0.0040 seconds
```

```
hbase(main):035:0> put 'location','row2','city','KCity'
0 row(s) in 0.0030 seconds

hbase(main):036:0> scan 'location'
ROW                              COLUMN+CELL
 row1                            column=city:,
timestamp=1528934092020, value=Mission
 row1                            column=country:,
timestamp=1528934074089, value=USA
 row1                            column=state:,
timestamp=1528934113061, value=KS
 row2                            column=city:,
timestamp=1528934231789, value=KCity
 row2                            column=country:,
timestamp=1528934213664, value=USA
 row2                            column=state:,
timestamp=1528934201679, value=KS
2 row(s) in 0.0270 seconds

hbase(main):037:0> put 'location','row3','city','KCity'
0 row(s) in 0.0030 seconds

hbase(main):038:0> put 'location','row3','country','India'
0 row(s) in 0.0030 seconds

hbase(main):039:0> put 'location','row3','state','Jharkhand'
0 row(s) in 0.0050 seconds

hbase(main):040:0> scan 'location'
ROW                              COLUMN+CELL
 row1                            column=city:,
timestamp=1528934092020, value=Mission
 row1                            column=country:,
timestamp=1528934074089, value=USA
 row1                            column=state:,
timestamp=1528934113061, value=KS
 row2                            column=city:,
timestamp=1528934231789, value=KCity
 row2                            column=country:,
timestamp=1528934213664, value=USA
 row2                            column=state:,
timestamp=1528934201679, value=KS
 row3                            column=city:,
timestamp=1528934274373, value=KCity
 row3                            column=country:,
timestamp=1528934289785, value=India
 row3                            column=state:,
timestamp=1528934316419, value=Jharkhand
3 row(s) in 0.0230 seconds
```

```
hbase(main):041:0> create 'student_course','student_cf1','course_cf2'
0 row(s) in 2.2650 seconds

=> Hbase::Table - student_course
hbase(main):042:0> describe 'student_course'
Table student_course is ENABLED
student_course
COLUMN FAMILIES DESCRIPTION
{NAME => 'course_cf2', BLOOMFILTER => 'ROW', VERSIONS => '1',
IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCOD
ING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS
=> '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATIO
N_SCOPE => '0'}
{NAME => 'student_cf1', BLOOMFILTER => 'ROW', VERSIONS => '1',
IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCO
DING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS
=> '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATI
ON_SCOPE => '0'}
2 row(s) in 0.0220 seconds

hbase(main):043:0> put 'student_course','row','student_cf1:ID','1'
0 row(s) in 0.0150 seconds

hbase(main):044:0> describe 'student_course'
Table student_course is ENABLED
student_course
COLUMN FAMILIES DESCRIPTION
{NAME => 'course_cf2', BLOOMFILTER => 'ROW', VERSIONS => '1',
IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCOD
ING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS
=> '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATIO
N_SCOPE => '0'}
{NAME => 'student_cf1', BLOOMFILTER => 'ROW', VERSIONS => '1',
IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCO
DING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS
=> '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATI
ON_SCOPE => '0'}
2 row(s) in 0.0200 seconds

hbase(main):045:0> scan 'student_course'
ROW                               COLUMN+CELL
 row                              column=student_cf1:ID,
timestamp=1528934816985, value=1
1 row(s) in 0.0150 seconds

hbase(main):046:0> put
'student_course','row','student_cf1:name','sneha'
0 row(s) in 0.0040 seconds
```

```
hbase(main):047:0> put 'student_course','row','course_cf2:ID','CS590'
0 row(s) in 0.0030 seconds

hbase(main):048:0> put
'student_course','row','course_cf2:Instructor','MAyanka'
0 row(s) in 0.0040 seconds

hbase(main):049:0> put
'student_course','row','course_cf2:University','UMKC'
0 row(s) in 0.0040 seconds

hbase(main):050:0> scan 'student_course'
ROW                               COLUMN+CELL
 row                              column=course_cf2:ID,
timestamp=1528934897050, value=CS590
 row                              column=course_cf2:Instructor,
timestamp=1528934915581, value=MAyanka
 row                              column=course_cf2:University,
timestamp=1528934930512, value=UMKC
 row                              column=student_cf1:ID,
timestamp=1528934816985, value=1
 row                              column=student_cf1:name,
timestamp=1528934866397, value=sneha
1 row(s) in 0.0160 seconds

hbase(main):051:0> put
'student_course','row1','course_cf2:University','UCM'
0 row(s) in 0.0050 seconds

hbase(main):052:0> put
'student_course','row1','course_cf2:Instructor','Jason'
0 row(s) in 0.0040 seconds

hbase(main):053:0> put 'student_course','row1','course_cf2:ID','CS490'
0 row(s) in 0.0040 seconds

hbase(main):054:0> put
'student_course','row1','student_cf1:name','smita'
0 row(s) in 0.0050 seconds

hbase(main):055:0> put 'student_course','row1','student_cf1:ID','2'
0 row(s) in 0.0030 seconds

hbase(main):056:0> scan 'student_course'
ROW                               COLUMN+CELL
 row                              column=course_cf2:ID,
timestamp=1528934897050, value=CS590
```

```
 row                              column=course_cf2:Instructor,
timestamp=1528934915581, value=MAyanka
 row                              column=course_cf2:University,
timestamp=1528934930512, value=UMKC
 row                              column=student_cf1:ID,
timestamp=1528934816985, value=1
 row                              column=student_cf1:name,
timestamp=1528934866397, value=sneha
 row1                             column=course_cf2:ID,
timestamp=1528934998214, value=CS490
 row1                             column=course_cf2:Instructor,
timestamp=1528934983926, value=Jason
 row1                             column=course_cf2:University,
timestamp=1528934970696, value=UCM
 row1                             column=student_cf1:ID,
timestamp=1528935039557, value=2
 row1                             column=student_cf1:name,
timestamp=1528935017471, value=smita
2 row(s) in 0.0180 seconds

hbase(main):057:0> whoami
snehamishra (auth:SIMPLE)
    groups: staff, everyone, localaccounts, _appserverusr, admin,
_appserveradm, _lpadmin, _appstore, _lpoperator, _developer,
_analyticsusers, com.apple.access_ftp, com.apple.access_screensharing,
com.apple.access_ssh, 1

hbase(main):058:0> version
1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

hbase(main):059:0> status 'summary'

