

NoSQL Database HBase - Truck IoT Data

1 NoSQL and HBase

1.1 NoSQL - Column Family Database

Column family databases are best known because of Google's BigTable implementation. They are very similar on the surface to a relational database, but they have critical conceptual differences. You will not be able to apply the same sort of solutions that you used in a relational database to a column database.

That is because column databases are not relational; they do not have what an RDBMS would recognize as a table.

Each row consists of a collection of columns/value pairs in column-family databases. A collection of similar rows then makes up a column family. (This would be equivalent to a collection of rows making up a table in a relational database.) The main difference is that rows do not have to contain the same columns in a column-family database.

RDBMS imposes a high cost of schema change for the low cost of query change. Column families impose little to no cost in schema change, for slightly more cost in query change.

1.2 Why HBase:

HBase is a column-oriented database that prides itself on consistency and scaling out. It is based

on BigTable, a high-performance, proprietary database developed by Google and described in the 2006 white paper "Bigtable: A Distributed Storage System for Structured Data."¹

Initially

created for natural language processing, HBase started life as a contrib package for Apache Hadoop. Since then, it has become a top-level Apache project.

2 Starting the HBase shell

The HBase shell is a JRuby-based command-line program we can use to interact with HBase. In

the shell, we can add and remove tables, alter table schema, add or delete data, and do many other tasks. Later we'll explore other means of connecting to HBase, but the shell will now be our home.

Typed 'hbase shell' to start the HBase shell.

```
tdend2@node00:~  
[tdend2@node00 ~]$ hbase shell  
HBase Shell  
Use "help" to get list of supported commands.  
Use "exit" to quit this interactive shell.  
For Reference, please visit: http://hbase.apache.org/2.0/book.html#shell  
Version 2.1.0-cdh6.3.2, rUnknown, Fri Nov 8 05:44:07 PST 2019  
Took 0.0012 seconds  
hbase(main):001:0> status  
1 active master, 0 backup masters, 8 servers, 0 dead, 2.5000 average load  
Took 0.4955 seconds  
hbase(main):002:0>
```

Helpcommand:

```
tdend2@node00:~  
hbase(main):002:0> help  
HBase Shell, version 2.1.0-cdh6.3.2, rUnknown, Fri Nov 8 05:44:07 PST 2019  
Type 'help "COMMAND"', (e.g. 'help "get"' -- the quotes are necessary) for help on a specific command.  
Commands are grouped. Type 'help "COMMAND_GROUP"', (e.g. 'help "general"') for help on a command group.  
  
COMMAND GROUPS:  
  Group name: general  
  Commands: processlist, status, table_help, version, whoami  
  
  Group name: ddl  
  Commands: alter, alter_async, alter_status, clone_table_schema, create, describe, disable, disable_all, drop, drop_all,  
list_regions, locate_region, show_filters  
  
  Group name: namespace  
  Commands: alter_namespace, create_namespace, describe_namespace, drop_namespace, list_namespace, list_namespace_tables  
  
  Group name: dml  
  Commands: append, count, delete, deleteall, get, get_counter, get_splits, incr, put, scan, truncate, truncate_preserve  
  
  Group name: tools  
  Commands: assign, balance_switch, balancer, balancer_enabled, catalogjanitor_enabled, catalogjanitor_run, catalogjanitor  
h, clear_block_cache, clear_compaction_queues, clear_deadservers, close_region, compact, compact_rs, compaction_state, fl  
ion, move, normalize, normalizer_enabled, normalizer_switch, split, splitormerge_enabled, splitormerge_switch, stop_maste  
  
  Group name: replication  
  Commands: add_peer, append_peer_exclude_namespaces, append_peer_exclude_tableCFs, append_peer_namespaces, append_peer_t  
able_replication, get_peer_config, list_peer_configs, list_peers, list_replicated_tables, remove_peer, remove_peer_exclud  
ove_peer_tableCFs, set_peer_bandwidth, set_peer_exclude_namespaces, set_peer_exclude_tableCFs, set_peer_namespaces, set_p  
Fs, update_peer_config  
  
  Group name: snapshots  
  Commands: clone_snapshot, delete_all_snapshot, delete_snapshot, delete_table_snapshots, list_snapshots, list_table_snap  
  
  Group name: configuration  
  Commands: update_all_config, update_config  
  
  Group name: quotas  
  Commands: list_quota_snapshots, list_quota_table_sizes, list_quotas, list_snapshot_sizes, set_quota  
  
  Group name: security  
  Commands: grant, list_security_capabilities, revoke, user_permission  
  
  Group name: procedures  
  Commands: list_locks, list_procedures  
  
  Group name: visibility labels  
  Commands: add_labels, clear_auths, get_auths, list_labels, set_auths, set_visibility  
  
  Group name: rsgroup  
  Commands: add_rsgroup, balance_rsgroup, get_rsgroup, get_server_rsgroup, get_table_rsgroup, list_rsgroups, move_namespa
```

Last half of result:

tdend2@node00:~

Commands: add_peer, append_peer_exclude_namespaces, append_peer_exclude_tableCFs, append_peer_namespaces, append_peer_tableCFs, append_peer_table_replication, get_peer_config, list_peer_configs, list_peers, list_replicated_tables, remove_peer, remove_peer_tableCFs, set_peer_bandwidth, set_peer_exclude_namespaces, set_peer_exclude_tableCFs, set_peer_namespaces, update_peer_config

