```
Last login: Sun Jun 17 23:40:37 on ttys000
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 ..
Snehas-MacBook-Pro:bigdata snehamishra$ cd ..
Snehas-MacBook-Pro:Documents snehamishra$ cd ..
Snehas-MacBook-Pro:~ snehamishra$ cd ..
Snehas-MacBook-Pro:Users 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 bin/
Snehas-MacBook-Pro:bin snehamishra$ ls
hbase        start-hbase.sh   stop-hbase.sh
Snehas-MacBook-Pro:bin snehamishra$ 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:bin snehamishra$ cd hbase
-bash: cd: hbase: Not a directory
Snehas-MacBook-Pro:bin snehamishra$ hbase shell
2018-06-17 23:45:13,606 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> scan location
NameError: undefined local variable or method `location' for #<Object:
0x609e57da>

hbase(main):002:0> 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 0.2280 seconds

hbase(main):003:0> create 'facebook','cf1','cf2','cf3'
0 row(s) in 1.3900 seconds

=> Hbase::Table - facebook
hbase(main):004:0> describe 'facebook'
Table facebook is ENABLED
facebook
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1', 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 => 'cf2', 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 => 'cf3', 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'}
3 row(s) in 0.0450 seconds

hbase(main):005:0> scan 'facebook'
ROW                   COLUMN+CELL
0 row(s) in 0.0250 seconds

hbase(main):006:0> alter 'facebook','delete'=>'cf1','cf2'
SyntaxError: (hbase):7: syntax error, unexpected end-of-file


hbase(main):007:0> alter 'facebook','delete'=>'cf1'
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9670 seconds

hbase(main):008:0> alter 'facebook','delete'=>'cf2'
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9150 seconds

hbase(main):009:0> scan 'facebook'
ROW                   COLUMN+CELL
0 row(s) in 0.0130 seconds

hbase(main):010:0> describe 'facebook'
Table facebook is ENABLED
facebook
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf3', 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'}
1 row(s) in 0.0180 seconds

hbase(main):011:0> alter 'facebook','cf1_terms'
```

```
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9270 seconds

hbase(main):012:0> describe 'facebook'
Table facebook is ENABLED
facebook
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1_terms', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DE
LETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
'FOREVER', COMPRESSION => 'N
ONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536',
REPLICATION_SCOPE =
> '0'}
{NAME => 'cf3', 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'}
2 row(s) in 0.0270 seconds

hbase(main):013:0> alter 'facebook','cf2_user'
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9140 seconds

hbase(main):014:0> describe 'facebook'
Table facebook is ENABLED
facebook
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1_terms', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DE
LETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
'FOREVER', COMPRESSION => 'N
ONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536',
REPLICATION_SCOPE =
> '0'}
{NAME => 'cf2_user', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DEL
ETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
'FOREVER', COMPRESSION => 'NO
NE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536',
REPLICATION_SCOPE =>
 '0'}
```

```
{NAME => 'cf3', 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'}
3 row(s) in 0.0150 seconds

hbase(main):015:0> alter 'facebook','delete'=>'cf3'
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9120 seconds

hbase(main):016:0> describe 'facebook'
Table facebook is ENABLED
facebook
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1_terms', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DE
LETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
'FOREVER', COMPRESSION => 'N
ONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536',
REPLICATION_SCOPE =
> '0'}
{NAME => 'cf2_user', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DEL
ETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
'FOREVER', COMPRESSION => 'NO
NE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536',
REPLICATION_SCOPE =>
 '0'}
2 row(s) in 0.0160 seconds

hbase(main):017:0> put 'facebook','cf1_terms:address','mission_KS'

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):018:0> put
'facebook','row1','cf1_terms:address','mission_KS'
0 row(s) in 0.0540 seconds

hbase(main):019:0> scan 'facebook'
ROW                      COLUMN+CELL
 row1                    column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
1 row(s) in 0.0140 seconds

hbase(main):020:0> put 'facebook','row1','cf2_user:user1','1'
0 row(s) in 0.0050 seconds

hbase(main):021:0> put 'facebook','row1','cf2_user:user1:msg1','10'
0 row(s) in 0.0030 seconds

hbase(main):022:0> put 'facebook','row1','cf2_user:user1:msg2','1'
0 row(s) in 0.0040 seconds

hbase(main):023:0> scan 'facebook'
ROW                      COLUMN+CELL