ERROR: Can't get master address from ZooKeeper; znode data == null

Here is some help for this command:
Show cluster status. Can be 'summary', 'simple', 'detailed', or
'replication'. The
default is 'summary'. Examples:

  hbase> status
  hbase> status 'simple'
  hbase> status 'summary'
  hbase> status 'detailed'
  hbase> status 'replication'
  hbase> status 'replication', 'source'
  hbase> status 'replication', 'sink'


hbase(main):060:0> status
```

```
ERROR: Can't get master address from ZooKeeper; znode data == null

Here is some help for this command:
Show cluster status. Can be 'summary', 'simple', 'detailed', or
'replication'. The
default is 'summary'. Examples:

  hbase> status
  hbase> status 'simple'
  hbase> status 'summary'
  hbase> status 'detailed'
  hbase> status 'replication'
  hbase> status 'replication', 'source'
  hbase> status 'replication', 'sink'


hbase(main):061:0> alter 'location','row3','city','ranchi'

ERROR: Connection refused

Here is some help for this command:
Alter a table. If the "hbase.online.schema.update.enable" property is
set to
false, then the table must be disabled (see help 'disable'). If the
"hbase.online.schema.update.enable" property is set to true, tables
can be
altered without disabling them first. Altering enabled tables has
caused problems
in the past, so use caution and test it before using in production.

You can use the alter command to add,
modify or delete column families or change table configuration
options.
Column families work in a similar way as the 'create' command. The
column family
specification can either be a name string, or a dictionary with the
NAME attribute.
Dictionaries are described in the output of the 'help' command, with
no arguments.

For example, to change or add the 'f1' column family in table 't1'
from
current value to keep a maximum of 5 cell VERSIONS, do:

  hbase> alter 't1', NAME => 'f1', VERSIONS => 5

You can operate on several column families:
  hbase> alter 't1', 'f1', {NAME => 'f2', IN_MEMORY => true}, {NAME =>
'f3', VERSIONS => 5}

To delete the 'f1' column family in table 'ns1:t1', use one of:

  hbase> alter 'ns1:t1', NAME => 'f1', METHOD => 'delete'
  hbase> alter 'ns1:t1', 'delete' => 'f1'

You can also change table-scope attributes like MAX_FILESIZE,
READONLY,
MEMSTORE_FLUSHSIZE, DURABILITY, etc. These can be put at the end;
for example, to change the max size of a region to 128MB, do:

  hbase> alter 't1', MAX_FILESIZE => '134217728'

You can add a table coprocessor by setting a table coprocessor
attribute:

  hbase> alter 't1',
    'coprocessor'=>'hdfs:///foo.jar|com.foo.FooRegionObserver|1001|
arg1=1,arg2=2'

Since you can have multiple coprocessors configured for a table, a
sequence number will be automatically appended to the attribute name
to uniquely identify it.

The coprocessor attribute must match the pattern below in order for
the framework to understand how to load the coprocessor classes:

  [coprocessor jar file location] | class name | [priority] |
[arguments]

You can also set configuration settings specific to this table or
column family:

  hbase> alter 't1', CONFIGURATION =>
{'hbase.hregion.scan.loadColumnFamiliesOnDemand' => 'true'}
  hbase> alter 't1', {NAME => 'f2', CONFIGURATION =>
{'hbase.hstore.blockingStoreFiles' => '10'}}

You can also remove a table-scope attribute:

  hbase> alter 't1', METHOD => 'table_att_unset', NAME =>
'MAX_FILESIZE'

  hbase> alter 't1', METHOD => 'table_att_unset', NAME =>
'coprocessor$1'

You can also set REGION_REPLICATION:
  hbase> alter 't1', {REGION_REPLICATION => 2}

There could be more than one alteration in one command:

  hbase> alter 't1', { NAME => 'f1', VERSIONS => 3 },
   { MAX_FILESIZE => '134217728' }, { METHOD => 'delete', NAME => 'f2'
},
   OWNER => 'johndoe', METADATA => { 'mykey' => 'myvalue' }


hbase(main):062:0> alter 'location','country','state','city','cf4'

ERROR: Connection refused

Here is some help for this command:
Alter a table. If the "hbase.online.schema.update.enable" property is
set to
false, then the table must be disabled (see help 'disable'). If the
"hbase.online.schema.update.enable" property is set to true, tables
can be
altered without disabling them first. Altering enabled tables has
caused problems
in the past, so use caution and test it before using in production.

You can use the alter command to add,
modify or delete column families or change table configuration
options.
Column families work in a similar way as the 'create' command. The
column family
specification can either be a name string, or a dictionary with the
NAME attribute.
Dictionaries are described in the output of the 'help' command, with
no arguments.

For example, to change or add the 'f1' column family in table 't1'
from
current value to keep a maximum of 5 cell VERSIONS, do:

  hbase> alter 't1', NAME => 'f1', VERSIONS => 5

You can operate on several column families:

  hbase> alter 't1', 'f1', {NAME => 'f2', IN_MEMORY => true}, {NAME =>
'f3', VERSIONS => 5}

To delete the 'f1' column family in table 'ns1:t1', use one of:

  hbase> alter 'ns1:t1', NAME => 'f1', METHOD => 'delete'
  hbase> alter 'ns1:t1', 'delete' => 'f1'

You can also change table-scope attributes like MAX_FILESIZE,
READONLY,
MEMSTORE_FLUSHSIZE, DURABILITY, etc. These can be put at the end;
for example, to change the max size of a region to 128MB, do:

  hbase> alter 't1', MAX_FILESIZE => '134217728'

You can add a table coprocessor by setting a table coprocessor
attribute:

  hbase> alter 't1',
    'coprocessor'=>'hdfs:///foo.jar|com.foo.FooRegionObserver|1001|
arg1=1,arg2=2'

Since you can have multiple coprocessors configured for a table, a
sequence number will be automatically appended to the attribute name
to uniquely identify it.