Group name: snapshots

Commands: clone_snapshot, delete_all_snapshot, delete_snapshot, delete_table_snapshots, list_snapshots

Group name: configuration

Commands: update_all_config, update_config

Group name: quotas

Commands: list_quota_snapshots, list_quota_table_sizes, list_quotas, list_snapshot_sizes, set_quota

Group name: security

Commands: grant, list_security_capabilities, revoke, user_permission

Group name: procedures

Commands: list_locks, list_procedures

Group name: visibility labels

Commands: add_labels, clear_auths, get_auths, list_labels, set_auths, set_visibility

Group name: rsgroup

Commands: add_rsgroup, balance_rsgroup, get_rsgroup, get_server_rsgroup, get_table_rsgroup, list_rsgroups, list_servers_tables_rsgroup, move_tables_rsgroup, remove_rsgroup, remove_servers_rsgroup

SHELL USAGE:

Quote all names in HBase Shell such as table and column names. Commas delimit command parameters. Type <RETURN> after entering a command to run it.

Dictionaries of configuration used in the creation and alteration of tables are Ruby Hashes. They look like this:

```
{'key1' => 'value1', 'key2' => 'value2', ...}
```

and are opened and closed with curly-braces. Key/values are delimited by the '=' character combination. Usually keys are predefined constants such as TABLE_NAME, VERSIONS, COMPRESSION, etc. Constants do not need to be quoted. Type 'Object.constants' to see a (messy) list of all constants in the environment.

If you are using binary keys or values and need to enter them in the shell, use double-quoted hexadecimal representation. For example:

```
hbase> get 't1', "key\x03\x3f\xcd"
hbase> get 't1', "key\003\023\011"
hbase> put 't1', "test\xef\xff", 'f1:', "\x01\x33\x40"
```

The HBase shell is the (J)Ruby IRB with the above HBase-specific commands added.

For more on the HBase Shell, see <http://hbase.apache.org/book.html>

hbase(main):003:0>

```

tdend2@node00:~
hbase(main):003:0> Object.constants
=> [:Fixnum, :RUBY_PLATFORM, :STDERR, :Rational, :MapJavaProxy, :RESERVED_SIGNALS, :String, :Math,
:Numeric, :SystemExit, :VisibilityClient, :Enumerator, :JavaInterfaceTemplate, :InterruptedException,
:IOException, :STDOUT, :Pathname, :Method, :NotImplementedError, :Mutex, :Interrupt, :Gem, :Float,
:Y_PATCHLEVEL, :Object, :Module, :Marshal, :FloatDomainError, :Arrays, :ClosedQueueError, :Hbase,
:ArrayJavaProxy, :URI, :Complex, :Proc, :D_ARG, :Signal, :RUBY_REVISION, :JBoolean, :RbConfig, :Th
, :Java, :Pattern, :HBaseConstants, :StopIteration, :Dir, :RUBY_ENGINE, :MonitorMixin, :Thread, :
erConfigUtil, :Comparable, :RUBY_ENGINE_VERSION, :HBaseQuotasConstants, :Regexp, :Readline, :CROS
nfig, :NameError, :RUBY_RELEASE_DATE, :Random, :SyntaxError, :NoMemoryError, :JRUBY_REVISION, :Ja
:SpanReceiverHost, :SpaceViolationPolicy, :TRUE, :Encoding, :NoMethodError, :Range, :QuotaSetting
JumpError, :Class, :ArrayJavaProxyCreator, :QuotaFilter, :Exception2MessageMapper, :RegionSplitte
nfiguration, :RubyLex, :Errno, :Fiber, :Time, :JSON, :ENV, :ARGF, :Struct, :StringIO, :ConditionV
or, :JavaProxyMethods, :RangeError, :GC, :Pair, :Continuation, :SIGNALS, :Data, :SignalException,
TOPLEVEL_BINDING, :Delegator, :IOError, :RegexpError, :UnboundMethod, :FALSE, :JavaPackageModuleT
hbase(main):004:0>

```

3.2 Data Definition Language (DDL) commands in HBase

3.2.1 Namespace

A namespace is a logical grouping of tables analogous to a database in relational database systems. This abstraction lays the groundwork for multi-tenancy-related features:

- Quota Management (HBASE-8410) - Restrict the amount of resources (e.g., regions, tables) a namespace can consume.
- Namespace Security Administration (HBASE-9206) - provide another level of security administration for tenants.
- Region server groups (HBASE-6721) - A namespace/table can be pinned onto a subset of region servers thus guaranteeing a coarse level of isolation.