 row1                    column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                    column=cf2_user:user1,
timestamp=1529300445870, value=1
1 row(s) in 0.0200 seconds

hbase(main):024:0> describe 'facebook'
Table facebook is ENABLED
facebook
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf1_terms', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_
CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER',
COMPRESSION => 'NONE', MIN_VE
```

```
RSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536',
REPLICATION_SCOPE => '0'}
{NAME => 'cf2_user', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_C
ELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER',
COMPRESSION => 'NONE', MIN_VER
SIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536',
REPLICATION_SCOPE => '0'}
2 row(s) in 0.0210 seconds

hbase(main):025:0> put 'facebook','row2','cf2_user:user2:msg1','3'
0 row(s) in 0.0140 seconds

hbase(main):026:0> scan 'facebook'
ROW                       COLUMN+CELL
 row1                     column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                     column=cf2_user:user1,
timestamp=1529300445870, value=1
 row2                     column=cf2_user:user2,
timestamp=1529300693177, value=3
2 row(s) in 0.0250 seconds

hbase(main):027:0> put 'facebook','row2','cf2_user:user2:msg2','5'
0 row(s) in 0.0040 seconds

hbase(main):028:0> scan 'facebook'
ROW                       COLUMN+CELL
 row1                     column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                     column=cf2_user:user1,
timestamp=1529300445870, value=1
 row2                     column=cf2_user:user2,
timestamp=1529300731539, value=5
2 row(s) in 0.0080 seconds

hbase(main):029:0> put 'facebook','row2','cf1_term:address','ranchi'

ERROR: Unknown column family! Valid column names: cf1_terms:*,
cf2_user:*

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):030:0> put 'facebook','row2','cf1_terms:address','ranchi'
0 row(s) in 0.0070 seconds

hbase(main):031:0> scan 'facebook'
ROW                      COLUMN+CELL
 row1                    column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                    column=cf2_user:user1,
timestamp=1529300445870, value=1
 row2                    column=cf1_terms:address,
timestamp=1529300845756, value=ranchi
 row2                    column=cf2_user:user2,
timestamp=1529300731539, value=5
2 row(s) in 0.0170 seconds

hbase(main):032:0> put
'facebook','row3','cf1_terms:contact','91320456'
0 row(s) in 0.0040 seconds

hbase(main):033:0> put 'facebook','row3','cf2_user:user2','9'
0 row(s) in 0.0030 seconds

hbase(main):034:0> put 'facebook','row3','cf2_user:user3','7'
0 row(s) in 0.0030 seconds

hbase(main):035:0> scan 'facebook'
ROW                      COLUMN+CELL
 row1                    column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                    column=cf2_user:user1,
timestamp=1529300445870, value=1
 row2                    column=cf1_terms:address,
timestamp=1529300845756, value=ranchi
```

```
 row2                      column=cf2_user:user2,
timestamp=1529300731539, value=5
 row3                      column=cf1_terms:contact,
timestamp=1529300909293, value=91320456
 row3                      column=cf2_user:user2,
timestamp=1529300941615, value=9
 row3                      column=cf2_user:user3,
timestamp=1529300949678, value=7
3 row(s) in 0.0170 seconds

hbase(main):036:0> SELECT * FROM 'facebook'
SyntaxError: (hbase):36: syntax error, unexpected tSTRING_BEG

SELECT * FROM 'facebook'
              ^

hbase(main):037:0> SELECT * FROM facebook
SyntaxError: (hbase):37: syntax error, unexpected tIDENTIFIER

SELECT * FROM facebook
                    ^

hbase(main):038:0> USE hbase
NameError: undefined local variable or method `hbase' for #<Object:
0x609e57da>

hbase(main):039:0> put 'facebook','row4','cf1_terms:name','Sneha'
0 row(s) in 0.0040 seconds

hbase(main):040:0> put 'facebook','row4','cf2_user:user3','1'
0 row(s) in 0.0030 seconds

hbase(main):041:0> scan 'facebook'
ROW                       COLUMN+CELL
 row1                      column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                      column=cf2_user:user1,
timestamp=1529300445870, value=1
 row2                      column=cf1_terms:address,
timestamp=1529300845756, value=ranchi
 row2                      column=cf2_user:user2,
timestamp=1529300731539, value=5
 row3                      column=cf1_terms:contact,
timestamp=1529300909293, value=91320456
 row3                      column=cf2_user:user2,
timestamp=1529300941615, value=9
 row3                      column=cf2_user:user3,
timestamp=1529300949678, value=7
```

```
 row4                    column=cf1_terms:name,
timestamp=1529301914418, value=Sneha
 row4                    column=cf2_user:user3,
timestamp=1529301942809, value=1
4 row(s) in 0.0220 seconds

hbase(main):042:0> scan 'facebook',{COLUMNS=>'cf1_terms'}
ROW                      COLUMN+CELL