The coprocessor attribute must match the pattern below in order for
the framework to understand how to load the coprocessor classes:

  [coprocessor jar file location] | class name | [priority] |
[arguments]

You can also set configuration settings specific to this table or
column family:

  hbase> alter 't1', CONFIGURATION =>
{'hbase.hregion.scan.loadColumnFamiliesOnDemand' => 'true'}
  hbase> alter 't1', {NAME => 'f2', CONFIGURATION =>
{'hbase.hstore.blockingStoreFiles' => '10'}}

You can also remove a table-scope attribute:

  hbase> alter 't1', METHOD => 'table_att_unset', NAME =>
'MAX_FILESIZE'

  hbase> alter 't1', METHOD => 'table_att_unset', NAME =>
'coprocessor$1'

You can also set REGION_REPLICATION:

  hbase> alter 't1', {REGION_REPLICATION => 2}

There could be more than one alteration in one command:

  hbase> alter 't1', { NAME => 'f1', VERSIONS => 3 },
   { MAX_FILESIZE => '134217728' }, { METHOD => 'delete', NAME => 'f2'
},
   OWNER => 'johndoe', METADATA => { 'mykey' => 'myvalue' }


hbase(main):063:0> list
TABLE

ERROR: Can't get master address from ZooKeeper; znode data == null

Here is some help for this command:
List all tables in hbase. Optional regular expression parameter could
be used to filter the output. Examples:

  hbase> list
  hbase> list 'abc.*'
  hbase> list 'ns:abc.*'
  hbase> list 'ns:.*'


hbase(main):064:0> JPS
NameError: uninitialized constant JPS

hbase(main):065:0> jps
NameError: undefined local variable or method `jps' for #<Object:
0x234c5e41>

hbase(main):066:0> exit
Snehas-MacBook-Pro:libexec snehamishra$ jps
2629 NameNode
2709 DataNode
6679 Jps
2919 ResourceManager
5160 HQuorumPeer
2808 SecondaryNameNode
3003 NodeManager
Snehas-MacBook-Pro:libexec snehamishra$ bin/stop-hbase.sh
stopping hbasecat: /tmp/hbase-snehamishra-master.pid: No such file or
directory

localhost: stopping zookeeper.
Snehas-MacBook-Pro:libexec snehamishra$ jps
2629 NameNode
2709 DataNode
2919 ResourceManager
2808 SecondaryNameNode
7049 Jps
3003 NodeManager
Snehas-MacBook-Pro:libexec snehamishra$ bin/start-hbase.sh
```

```
localhost: starting zookeeper, logging to /usr/local/var/log/hbase/
hbase-snehamishra-zookeeper-Snehas-MBP.kc.umkc.edu.out
starting master, logging to /usr/local/var/log/hbase/hbase-
snehamishra-master-Snehas-MBP.kc.umkc.edu.out
starting regionserver, logging to /usr/local/var/log/hbase/hbase-
snehamishra-1-regionserver-Snehas-MBP.kc.umkc.edu.out
Snehas-MacBook-Pro:libexec snehamishra$ jps
7650 Jps
2629 NameNode
2709 DataNode
2919 ResourceManager
7319 HQuorumPeer
2808 SecondaryNameNode
7531 HRegionServer
3003 NodeManager
7373 HMaster
Snehas-MacBook-Pro:libexec snehamishra$ hbase shell
2018-06-13 19:44:03,279 WARN  [main] util.NativeCodeLoader: Unable to
load native-hadoop library for your platform... using builtin-java
classes where applicable
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hbase/1.2.6_2/
libexec/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/
StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/usr/local/Cellar/hadoop/3.1.0/
libexec/share/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/
impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an
explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

hbase(main):001:0> list
TABLE
location
student_course
2 row(s) in 0.1640 seconds

=> ["location", "student_course"]
hbase(main):002:0> status
1 active master, 0 backup masters, 1 servers, 0 dead, 4.0000 average
load