create_namespace 'tdend2'

```

tdend2@node00:~
hbase(main):004:0> create_namespace 'tdend2'

ERROR: org.apache.hadoop.hbase.NamespaceExistException: Namespace tdend2 already exists
    at org.apache.hadoop.hbase.master.procedure.CreateNamespaceProcedure.prepareCreate(CreateNamespaceProc
    at org.apache.hadoop.hbase.master.procedure.CreateNamespaceProcedure.executeFromState(CreateNamespaceP
    at org.apache.hadoop.hbase.master.procedure.CreateNamespaceProcedure.executeFromState(CreateNamespaceP
    at org.apache.hadoop.hbase.procedure2.StateMachineProcedure.execute(StateMachineProcedure.java:189)
    at org.apache.hadoop.hbase.procedure2.Procedure.doExecute(Procedure.java:965)
    at org.apache.hadoop.hbase.procedure2.ProcedureExecutor.execProcedure(ProcedureExecutor.java:1742)
    at org.apache.hadoop.hbase.procedure2.ProcedureExecutor.executeProcedure(ProcedureExecutor.java:1481)
    at org.apache.hadoop.hbase.procedure2.ProcedureExecutor.access$1200(ProcedureExecutor.java:78)
    at org.apache.hadoop.hbase.procedure2.ProcedureExecutor$WorkerThread.run(ProcedureExecutor.java:2058)

For usage try 'help "create_namespace"'

Took 1.2944 seconds
hbase(main):005:0>

```

3.2.2 Create a table

When you create a table, you should add your namespace in front of your table name
create '<your-namespace>:safe_table', 'will_NOT_be_deleted'
create 'tdend2:safe_table', 'will_NOT_be_deleted'

```
tdend2@node00:~  
hbase(main):005:0> create 'tdend2:safe_table', 'will_NOT_be_deleted'  
Created table tdend2:safe_table  
Took 0.7960 seconds  
=> Hbase::Table - tdend2:safe_table  
hbase(main):006:0>
```

'list' command is you will see all the tables in the HBase. If 30 users create 5 tables per user, you will see 150 tables.

```
tdend2@node00:~  
hbase(main):006:0> list  
TABLE  
abeer:del_table  
abeer:mytable2_tbl  
abeer:mytable_tbl  
abeer:truck_event  
ibiga2:truck_event  
mkhal44:mytable2_tbl  
mkhal44:mytable_tbl  
mkhal44:truck_event  
nhasa29:mytable_tbl  
nhasa29:truck_event  
operv2:mytable2_tbl  
operv2:mytable_tbl  
operv2:truck_event  
spisa3:truck_event  
sslee777:table4compress  
sslee777:table4compress_c  
sslee777:truck_event  
tdend2:safe_table  
vkond9:truck_event  
19 row(s)  
Took 0.0292 seconds  
=> ["abeer:del_table", "abeer:mytable2_tbl", "abeer:mytable_tbl", "abeer:truck_event", "ibiga2:tr  
ble_tbl", "nhasa29:truck_event", "operv2:mytable2_tbl", "operv2:mytable_tbl", "operv2:truck_event  
k_event", "tdend2:safe_table", "vkond9:truck_event"]  
hbase(main):007:0>
```

The syntax to create a table in HBase is create '<table_name>','<column_family_name>'.
Let's create a table called 'mytable_tbl' with a column family of name 'mycolfam_cf'. Run the following command:

create '<your-namespace>:mytable_tbl', 'mycolfam_cf'
create 'tdend2:mytable_tbl', 'mycolfam_cf'

```
tdend2@node00:~  
hbase(main):007:0> create 'tdend2:mytable_tbl', 'mycolfam_cf'  
Created table tdend2:mytable_tbl  
Took 0.7410 seconds  
=> Hbase::Table - tdend2:mytable_tbl  
hbase(main):008:0>
```

The table is currently empty; it has no rows and thus no columns. Unlike a relational database, in HBase a column is specific to the row that contains it. It is only when adding rows that columns are created to store data.

Checked the table we've just created, typed the following command in the HBase shell

```

tdend2@node00:~
hbase(main):008:0> list
TABLE
abeer:del_table
abeer:mytable2_tbl
abeer:mytable_tbl
abeer:truck_event
ibiga2:truck_event
mkhal44:mytable2_tbl
mkhal44:mytable_tbl
mkhal44:truck_event
nhasa29:mytable_tbl
nhasa29:truck_event
operv2:mytable2_tbl
operv2:mytable_tbl
operv2:truck_event
spisa3:truck_event
sslee777:table4compress
sslee777:table4compress_c
sslee777:truck_event
tdend2:mytable_tbl
tdend2:safe_table
vkond9:truck_event
20 row(s)
Took 0.0118 seconds
=> ["abeer:del_table", "abeer:mytable2_tbl", "abeer:mytable_tbl", "abeer:truck_event", "ibiga2:tr
ble_tbl", "nhasa29:truck_event", "operv2:mytable2_tbl", "operv2:mytable_tbl", "operv2:truck_event
k_event", "tdend2:mytable_tbl", "tdend2:safe_table", "vkond9:truck_event"]
hbase(main):009:0>

```

'List_namespace_tables' displays only tables under a particular namespace will be show up.