 row1                    column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row2                    column=cf1_terms:address,
timestamp=1529300845756, value=ranchi
 row3                    column=cf1_terms:contact,
timestamp=1529300909293, value=91320456
 row4                    column=cf1_terms:name,
timestamp=1529301914418, value=Sneha
4 row(s) in 0.0210 seconds

hbase(main):043:0> scan 'facebook',{FILTER=>'Sneha'}
ROW                      COLUMN+CELL

ERROR: Incorrect Filter String

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):044:0> scan 'facebook',{FILTER=>"PrefixFilter('3')"}
ROW                       COLUMN+CELL

ERROR: Failed to get result within timeout, timeout=60000ms

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):045:0> scan 'facebook',{FILTER=>"PrefixFilter('3')"}
ROW                       COLUMN+CELL
0 row(s) in 0.0350 seconds

hbase(main):046:0> scan 'facebook',{FILTER=>"PrefixFilter('row3')"}
ROW                       COLUMN+CELL
 row3                     column=cf1_terms:contact,
timestamp=1529300909293, value=91320456
 row3                     column=cf2_user:user2,
timestamp=1529300941615, value=9
 row3                     column=cf2_user:user3,
timestamp=1529300949678, value=7
1 row(s) in 0.0150 seconds

hbase(main):047:0> scan 'facebook',
{FILTER=>"MultipleColumnPrefixFilter('user2')"}
ROW                       COLUMN+CELL
 row2                     column=cf2_user:user2,
timestamp=1529300731539, value=5
 row3                     column=cf2_user:user2,
timestamp=1529300941615, value=9
2 row(s) in 0.0250 seconds

hbase(main):048:0> scan 'facebook',{FILTER=>"ColumnCountGetFilter(0)"}
ROW                       COLUMN+CELL
0 row(s) in 0.0170 seconds

hbase(main):049:0> scan 'facebook',{FILTER=>"ColumnCountGetFilter(1)"}
ROW                       COLUMN+CELL
```

```
 row1                      column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row2                      column=cf1_terms:address,
timestamp=1529300845756, value=ranchi
 row3                      column=cf1_terms:contact,
timestamp=1529300909293, value=91320456
 row4                      column=cf1_terms:name,
timestamp=1529301914418, value=Sneha
4 row(s) in 0.0280 seconds

hbase(main):050:0> scan 'facebook',{FILTER=>"ColumnCountGetFilter(2)"}
ROW                        COLUMN+CELL
 row1                      column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                      column=cf2_user:user1,
timestamp=1529300445870, value=1
 row2                      column=cf1_terms:address,
timestamp=1529300845756, value=ranchi
 row2                      column=cf2_user:user2,
timestamp=1529300731539, value=5
 row3                      column=cf1_terms:contact,
timestamp=1529300909293, value=91320456
 row3                      column=cf2_user:user2,
timestamp=1529300941615, value=9
 row4                      column=cf1_terms:name,
timestamp=1529301914418, value=Sneha
4 row(s) in 0.0100 seconds

hbase(main):051:0> scan 'facebook',{FILTER=>"ColumnCountGetFilter(3)"}
ROW                        COLUMN+CELL
 row1                      column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                      column=cf2_user:user1,
timestamp=1529300445870, value=1
 row2                      column=cf1_terms:address,
timestamp=1529300845756, value=ranchi
 row2                      column=cf2_user:user2,
timestamp=1529300731539, value=5
 row3                      column=cf1_terms:contact,
timestamp=1529300909293, value=91320456
 row3                      column=cf2_user:user2,
timestamp=1529300941615, value=9
 row3                      column=cf2_user:user3,
timestamp=1529300949678, value=7
 row4                      column=cf1_terms:name,
timestamp=1529301914418, value=Sneha
 row4                      column=cf2_user:user3,
timestamp=1529301942809, value=1
4 row(s) in 0.0110 seconds
```

```
hbase(main):052:0> scan 'facebook',{FILTER=>"ColumnCountGetFilter(4)"}
ROW                          COLUMN+CELL
 row1                        column=cf1_terms:address,
timestamp=1529298834841, value=mission_KS
 row1                        column=cf2_user:user1,
timestamp=1529300445870, value=1
 row2                        column=cf1_terms:address,
timestamp=1529300845756, value=ranchi
 row2                        column=cf2_user:user2,
timestamp=1529300731539, value=5
 row3                        column=cf1_terms:contact,
timestamp=1529300909293, value=91320456
 row3                        column=cf2_user:user2,
timestamp=1529300941615, value=9
 row3                        column=cf2_user:user3,
timestamp=1529300949678, value=7
 row4                        column=cf1_terms:name,
timestamp=1529301914418, value=Sneha
 row4                        column=cf2_user:user3,
timestamp=1529301942809, value=1
4 row(s) in 0.0110 seconds

hbase(main):053:0> exit
Snehas-MacBook-Pro:bin snehamishra$ ls
hbase        start-hbase.sh   stop-hbase.sh
Snehas-MacBook-Pro:bin snehamishra$ stop-hbase.sh
stopping hbase....................
localhost: stopping zookeeper.
Snehas-MacBook-Pro:bin snehamishra$
```