hbase(main):003:0> balancer
true
0 row(s) in 0.0240 seconds
```

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

ERROR: wrong number of arguments (0 for 1)

Here is some help for this command:
Enable/Disable balancer. Returns previous balancer state.
Examples:

  hbase> balance_switch true
  hbase> balance_switch false


hbase(main):005:0> balance_switch true
true
0 row(s) in 0.0280 seconds

hbase(main):006:0> create 'user_action','user','action'
0 row(s) in 1.3600 seconds

=> Hbase::Table - user_action
hbase(main):007:0> describe 'user_action'
Table user_action is ENABLED
user_action
COLUMN FAMILIES DESCRIPTION
{NAME => 'action', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING
=> 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SC
OPE => '0'}
{NAME => 'user', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING =>
 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0',
BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOP
E => '0'}
2 row(s) in 0.0850 seconds

hbase(main):008:0> put 'user_action', 'row1','user:ID','1'
0 row(s) in 0.0810 seconds

hbase(main):009:0> put 'user_action', 'row1','user:name','Sneha'
0 row(s) in 0.0090 seconds

hbase(main):010:0> put 'user_action', 'row1','action','register'
0 row(s) in 0.0100 seconds

hbase(main):011:0> scan 'user_action'
ROW                               COLUMN+CELL
 row1                             column=action:,
timestamp=1528937412865, value=register
```

```
 row1                             column=user:ID,
timestamp=1528937380374, value=1
 row1                             column=user:name,
timestamp=1528937396551, value=Sneha
1 row(s) in 0.0380 seconds

hbase(main):012:0> put 'user_action', 'row2','action','login'
0 row(s) in 0.0030 seconds

hbase(main):013:0> put 'user_action', 'row2','user:name','Sneha'
0 row(s) in 0.0040 seconds

hbase(main):014:0> put 'user_action', 'row2','user:ID','1'
0 row(s) in 0.0030 seconds

hbase(main):015:0> put 'user_action', 'row3','user:name','Aditya'
0 row(s) in 0.0030 seconds

hbase(main):016:0> put 'user_action', 'row3','user:ID','2'
0 row(s) in 0.0070 seconds

hbase(main):017:0> put 'user_action', 'row3','action','login'
0 row(s) in 0.0070 seconds

hbase(main):018:0> scan 'user_action'
ROW                               COLUMN+CELL
 row1                             column=action:,
timestamp=1528937412865, value=register
 row1                             column=user:ID,
timestamp=1528937380374, value=1
 row1                             column=user:name,
timestamp=1528937396551, value=Sneha
 row2                             column=action:,
timestamp=1528937438186, value=login
 row2                             column=user:ID,
timestamp=1528937478482, value=1
 row2                             column=user:name,
timestamp=1528937469465, value=Sneha
 row3                             column=action:,
timestamp=1528937526284, value=login
 row3                             column=user:ID,
timestamp=1528937515104, value=2
 row3                             column=user:name,
timestamp=1528937503720, value=Aditya
3 row(s) in 0.0280 seconds

hbase(main):019:0> create 'user_friend','user','friend'
0 row(s) in 1.2640 seconds
```

```
=> Hbase::Table - user_friend
hbase(main):020:0> describe 'user_friend'
Table user_friend is ENABLED
user_friend
COLUMN FAMILIES DESCRIPTION
{NAME => 'friend', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING
=> 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS =>
'0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SC
OPE => '0'}
{NAME => 'user', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING =>
 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0',
BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOP
E => '0'}
2 row(s) in 0.0240 seconds

hbase(main):021:0> put 'user_friend','user:ID','1'

ERROR: wrong number of arguments (3 for 4)

Here is some help for this command:
Put a cell 'value' at specified table/row/column and optionally
timestamp coordinates.  To put a cell value into table 'ns1:t1' or
't1'
at row 'r1' under column 'c1' marked with the time 'ts1', do:

  hbase> put 'ns1:t1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value', ts1
  hbase> put 't1', 'r1', 'c1', 'value',
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1,
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {VISIBILITY=>'PRIVATE|
SECRET'}

The same commands also can be run on a table reference. Suppose you
had a reference
t to table 't1', the corresponding command would be:

  hbase> t.put 'r1', 'c1', 'value', ts1,
{ATTRIBUTES=>{'mykey'=>'myvalue'}}


hbase(main):022:0> put 'user_friend','row1','user:ID','1'
0 row(s) in 0.0110 seconds

hbase(main):023:0> put 'user_friend','row1','user:name','Sneha'
```

```
0 row(s) in 0.0050 seconds

hbase(main):024:0> put 'user_friend','row1','friend:name','Aditya'
0 row(s) in 0.0130 seconds

hbase(main):025:0> put 'user_friend','row1','friend:ID','10'
0 row(s) in 0.0030 seconds

hbase(main):026:0> put 'user_friend','row2','friend:name','Swati'
0 row(s) in 0.0060 seconds

hbase(main):027:0> put 'user_friend','row2','friend:ID','12'
0 row(s) in 0.0030 seconds

hbase(main):028:0> scan 'user_friend'
ROW                              COLUMN+CELL
 row1                            column=friend:ID,
timestamp=1528937826330, value=10
 row1                            column=friend:name,
timestamp=1528937816627, value=Aditya
 row1                            column=user:ID,
timestamp=1528937787330, value=1
 row1                            column=user:name,
timestamp=1528937797559, value=Sneha
 row2                            column=friend:ID,
timestamp=1528937865821, value=12
 row2                            column=friend:name,
timestamp=1528937857642, value=Swati
2 row(s) in 0.0270 seconds

hbase(main):029:0> create 'access_log','user','log'
0 row(s) in 1.2520 seconds

=> Hbase::Table - access_log
hbase(main):030:0> describe 'access_log'
Table access_log is ENABLED
access_log
COLUMN FAMILIES DESCRIPTION
{NAME => 'log', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING =>
'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0',
BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE
 => '0'}
{NAME => 'user', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING =>