`list_namespace_tables '<your-namespace>'`

`list_namespace_tables 'tdend2'`

```

tdend2@node00:~
hbase(main):009:0> list_namespace_tables 'tdend2'
TABLE
mytable_tbl
safe_table
2 row(s)
Took 0.0300 seconds
=> ["mytable_tbl", "safe_table"]
hbase(main):010:0>

```

3.2.3 Delete tables

If you want to delete/drop tables, use drop command. But we need to disable the tables first disable 'del_table'. I tried to create a table, disabled and deleted as shown below:


```

tdend2@node00:~
hbase(main):001:0> create 'tdend2:del_tbl', 'mycolfam_cf'

ERROR: Table already exists: tdend2:del_tbl!

For usage try 'help "create"'

Took 1.3579 seconds
hbase(main):002:0> disable 'tdend2:del_tbl', 'mycolfam_cf'

ERROR: wrong number of arguments (2 for 1)

For usage try 'help "disable"'

Took 0.0065 seconds
hbase(main):003:0> disable 'tdend2:del_tbl'
Took 1.0012 seconds
hbase(main):004:0> drop 'tdend2:del_tbl'
Took 0.2314 seconds
hbase(main):005:0> list_namespace_tables 'tdend2'
TABLE
mytable_tbl
safe_table
2 row(s)
Took 0.0234 seconds
=> ["mytable_tbl", "safe_table"]
hbase(main):006:0>

```

3.3 Data Manipulation Language (DML) commands in HBase

3.3.1 Inserting and Reading Data: put and get/scan commands

First, we will use the put command to add data to an HBase table.

put '<your-namespace>:mytable_tbl', 'myrowkey_key', 'mycolfam_cf:mycolname_col', 'Welcome to My HBase!'

put 'tdend2:mytable_tbl', 'myrowkey_key', 'mycolfam_cf:mycolname_col', 'Welcome to My HBase!'

```

tdend2@node00:~
hbase(main):006:0> put 'tdend2:mytable_tbl', 'myrowkey_key', 'mycolfam_cf:mycolname_col', 'Welcome to My HBase!'
Took 0.1063 seconds
hbase(main):007:0>

```

This command inserts a new row into the 'mytable_tbl' table with the key 'myrowkey_key', adding 'Welcome to My HBase!' to the column (column family+column name/qualifier) called 'mycolfam_cf:mycolname_col'.

We can query the data for the 'myrowkey_key' row using get, which requires two parameters: the table name and the row key.

We can also use 'scan' command.

Get command:

get 'tdend2:mytable_tbl', 'myrowkey_key'

```
tdend2@node00:~  
hbase(main):011:0> get 'tdend2:mytable_tbl', 'myrowkey_key'  
COLUMN                                CELL  
mycolfam_cf:mycolname_col            timestamp=1727459584874, value=Welcome to My HBase!  
1 row(s)  
Took 0.0404 seconds  
hbase(main):012:0>
```

Scan - command:

```
tdend2@node00:~  
Took 0.0831 seconds  
hbase(main):013:0> scan 'tdend2:mytable_tbl'  
ROW                                    COLUMN+CELL  
myrowkey_key                          column=mycolfam_cf:mycolname_col, timestamp=1727459584874, value=Welcome to My HBase!  
1 row(s)  
Took 0.0121 seconds  
hbase(main):014:0>
```

Notice the timestamp field in the output. HBase stores an integer timestamp for all data values, representing time in milliseconds since the epoch (00:00:00UTC on January 1, 1970). When a new value is written to the same cell, the old value will be retained, indexed by its timestamp. This is a great feature. Most databases require you to specifically handle historical data explicitly, but in HBase, versioning is automatic. If we do not want the timestamp to be in MSE, the put and get commands allow to specify a timestamp explicitly by inserting an integer value of your choice. This gives an extra dimension to work with if we need it. If we don't explicitly specify a timestamp, HBase will use the current time when inserting, and it will return the most recent version when reading.

3.4 Loading/populate data using HBase utility

This utility should be called outside the HBase shell—means in Linux shell.

3.4.1 Trucking IoT Data

It's time to get started with some real data. For this hands-on exercise, you'll make use of the Trucking IoT Data.

- Dataset:

https://www.cloudera.com/content/dam/www/marketing/tutorials/beginnersguide-to-apache-pig/assets/driver_data.zip

- Related GitHub project: <https://github.com/hortonworks-gallery/iot-truck-streaming>

We are looking at a use case where we have a truck fleet. Each truck has been equipped to log

location and event data. These events are streamed back to a datacenter where we will be processing the data. The company wants to use this data to better understand risk.

The dataset, Trucking IoT, contains the following files:

- drivers.csv

- o This has driver information. It contains records showing driverId, name, ssn, location, certified, and wage-plan.

- timesheet.csv

- o This contains records showing driverId, week, hours-logged, and miles-logged.

- truck_event_text_partition.csv

- o This contains records showing driverId, truckId, eventTime, eventType, longitude, latitude, eventKey, CorrelationId, driverName, routeId, routeName, and eventDate

The dataset is located in /home/data/CSC534BDA/datasets/Truck-IoT of Linux file system (Not in the HDFS) of the cluster.

```
tdend2@node00:~  
[tdend2@node00 ~]$ ls -alFh /home/data/CSC534BDA/datasets/Truck-IoT  
total 2.2M  
drwxrwxr-x 2 sslee777 sslee777 84 Sep 15 2020 ./  
drwxrwxr-x 6 sslee777 sslee777 76 Oct 22 2021 ../  
-rw-rw-r-- 1 sslee777 sslee777 2.0K Sep 15 2020 drivers.csv  
-rw-rw-r-- 1 sslee777 sslee777 26K Sep 15 2020 timesheet.csv  
-rw-rw-r-- 1 sslee777 sslee777 2.2M Sep 15 2020 truck_event_text_partition.csv  
[tdend2@node00 ~]$
```

We will use truck_event_text_partition.csv file for this exercise.

To see the first 10 rows of truck_event_text_partition.csv, use 'head' Linux commands.

head /home/data/CSC534BDA/datasets/Truck-IoT/truck_event_text_partition.csv

```
tdend2@node00:~  
[tdend2@node00 ~]$ head /home/data/CSC534BDA/datasets/Truck-IoT/truck_event_text_partition.csv  
driverId,truckId,eventTime,eventType,longitude,latitude,eventKey,CorrelationId,driverName,routeId,routeName,eventDate  
14,25,59:21.4,Normal,-94.58,37.03,14|25|9223370572464814373,3.66E+18,Adis Cesir,160405074,Joplin to Kansas City Route 2,2016-05-27-22  
18,16,59:21.7,Normal,-89.66,39.78,18|16|9223370572464814089,3.66E+18,Grant Liu,1565885487,Springfield to KC Via Hanibal,2016-05-27-22  
27,105,59:21.7,Normal,-90.21,38.65,27|105|9223370572464814070,3.66E+18,Mark Lochbihler,1325562373,Springfield to KC Via Columbia Route 2,2016-05-27-22  
11,74,59:21.7,Normal,-90.2,38.65,11|74|9223370572464814123,3.66E+18,Jamie Engesser,1567254452,Saint Louis to Memphis Route2,2016-05-27-22  
22,87,59:21.7,Normal,-90.04,35.19,22|87|9223370572464814101,3.66E+18,Nadeem Asghar,1198242881, Saint Louis to Chicago Route2,2016-05-27-22  
22,87,59:22.3,Normal,-90.37,35.21,22|87|9223370572464813486,3.66E+18,Nadeem Asghar,1198242881, Saint Louis to Chicago Route2,2016-05-27-22  
23,68,59:22.4,Normal,-89.91,40.86,23|68|9223370572464813450,3.66E+18,Adam Diaz,160405074,Joplin to Kansas City Route 2,2016-05-27-22  
11,74,59:22.5,Normal,-89.74,39.1,11|74|9223370572464813355,3.66E+18,Jamie Engesser,1567254452,Saint Louis to Memphis Route2,2016-05-27-22  
20,41,59:22.5,Normal,-93.36,41.69,20|41|9223370572464813344,3.66E+18,Chris Harris,160779139,Des Moines to Chicago Route 2,2016-05-27-22  
[tdend2@node00 ~]$
```

3.4.2 Creating a table

Created a table, truck_event, with column family name 'events'

```
create 'tdend2:truck_event', 'events'
```

```
tdend2@node00:~  
hbase(main):001:0> create 'tdend2:truck_event', 'events'  
Created table tdend2:truck_event  
Took 1.1908 seconds  
=> Hbase::Table - tdend2:truck_event  
hbase(main):002:0>
```

3.4.3 Load/populate data

The data is already loaded to HDFS and can be found in /user/data/CSC534BDA/Truck-IoT/ in HDFS. We will use

'eventKey' column as a RowKey (similar to the primary key in RDBMS) of the table, so 'eventKey' column is replaced by HBASE_ROW_KEY. For more detail, see ImportTsv utility reference guide in HBase: <http://hbase.apache.org/book.html#importtsv>

Note: Typing below codes is recommended! You may not be able to copy below codes perfectly, because PDF conversion is not good at long texts. Don't forget to use your namespace.

```
hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator=,  
-Dimporttsv.columns="  
events:driverId,events:truckId,events::eventTime,events:eventType,events:longitude,events:latitude,HBASE_ROW_KEY,events:CorrelationId,events:driverName,events:routId,events:routeName,events:eventDate" 'tdend2:truck_event'  
hdfs://node00.sun:8020/user/data/CSC534BDA/Truck-IoT/truck_event_text_partition.csv
```

```
tdend2@node00:~  
[tdend2@node00 ~]$ hbase org.apache.hadoop.hbase.mapreduce.ImportTsv \  
> -Dimporttsv.separator=', ' \  
> -Dimporttsv.columns="events:driverId,events:truckId,events:eventTime,events:eventType,events:lon-  
gitude,events:latitude,HBASE_ROW_KEY,events:CorrelationId,events:driverName,events:routId,events:routeName,events:eventDate" \  
> 'tdend2:truck_event' \  
> hdfs://node00.sun:8020/user/data/CSC534BDA/Truck-IoT/truck_event_text_partition.csv  
24/09/27 16:22:30 INFO zookeeper.ZooKeeper: Client environment:zookeeper.version=3.4.5-cdh6.3.2--  
24/09/27 16:22:30 INFO zookeeper.ZooKeeper: Client environment:host.name=node00.sun  
24/09/27 16:22:30 INFO zookeeper.ZooKeeper: Client environment:java.version=1.8.0_181  
24/09/27 16:22:30 INFO zookeeper.ZooKeeper: Client environment:java.vendor=Oracle Corporation  
24/09/27 16:22:30 INFO zookeeper.ZooKeeper: Client environment:java.home=/usr/java/jdk1.8.0_181-c
```