 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0',
BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOP
E => '0'}
2 row(s) in 0.0290 seconds
```

```
hbase(main):031:0> put 'user_log', 'row1','user:name','Sneha'
2018-06-13 19:59:17,704 ERROR [main] client.AsyncProcess: Failed to
get region location
org.apache.hadoop.hbase.TableNotFoundException: Table 'user_log' was
not found, got: user_friend.
	at
org.apache.hadoop.hbase.client.ConnectionManager$HConnectionImplementa
tion.locateRegionInMeta(ConnectionManager.java:1300)
	at
org.apache.hadoop.hbase.client.ConnectionManager$HConnectionImplementa
tion.locateRegion(ConnectionManager.java:1181)
	at
org.apache.hadoop.hbase.client.AsyncProcess.submit(AsyncProcess.java:
410)
	at
org.apache.hadoop.hbase.client.AsyncProcess.submit(AsyncProcess.java:
359)
	at
org.apache.hadoop.hbase.client.BufferedMutatorImpl.backgroundFlushComm
its(BufferedMutatorImpl.java:238)
	at
org.apache.hadoop.hbase.client.BufferedMutatorImpl.flush(BufferedMutat
orImpl.java:190)
	at org.apache.hadoop.hbase.client.HTable.flushCommits(HTable.java:
1434)
	at org.apache.hadoop.hbase.client.HTable.put(HTable.java:1018)
	at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
	at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.j
ava:62)
	at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccess
orImpl.java:43)
	at java.lang.reflect.Method.invoke(Method.java:498)
	at
org.jruby.javasupport.JavaMethod.invokeDirectWithExceptionHandling(Jav
aMethod.java:450)
	at org.jruby.javasupport.JavaMethod.invokeDirect(JavaMethod.java:
311)
	at
org.jruby.java.invokers.InstanceMethodInvoker.call(InstanceMethodInvok
er.java:59)
	at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
167)
	at org.jruby.ast.CallOneArgNode.interpret(CallOneArgNode.java:57)
	at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
	at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
```

```
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
    at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:120)
    at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:134)
    at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:174)
    at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
69)
    at org.jruby.ast.CallManyArgsNode.interpret(CallManyArgsNode.java:
59)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_BLOCK(ASTInterpreter.java
:111)
    at
org.jruby.runtime.InterpretedBlock.evalBlockBody(InterpretedBlock.java
:374)
    at org.jruby.runtime.InterpretedBlock.yield(InterpretedBlock.java:
295)
    at
org.jruby.runtime.InterpretedBlock.yieldSpecific(InterpretedBlock.java
:229)
    at org.jruby.runtime.Block.yieldSpecific(Block.java:99)
    at org.jruby.ast.ZYieldNode.interpret(ZYieldNode.java:25)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
    at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:169)
    at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:191)
    at
org.jruby.runtime.callsite.CachingCallSite.callBlock(CachingCallSite.j
ava:142)
    at
org.jruby.runtime.callsite.CachingCallSite.callIter(CachingCallSite.ja
va:153)
```

```
    at 
org.jruby.ast.FCallNoArgBlockNode.interpret(FCallNoArgBlockNode.java:
32)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at 
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
    at 
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:120)
    at 
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:134)
    at 
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:174)
    at 
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
69)
    at 
org.jruby.ast.FCallManyArgsNode.interpret(FCallManyArgsNode.java:60)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at 
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
    at 
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:120)
    at 
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:165)
    at org.jruby.RubyClass.finvoke(RubyClass.java:573)
    at org.jruby.RubyBasicObject.send(RubyBasicObject.java:2801)
    at org.jruby.RubyKernel.send(RubyKernel.java:2117)
    at org.jruby.RubyKernel$s$send.call(RubyKernel$s$send.gen:65535)
    at 
org.jruby.internal.runtime.methods.DynamicMethod.call(DynamicMethod.ja
va:181)
    at 
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:282)
    at 
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
71)
    at 
org.jruby.ast.FCallSpecialArgNode.interpret(FCallSpecialArgNode.java:
45)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
```

```
	at 
org.jruby.evaluator.ASTInterpreter.INTERPRET_BLOCK(ASTInterpreter.java
:111)
	at 
org.jruby.runtime.InterpretedBlock.evalBlockBody(InterpretedBlock.java
:374)
	at org.jruby.runtime.InterpretedBlock.yield(InterpretedBlock.java:
295)
	at 
org.jruby.runtime.InterpretedBlock.yieldSpecific(InterpretedBlock.java
:229)
	at org.jruby.runtime.Block.yieldSpecific(Block.java:99)
	at org.jruby.ast.ZYieldNode.interpret(ZYieldNode.java:25)
	at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
	at org.jruby.ast.RescueNode.executeBody(RescueNode.java:216)
	at 
org.jruby.ast.RescueNode.interpretWithJavaExceptions(RescueNode.java:
120)
	at org.jruby.ast.RescueNode.interpret(RescueNode.java:110)
	at 
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
	at 
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:120)
	at 
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:165)
	at 
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:272)
	at 
org.jruby.runtime.callsite.CachingCallSite.callBlock(CachingCallSite.j
ava:80)
	at 
org.jruby.runtime.callsite.CachingCallSite.callIter(CachingCallSite.ja
va:89)
	at 
org.jruby.ast.FCallSpecialArgBlockNode.interpret(FCallSpecialArgBlockN
ode.java:42)
	at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
	at org.jruby.ast.RescueNode.executeBody(RescueNode.java:216)
	at 
org.jruby.ast.RescueNode.interpretWithJavaExceptions(RescueNode.java:
120)
	at org.jruby.ast.RescueNode.interpret(RescueNode.java:110)
	at 
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
```

```
	at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:120)
	at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:134)