Last half of output after loading the data:

```

tdend2@node00:~
24/09/27 16:22:34 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
24/09/27 16:22:34 INFO impl.YarnClientImpl: Submitted application application_1722897143033_0623
24/09/27 16:22:34 INFO mapreduce.Job: The url to track the job: http://node00.sun:8088/proxy/app/
24/09/27 16:22:34 INFO mapreduce.Job: Running job: job_1722897143033_0623
24/09/27 16:22:42 INFO mapreduce.Job: Job job_1722897143033_0623 running in uber mode : false
24/09/27 16:22:42 INFO mapreduce.Job: map 0% reduce 0%
24/09/27 16:22:50 INFO mapreduce.Job: map 100% reduce 0%
24/09/27 16:22:50 INFO mapreduce.Job: Job job_1722897143033_0623 completed successfully
24/09/27 16:22:50 INFO mapreduce.Job: Counters: 34
    File System Counters
        FILE: Number of bytes read=0
        FILE: Number of bytes written=259447
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=2272225
        HDFS: Number of bytes written=0
        HDFS: Number of read operations=2
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=0
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Launched map tasks=1
        Rack-local map tasks=1
        Total time spent by all maps in occupied slots (ms)=5840
        Total time spent by all reduces in occupied slots (ms)=0
        Total time spent by all map tasks (ms)=5840
        Total vcore-milliseconds taken by all map tasks=5840
        Total megabyte-milliseconds taken by all map tasks=5980160
    Map-Reduce Framework
        Map input records=17076
        Map output records=17076
        Input split bytes=148
        Spilled Records=0
        Failed Shuffles=0
        Merged Map outputs=0
        GC time elapsed (ms)=97
        CPU time spent (ms)=6670
        Physical memory (bytes) snapshot=378802176
        Virtual memory (bytes) snapshot=2709729280
        Total committed heap usage (bytes)=322437120
        Peak Map Physical memory (bytes)=378802176
        Peak Map Virtual memory (bytes)=2709729280
    ImportTsv
        Bad Lines=0
    File Input Format Counters
        Bytes Read=2272077
    File Output Format Counters
        Bytes Written=0
[tdend2@node00 ~]$

```

17076 as Map input records and output records and no bad lines (Bad Lines = 0). The import job was successful. However, double-checked the row count of the table by submitting another MR job, RowCounter.

hbase org.apache.hadoop.hbase.mapreduce.RowCounter 'tdend2:truck_event'

```
tdend2@node00:~  
[tdend2@node00 ~]$ hbase org.apache.hadoop.hbase.mapreduce.RowCounter 'tdend2:truck_event'  
24/09/27 16:29:13 INFO client.RMProxy: Connecting to ResourceManager at node00.sun/10.0.0.10:8032  
24/09/27 16:29:14 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /user/td  
24/09/27 16:29:16 INFO zookeeper.ZooKeeper: Client environment:zookeeper.version=3.4.5-cdh6.3.2--  
24/09/27 16:29:16 INFO zookeeper.ZooKeeper: Client environment:host.name=node00.sun  
24/09/27 16:29:16 INFO zookeeper.ZooKeeper: Client environment:java.version=1.8.0_181  
24/09/27 16:29:16 INFO zookeeper.ZooKeeper: Client environment:java.vendor=Oracle Corporation  
24/09/27 16:29:16 INFO zookeeper.ZooKeeper: Client environment:java.home=/usr/java/jdk1.8.0_181-c  
24/09/27 16:29:16 INFO zookeeper.ZooKeeper: Client environment:java.class.path=/opt/cloudera/parc
```

scan 'tdend2:truck_event', {'LIMIT' => 2}

```
tdend2@node00:~  
hbase(main):001:0> scan 'tdend2:truck_event', {'LIMIT' => 2}  
ROW COLUMN+CELL  
10|23|9223370572126391280 column=events:correlationId, timestamp=1727472149999, value=1000  
10|23|9223370572126391280 column=events:driverId, timestamp=1727472149999, value=10  
10|23|9223370572126391280 column=events:driverName, timestamp=1727472149999, value=George Vetticaden  
10|23|9223370572126391280 column=events:eventDate, timestamp=1727472149999, value=2016-06-02-20  
10|23|9223370572126391280 column=events:eventTime, timestamp=1727472149999, value=59:44.5  
10|23|9223370572126391280 column=events:eventType, timestamp=1727472149999, value=Normal  
10|23|9223370572126391280 column=events:latitude, timestamp=1727472149999, value=38.64  
10|23|9223370572126391280 column=events:longitude, timestamp=1727472149999, value=-90.18  
10|23|9223370572126391280 column=events:routeId, timestamp=1727472149999, value=1390372503  
10|23|9223370572126391280 column=events:routeName, timestamp=1727472149999, value=Saint Louis to Tulsa  
10|23|9223370572126391280 column=events:truckId, timestamp=1727472149999, value=23  
10|23|9223370572126392279 column=events:correlationId, timestamp=1727472149999, value=1000  
10|23|9223370572126392279 column=events:driverId, timestamp=1727472149999, value=10  
10|23|9223370572126392279 column=events:driverName, timestamp=1727472149999, value=George Vetticaden  
10|23|9223370572126392279 column=events:eventDate, timestamp=1727472149999, value=2016-06-02-20  
10|23|9223370572126392279 column=events:eventTime, timestamp=1727472149999, value=59:43.5  
10|23|9223370572126392279 column=events:eventType, timestamp=1727472149999, value=Normal  
10|23|9223370572126392279 column=events:latitude, timestamp=1727472149999, value=38.64  
10|23|9223370572126392279 column=events:longitude, timestamp=1727472149999, value=-90.18  
10|23|9223370572126392279 column=events:routeId, timestamp=1727472149999, value=1390372503  
10|23|9223370572126392279 column=events:routeName, timestamp=1727472149999, value=Saint Louis to Tulsa  
10|23|9223370572126392279 column=events:truckId, timestamp=1727472149999, value=23  
2 row(s)  
Took 0.5325 seconds  
hbase(main):002:0>
```

2 rows are obtained as limited to get only 2 rows from the table and also found 11 columns in each row.

Please write and run HBase commands against the table created above.

1. Write and run 11 HBase commands to insert a new row into the table.

a. Table name: <your-namespace>:truck_event

b. Rowkey: 2000

c. Column family name: events

d. Columns: values

i. driverId: <your-login or UIS NetID>

ii. truckId: 999

iii. eventTime: 01:01.1

iv. eventType: <Pick one from Normal, Overspeed, and Lane Departure>

v. longitude: -84.58

vi. latitude: 27.03

vii. eventKey (This is a RowKey)

viii. CorrelationId: 1000

ix. driverName: <Your name>

x. routeld: 888

xi. routeName: UIS to St. Louis

xii. eventDate: <current date> e.g. 2030-01-01-01, year-month-day-hour (24)

Inserted each column of a row using put command. Likewise, inserted 11 columns to a row.

```
put 'tdend2:truck_event', '2000', 'events:driverId', 'tdend2'
put 'tdend2:truck_event', '2000', 'events:truckId', '999'
put 'tdend2:truck_event', '2000', 'events:eventTime', '01:01.1'
put 'tdend2:truck_event', '2000', 'events:eventType', 'Overspeed'
put 'tdend2:truck_event', '2000', 'events:longitude', '-84.58'
put 'tdend2:truck_event', '2000', 'events:latitude', '27.03'
put 'tdend2:truck_event', '2000', 'events:correlationId', '1000'
put 'tdend2:truck_event', '2000', 'events:driverName', 'Tejashri Dendi'
put 'tdend2:truck_event', '2000', 'events:routeld', '888'
put 'tdend2:truck_event', '2000', 'events:routeName', 'UIS to St. Louis'
put 'tdend2:truck_event', '2000', 'events:eventDate', '2024-09-27-01'
```

```
tdend2@node00:~
```

```
hbase(main):004:0> put 'tdend2:truck_event', '2000', 'events:driverId', 'tdend2'
```

```
Took 0.0084 seconds
```

```
hbase(main):005:0> put 'tdend2:truck_event', '2000', 'events:truckId', '999'
```

```
Took 0.0067 seconds
```

```
hbase(main):006:0> put 'tdend2:truck_event', '2000', 'events:eventTime', '01:01.1'
```

```
Took 0.0040 seconds
```

```
hbase(main):007:0> put 'tdend2:truck_event', '2000', 'events:eventType', 'Overspeed'
```

```
Took 0.0092 seconds
```

```
hbase(main):008:0> put 'tdend2:truck_event', '2000', 'events:longitude', '-84.58'
```

```
Took 0.0044 seconds
```

```
hbase(main):009:0> put 'tdend2:truck_event', '2000', 'events:latitude', '27.03'
```

```
Took 0.0037 seconds
```

```
hbase(main):010:0> put 'tdend2:truck_event', '2000', 'events:correlationId', '1000'
```

```
Took 0.0057 seconds
```

```
hbase(main):011:0> put 'tdend2:truck_event', '2000', 'events:driverName', 'Tejashri Dendi'
```

```
Took 0.0064 seconds
```

```
hbase(main):012:0> put 'tdend2:truck_event', '2000', 'events:routeId', '888'
```

```
Took 0.0063 seconds
```

```
hbase(main):013:0> put 'tdend2:truck_event', '2000', 'events:routeName', 'UIS to St. Louis'
```

```
Took 0.0041 seconds
```

```
hbase(main):014:0> put 'tdend2:truck_event', '2000', 'events:eventDate', '2024-09-27-01'
```

```
Took 0.0057 seconds
```

```
hbase(main):015:0>
```


2. Write and run an HBase command to retrieve the row only you just inserted.

14

a. Table name: <your-namespace>:truck_event

b. Rowkey: 2000

c. Column family name: events

d. Note: the row only. No other rows.