	at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:174)
	at
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:282)
	at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
71)
	at
org.jruby.ast.CallSpecialArgNode.interpret(CallSpecialArgNode.java:73)
	at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
	at
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
	at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:120)
	at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:134)
	at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:174)
	at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
69)
	at
org.jruby.ast.FCallSpecialArgNode.interpret(FCallSpecialArgNode.java:
45)
	at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
	at
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
	at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:120)
	at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:134)
	at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:174)
```

```
    at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
69)
    at
org.jruby.ast.CallSpecialArgNode.interpret(CallSpecialArgNode.java:73)
    at org.jruby.ast.LocalAsgnNode.interpret(LocalAsgnNode.java:123)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
    at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:120)
    at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:134)
    at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:174)
    at
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:282)
    at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
71)
    at
org.jruby.ast.FCallManyArgsNode.interpret(FCallManyArgsNode.java:60)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.RootNode.interpret(RootNode.java:129)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_EVAL(ASTInterpreter.java:
95)
    at
org.jruby.evaluator.ASTInterpreter.evalWithBinding(ASTInterpreter.java
:166)
    at org.jruby.RubyKernel.evalCommon(RubyKernel.java:1155)
    at org.jruby.RubyKernel.eval(RubyKernel.java:1112)
    at org.jruby.RubyKernel$s$0$3$eval.call(RubyKernel$s$0$3$eval.gen:
65535)
    at
org.jruby.internal.runtime.methods.DynamicMethod.call(DynamicMethod.ja
va:181)
    at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
69)
    at
rubyjit.IRB::WorkSpace#evaluate_2B40A7B35DF97C801F371648C2CA40C78B0EDB
57.__file__(file:/usr/local/Cellar/hbase/1.2.6_2/libexec/lib/jruby-
complete-1.6.8.jar!/META-INF/jruby.home/lib/ruby/1.8/irb/workspace.rb:
81)
    at
rubyjit.IRB::WorkSpace#evaluate_2B40A7B35DF97C801F371648C2CA40C78B0EDB
57.__file__(file:/usr/local/Cellar/hbase/1.2.6_2/libexec/lib/jruby-
complete-1.6.8.jar!/META-INF/jruby.home/lib/ruby/1.8/irb/workspace.rb)
    at
org.jruby.internal.runtime.methods.JittedMethod.call(JittedMethod.java
:107)
    at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
69)
    at org.jruby.ast.CallManyArgsNode.interpret(CallManyArgsNode.java:
59)
    at org.jruby.ast.FCallOneArgNode.interpret(FCallOneArgNode.java:
36)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
    at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:233)
    at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:215)
    at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
201)
    at org.jruby.ast.CallTwoArgNode.interpret(CallTwoArgNode.java:59)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
    at org.jruby.ast.RescueNode.executeBody(RescueNode.java:216)
    at
org.jruby.ast.RescueNode.interpretWithJavaExceptions(RescueNode.java:
120)
    at org.jruby.ast.RescueNode.interpret(RescueNode.java:110)
    at org.jruby.ast.BeginNode.interpret(BeginNode.java:83)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_BLOCK(ASTInterpreter.java
:111)
    at
org.jruby.runtime.InterpretedBlock.evalBlockBody(InterpretedBlock.java
:374)
    at org.jruby.runtime.InterpretedBlock.yield(InterpretedBlock.java:
295)
```

```
    at
org.jruby.runtime.InterpretedBlock.yieldSpecific(InterpretedBlock.java
:229)
    at org.jruby.runtime.Block.yieldSpecific(Block.java:99)
    at
rubyjit.IRB::Irb#signal_status_6F6D59AFE4A05E3406BD6C100B588086254C188
B.chained_0_ensure_1$RUBY$__ensure__(file:/usr/local/Cellar/hbase/
1.2.6_2/libexec/lib/jruby-complete-1.6.8.jar!/META-INF/jruby.home/lib/
ruby/1.8/irb.rb:271)
    at
rubyjit.IRB::Irb#signal_status_6F6D59AFE4A05E3406BD6C100B588086254C188
B.__file__(file:/usr/local/Cellar/hbase/1.2.6_2/libexec/lib/jruby-
complete-1.6.8.jar!/META-INF/jruby.home/lib/ruby/1.8/irb.rb:270)
    at
rubyjit.IRB::Irb#signal_status_6F6D59AFE4A05E3406BD6C100B588086254C188
B.__file__(file:/usr/local/Cellar/hbase/1.2.6_2/libexec/lib/jruby-
complete-1.6.8.jar!/META-INF/jruby.home/lib/ruby/1.8/irb.rb)
    at
org.jruby.internal.runtime.methods.JittedMethod.call(JittedMethod.java
:187)
    at
org.jruby.runtime.callsite.CachingCallSite.callBlock(CachingCallSite.j
ava:176)
    at
org.jruby.runtime.callsite.CachingCallSite.callIter(CachingCallSite.ja
va:187)
    at
org.jruby.ast.FCallOneArgBlockNode.interpret(FCallOneArgBlockNode.java
:34)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_BLOCK(ASTInterpreter.java
:111)
    at
org.jruby.runtime.InterpretedBlock.evalBlockBody(InterpretedBlock.java
:374)
    at
org.jruby.runtime.InterpretedBlock.yieldSpecific(InterpretedBlock.java
:260)
    at org.jruby.runtime.Block.yieldSpecific(Block.java:117)
    at org.jruby.ast.YieldTwoNode.interpret(YieldTwoNode.java:31)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.IfNode.interpret(IfNode.java:117)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
    at org.jruby.ast.RescueNode.executeBody(RescueNode.java:216)
    at
org.jruby.ast.RescueNode.interpretWithJavaExceptions(RescueNode.java:
120)
```

```
    at org.jruby.ast.RescueNode.interpret(RescueNode.java:110)
    at org.jruby.ast.BeginNode.interpret(BeginNode.java:83)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_BLOCK(ASTInterpreter.java
:111)
    at
org.jruby.runtime.InterpretedBlock.evalBlockBody(InterpretedBlock.java
:374)
    at org.jruby.runtime.InterpretedBlock.yield(InterpretedBlock.java:
295)
    at
org.jruby.runtime.InterpretedBlock.yieldSpecific(InterpretedBlock.java
:229)
    at org.jruby.runtime.Block.yieldSpecific(Block.java:99)
    at org.jruby.RubyKernel.loop(RubyKernel.java:1439)
    at org.jruby.RubyKernel$s$0$0$loop.call(RubyKernel$s$0$0$loop.gen:
65535)
    at
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:302)
    at
org.jruby.runtime.callsite.CachingCallSite.callBlock(CachingCallSite.j
ava:144)
    at
org.jruby.runtime.callsite.CachingCallSite.callIter(CachingCallSite.ja
va:153)
    at
org.jruby.ast.FCallNoArgBlockNode.interpret(FCallNoArgBlockNode.java:
32)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_BLOCK(ASTInterpreter.java
:111)
    at
org.jruby.runtime.InterpretedBlock.evalBlockBody(InterpretedBlock.java
:374)
    at org.jruby.runtime.InterpretedBlock.yield(InterpretedBlock.java:
347)
    at org.jruby.runtime.InterpretedBlock.yield(InterpretedBlock.java:
304)
    at org.jruby.runtime.Block.yield(Block.java:130)
    at org.jruby.RubyContinuation.enter(RubyContinuation.java:106)
    at org.jruby.RubyKernel.rbCatch(RubyKernel.java:1212)
    at
org.jruby.RubyKernel$s$1$0$rbCatch.call(RubyKernel$s$1$0$rbCatch.gen:
65535)
```

```
    at
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:322)
    at
org.jruby.runtime.callsite.CachingCallSite.callBlock(CachingCallSite.j
ava:178)
    at
org.jruby.runtime.callsite.CachingCallSite.callIter(CachingCallSite.ja
va:187)
    at
org.jruby.ast.FCallOneArgBlockNode.interpret(FCallOneArgBlockNode.java
:34)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
    at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:169)
    at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:191)
    at
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:302)
    at
org.jruby.runtime.callsite.CachingCallSite.callBlock(CachingCallSite.j
ava:144)
    at
org.jruby.runtime.callsite.CachingCallSite.callIter(CachingCallSite.ja
va:153)
    at
org.jruby.ast.CallNoArgBlockNode.interpret(CallNoArgBlockNode.java:64)
    at org.jruby.ast.NewlineNode.interpret(NewlineNode.java:104)
    at org.jruby.ast.BlockNode.interpret(BlockNode.java:71)
    at
org.jruby.evaluator.ASTInterpreter.INTERPRET_METHOD(ASTInterpreter.jav
a:74)
    at
org.jruby.internal.runtime.methods.InterpretedMethod.call(InterpretedM
ethod.java:147)
    at
org.jruby.internal.runtime.methods.DefaultMethod.call(DefaultMethod.ja
va:183)
    at
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:292)
```

```
    at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
135)
    at usr.local.Cellar.hbase.$1_dot_2_dot_6_2.libexec.bin.
$_dot_dot_.bin.hirb.block_2$RUBY$start(/usr/local/Cellar/hbase/
1.2.6_2/libexec/bin/../bin/hirb.rb:205)
    at usr$local$Cellar$hbase$$1_dot_2_dot_6_2$libexec$bin$
$_dot_dot_$bin$hirb$block_2$RUBY$start.call(usr$local$Cellar$hbase$
$1_dot_2_dot_6_2$libexec$bin$$_dot_dot_$bin$hirb$block_2$RUBY$start:
65535)
    at org.jruby.runtime.CompiledBlock.yield(CompiledBlock.java:112)
    at org.jruby.runtime.CompiledBlock.yield(CompiledBlock.java:95)
    at org.jruby.runtime.Block.yield(Block.java:130)
    at org.jruby.RubyContinuation.enter(RubyContinuation.java:106)
    at org.jruby.RubyKernel.rbCatch(RubyKernel.java:1212)
    at
org.jruby.RubyKernel$s$1$0$rbCatch.call(RubyKernel$s$1$0$rbCatch.gen:
65535)
    at
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:322)
    at
org.jruby.runtime.callsite.CachingCallSite.callBlock(CachingCallSite.j
ava:178)
    at
org.jruby.runtime.callsite.CachingCallSite.callIter(CachingCallSite.ja
va:187)
    at usr.local.Cellar.hbase.$1_dot_2_dot_6_2.libexec.bin.
$_dot_dot_.bin.hirb.method__5$RUBY$start(/usr/local/Cellar/hbase/
1.2.6_2/libexec/bin/../bin/hirb.rb:204)
    at usr$local$Cellar$hbase$$1_dot_2_dot_6_2$libexec$bin$
$_dot_dot_$bin$hirb$method__5$RUBY$start.call(usr$local$Cellar$hbase$
$1_dot_2_dot_6_2$libexec$bin$$_dot_dot_$bin$hirb$method__5$RUBY$start:
65535)
    at
org.jruby.internal.runtime.methods.DynamicMethod.call(DynamicMethod.ja
va:203)
    at
org.jruby.internal.runtime.methods.CompiledMethod.call(CompiledMethod.
java:255)
    at
org.jruby.runtime.callsite.CachingCallSite.cacheAndCall(CachingCallSit
e.java:292)
    at
org.jruby.runtime.callsite.CachingCallSite.call(CachingCallSite.java:
135)
    at usr.local.Cellar.hbase.$1_dot_2_dot_6_2.libexec.bin.
$_dot_dot_.bin.hirb.__file__(/usr/local/Cellar/hbase/1.2.6_2/libexec/
bin/../bin/hirb.rb:210)
```

```
    at usr.local.Cellar.hbase.$1_dot_2_dot_6_2.libexec.bin.
$_dot_dot_.bin.hirb.load(/usr/local/Cellar/hbase/1.2.6_2/libexec/
bin/../bin/hirb.rb)
    at org.jruby.Ruby.runScript(Ruby.java:697)
    at org.jruby.Ruby.runScript(Ruby.java:690)
    at org.jruby.Ruby.runNormally(Ruby.java:597)
    at org.jruby.Ruby.runFromMain(Ruby.java:446)
    at org.jruby.Main.doRunFromMain(Main.java:369)
    at org.jruby.Main.internalRun(Main.java:258)
    at org.jruby.Main.run(Main.java:224)
    at org.jruby.Main.run(Main.java:208)
    at org.jruby.Main.main(Main.java:188)

ERROR: Failed 1 action: Table 'user_log' was not found, got:
user_friend.: 1 time,

Here is some help for this command:
Put a cell 'value' at specified table/row/column and optionally
timestamp coordinates.  To put a cell value into table 'ns1:t1' or
't1'
at row 'r1' under column 'c1' marked with the time 'ts1', do:

  hbase> put 'ns1:t1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value'
  hbase> put 't1', 'r1', 'c1', 'value', ts1
  hbase> put 't1', 'r1', 'c1', 'value',
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1,
{ATTRIBUTES=>{'mykey'=>'myvalue'}}
  hbase> put 't1', 'r1', 'c1', 'value', ts1, {VISIBILITY=>'PRIVATE|
SECRET'}

The same commands also can be run on a table reference. Suppose you
had a reference
t to table 't1', the corresponding command would be:

  hbase> t.put 'r1', 'c1', 'value', ts1,
{ATTRIBUTES=>{'mykey'=>'myvalue'}}


hbase(main):032:0> put 'access_log', 'row1','user:name','Sneha'
0 row(s) in 0.0070 seconds

hbase(main):033:0> put 'access_log', 'row1','user:ID','1'
0 row(s) in 0.0040 seconds

hbase(main):034:0> put 'access_log', 'row1','log:time','1pm'
0 row(s) in 0.0050 seconds
```

```
hbase(main):035:0> put 'access_log', 'row1','log:ip','10.10.1.70'
0 row(s) in 0.0030 seconds

hbase(main):036:0> put 'access_log', 'row1','log:url','g.com'
0 row(s) in 0.0070 seconds

hbase(main):037:0> scan 'access_log