e. You may want to check the help page 'help <command>'

get 'tdend2:truck_event', '2000', 'events'

```
tdend2@node00:~  
hbase(main):015:0> get 'tdend2:truck_event', '2000', 'events'  
COLUMN                                CELL  
events:correlationId                  timestamp=1727474286479, value=1000  
events:driverId                       timestamp=1727474207096, value=tdend2  
events:driverName                     timestamp=1727474303286, value=Tejashri Dendi  
events:eventDate                      timestamp=1727474335546, value=2024-09-27-01  
events:eventTime                      timestamp=1727474239055, value=01:01.1  
events:eventType                      timestamp=1727474251258, value=Overspeed  
events:latitude                       timestamp=1727474274054, value=27.03  
events:longitude                      timestamp=1727474262124, value=-84.58  
events:routeId                       timestamp=1727474314644, value=888  
events:routeName                     timestamp=1727474324876, value=UIS to St. Louis  
events:truckId                       timestamp=1727474225074, value=999  
1 row(s)  
Took 0.0237 seconds  
hbase(main):016:0>
```

Used **get command** as it allows specific row retrieval as scan command gives the information of the entire table.

3. Write and run two HBase commands to update the value of the row you inserted/retrieved and show the changes.

a. Table name: <your-namespace>:truck_event

b. Rowkey: 2000

c. Column family name: events

d. Column name(qualifier): routeName

e. NEW value: St. Louis to UIS

f. Write and run your second command to show the changes

i. Expected changes: 'UIS to St. Louis' → 'St. Louis to UIS'

```
tdend2@node00:~  
hbase(main):016:0> put 'tdend2:truck_event', '2000', 'events:routeName', 'St. Louis to UIS'  
Took 0.0120 seconds  
hbase(main):017:0> get 'tdend2:truck_event', '2000', 'events'  
COLUMN                                CELL  
events:correlationId                  timestamp=1727474286479, value=1000  
events:driverId                       timestamp=1727474207096, value=tdend2  
events:driverName                     timestamp=1727474303286, value=Tejashri Dendi  
events:eventDate                      timestamp=1727474335546, value=2024-09-27-01  
events:eventTime                      timestamp=1727474239055, value=01:01.1  
events:eventType                      timestamp=1727474251258, value=Overspeed  
events:latitude                       timestamp=1727474274054, value=27.03  
events:longitude                      timestamp=1727474262124, value=-84.58  
events:routeId                       timestamp=1727474314644, value=888  
events:routeName                      timestamp=1727474637928, value=St. Louis to UIS  
events:truckId                       timestamp=1727474225074, value=999  
1 row(s)  
Took 0.0233 seconds  
hbase(main):018:0>
```

Updated the value of a column using

put command- `put 'tdend2:truck_event', '2000', 'events:routeName', 'St. Louis to UIS'` and used **get command** to confirm the changes done as it allows parameters to query the data. `get 'tdend2:truck_event', '2000', 'events'`

4. Write and run an HBase command to retrieve two columns of the row you inserted/retrieved

- a. Table name: <your-namespace>:truck_event
- b. Rowkey: 2000
- c. Column family name: events
- d. Two columns we want to see: driverName and routeName ONLY
- e. Note: Two columns (qualifiers) only. No other columns.
- f. You may want to check the help page 'help <command>'

`get 'tdend2:truck_event', '2000', 'events:driverName', 'events:routeName'`

```
tdend2@node00:~
hbase(main):019:0> get 'tdend2:truck_event', '2000', 'events:driverName', 'events:routeName'
COLUMN                                CELL
events:driverName                      timestamp=1727474303286, value=Tejashri Dendi
events:routeName                      timestamp=1727474637928, value=St. Louis to UIS
1 row(s)
Took 0.0187 seconds
hbase(main):020:0>
```

Used get command for the retrieval.

5. Write and run an HBase command in non-interactive mode

- a. Retrieve the meta-data of your table in the Linux shell.
 - b. Table name: <your-namespace>:truck_event
 - c. Use Both Linux commands and HBase shell commands
 - i. Linux command: echo
 - ii. HBase shell command: describe (retrieve meta-data of a table)
 - d. This code should run outside of the HBase shell.
 - i. Don't log in to the HBase shell.
 - ii. Should be run in a Linux shell
 - iii. e.g. [sslee777@node00 ~]\$ <your command here....>
 - e. This scripting is quite useful in some cases, e.g., when you want to run HBase commands outside the HBase shell.
 - i. Batch processing
 - ii. Administration
 - iii. Automation – Cron jobs
 - iv. Etc
- ```
echo "describe 'tdend2:truck_event'" | hbase shell -n
```

```

tdend2@node00:~
[tdend2@node00 ~]$ echo "describe 'tdend2:truck_event'" | hbase shell -n
Table tdend2:truck_event is ENABLED
tdend2:truck_event
COLUMN FAMILIES DESCRIPTION
{NAME => 'events', VERSIONS => '1', EVICT_BLOCKS_ON_CLOSE => 'false', NEW_VERSION_BEHAVIOR => 'false', TTL => 'FOREVER', MIN_VERSIONS => '0', REPLICATION_SCOPE => '0', BLOOMFILTER => 'ROW', CACHE_BLOCKS_ON_OPEN => 'false', COMPRESSION => 'NONE', BLOCKCACHE => 'true', BLOCKSIZE => '65536'}
1 row(s)
Took 0.5922 seconds

[tdend2@node00 ~]$

```

Echo command is used to display the results to the console whereas describe command gave the information of the meta data of table.

=====THE END=====