hbase(main):038:0' scan 'access_log'
hbase(main):039:0' '
SyntaxError: (hbase):38: syntax error, unexpected tIDENTIFIER

scan 'access_log'
               ^

hbase(main):040:0> scan 'access_log'
ROW                            COLUMN+CELL
 row1                          column=log:ip,
timestamp=1528938077930, value=10.10.1.70
 row1                          column=log:time,
timestamp=1528938052733, value=1pm
 row1                          column=log:url,
timestamp=1528938095874, value=g.com
 row1                          column=user:ID,
timestamp=1528938002044, value=1
 row1                          column=user:name,
timestamp=1528937989398, value=Sneha
1 row(s) in 0.0080 seconds

hbase(main):041:0> scan 'student_course'
ROW                                                COLUMN+CELL
 row
column=course_cf2:ID, timestamp=1528934897050, value=CS590
 row
column=course_cf2:Instructor, timestamp=1528934915581, value=MAyanka
 row
column=course_cf2:University, timestamp=1528934930512, value=UMKC
 row
column=student_cf1:ID, timestamp=1528934816985, value=1
 row
column=student_cf1:name, timestamp=1528934866397, value=sneha
 row1
column=course_cf2:ID, timestamp=1528934998214, value=CS490
 row1
column=course_cf2:Instructor, timestamp=1528934983926, value=Jason
 row1
column=course_cf2:University, timestamp=1528934970696, value=UCM
 row1
column=student_cf1:ID, timestamp=1528935039557, value=2
```

```
 row1
column=student_cf1:name, timestamp=1528935017471, value=smita
2 row(s) in 0.0460 seconds

hbase(main):042:0> scan 'user_actiuon'
ROW                                                 COLUMN+CELL

ERROR: Unknown table user_actiuon!

Here is some help for this command:
Scan a table; pass table name and optionally a dictionary of scanner
specifications.  Scanner specifications may include one or more of:
TIMERANGE, FILTER, LIMIT, STARTROW, STOPROW, ROWPREFIXFILTER,
TIMESTAMP,
MAXLENGTH or COLUMNS, CACHE or RAW, VERSIONS, ALL_METRICS or METRICS

If no columns are specified, all columns will be scanned.
To scan all members of a column family, leave the qualifier empty as
in
'col_family'.

The filter can be specified in two ways:
1. Using a filterString - more information on this is available in the
Filter Language document attached to the HBASE-4176 JIRA
2. Using the entire package name of the filter.

If you wish to see metrics regarding the execution of the scan, the
ALL_METRICS boolean should be set to true. Alternatively, if you would
prefer to see only a subset of the metrics, the METRICS array can be
defined to include the names of only the metrics you care about.

Some examples:

  hbase> scan 'hbase:meta'
  hbase> scan 'hbase:meta', {COLUMNS => 'info:regioninfo'}
  hbase> scan 'ns1:t1', {COLUMNS => ['c1', 'c2'], LIMIT => 10,
STARTROW => 'xyz'}
  hbase> scan 't1', {COLUMNS => ['c1', 'c2'], LIMIT => 10, STARTROW =>
'xyz'}
  hbase> scan 't1', {COLUMNS => 'c1', TIMERANGE => [1303668804,
1303668904]}
  hbase> scan 't1', {REVERSED => true}
  hbase> scan 't1', {ALL_METRICS => true}
  hbase> scan 't1', {METRICS => ['RPC_RETRIES', 'ROWS_FILTERED']}
  hbase> scan 't1', {ROWPREFIXFILTER => 'row2', FILTER => "
    (QualifierFilter (>=, 'binary:xyz')) AND (TimestampsFilter ( 123,
456))"}
  hbase> scan 't1', {FILTER =>
    org.apache.hadoop.hbase.filter.ColumnPaginationFilter.new(1, 0)}
  hbase> scan 't1', {CONSISTENCY => 'TIMELINE'}
For setting the Operation Attributes
  hbase> scan 't1', { COLUMNS => ['c1', 'c2'], ATTRIBUTES => {'mykey'
=> 'myvalue'}}
  hbase> scan 't1', { COLUMNS => ['c1', 'c2'], AUTHORIZATIONS =>
['PRIVATE','SECRET']}
For experts, there is an additional option -- CACHE_BLOCKS -- which
switches block caching for the scanner on (true) or off (false).  By
default it is enabled.  Examples:

  hbase> scan 't1', {COLUMNS => ['c1', 'c2'], CACHE_BLOCKS => false}

Also for experts, there is an advanced option -- RAW -- which
instructs the
scanner to return all cells (including delete markers and uncollected
deleted
cells). This option cannot be combined with requesting specific
COLUMNS.
Disabled by default.  Example:

  hbase> scan 't1', {RAW => true, VERSIONS => 10}

Besides the default 'toStringBinary' format, 'scan' supports custom
formatting
by column.  A user can define a FORMATTER by adding it to the column
name in
the scan specification.  The FORMATTER can be stipulated:

 1. either as a org.apache.hadoop.hbase.util.Bytes method name (e.g,
toInt, toString)
 2. or as a custom class followed by method name: e.g.
'c(MyFormatterClass).format'.

Example formatting cf:qualifier1 and cf:qualifier2 both as Integers:
  hbase> scan 't1', {COLUMNS => ['cf:qualifier1:toInt',
    'cf:qualifier2:c(org.apache.hadoop.hbase.util.Bytes).toInt'] }

Note that you can specify a FORMATTER by column only (cf:qualifier).
You cannot
specify a FORMATTER for all columns of a column family.

Scan can also be used directly from a table, by first getting a
reference to a
table, like such:

  hbase> t = get_table 't'
  hbase> t.scan
```

```
Note in the above situation, you can still provide all the filtering,
columns,
options, etc as described above.



hbase(main):043:0> scan 'user_action'
ROW                                                  COLUMN+CELL
 row1                                                column=action:,
timestamp=1528937412865, value=register
 row1                                                column=user:ID,
timestamp=1528937380374, value=1
 row1                                                column=user:name,
timestamp=1528937396551, value=Sneha
 row2                                                column=action:,
timestamp=1528937438186, value=login
 row2                                                column=user:ID,
timestamp=1528937478482, value=1
 row2                                                column=user:name,
timestamp=1528937469465, value=Sneha
 row3                                                column=action:,
timestamp=1528937526284, value=login
 row3                                                column=user:ID,
timestamp=1528937515104, value=2
 row3                                                column=user:name,
timestamp=1528937503720, value=Aditya
3 row(s) in 0.0080 seconds

hbase(main):044:0>
```