ASSIGNMENT

TASK1

a) What is NoSQL database?

Sol: NoSQL is a modified approach to handle the growing data in the web world which is not handled by relational database management systems (RDBMS).NoSQL databases do not rely on table,column etc only and use more flexible data models. NoSQL can mean “not SQL” or “not only SQL.” As RDBMS have increasingly failed to meet the performance, scalability, and flexibility needs that next-generation, data-intensive applications require, NoSQL databases have been adopted by mainstream enterprises. NoSQL is particularly useful for storing semi-structured and unstructured data, which is growing far more rapidly than structured data and does not fit the relational schemas of RDBMS. Common types of unstructured data include: user and session data; chat, messaging, and log data; time series data such as IoT and device data; and large objects such as video and images.

there are mainly 4 types of nosql databases available for use in enterprises:

i) wide column based e.g cassandra

ii) key-value based e.g hbase

iii) graph based e.g apache giraffe

iv) document based e.g MongoDB

b) How does data get stored in NoSQl database?

Sol: Data is stored in key-value pair mainly. The exact storing of data depends upon which NoSQL is being used.

Though the important function is that NoSQL can have column family feature which helps in storing data in multiple column inside those column families and improves the performance of database. It also gives an option for horizontal scalability.

c) What is a column family in HBase?

Sol: Column family in hbase is the segregation of column in NoSQL to improve the performance of NoSQL. Random r/w operation can be made easy. Every column family can have multiple column which need not be mentioned while creating the table. It also gives feature like versioning to hold to previous values of key with timestamp.

d) How many maximum number of columns can be added to HBase table?

Sol: There is no limit to the maximum no of column but there is only 3 max column family is recommended. The hbase also has feature of sparsing so it is not restricted to limited column.

e) Why columns are not defined at the time of table creation in HBase?

Sol: The column family is specified during the creation of the table but column qualifiers are mentioned on the fly/while loading the data. The column qualifiers make the model as the self describing data model because the qualifiers can be dynamically specified as ne rows are created and inserted into the table.

f) How does data get managed in HBase?

Sol: The main characteristics that make Hbase an excellent data management platform are fault tolerance, speed and usability. Fault tolerance is provided by automatic fail-over, automatically sharded and load balanced tables, strong consistency in row level operations and replication. Speed is provided by almost real time lookups, in memory caching and server side processing. Usability is provided by a flexible data model that allows many uses, a simple Java API and ability to export metrics.

The Hbase data model is different from the model provided by relational databases. Hbase is referred to by many terms like a key-value store, column oriented database and versioned map of maps which are correct. The easiest way of visualizing a Hbase data model is a table that has rows and tables.

This is the only similarity shared by Hbase model and the relational model. Data in Hbase is organized into tables. Any characters that are legal in file paths are used to name tables. Tables are further organized into rows that store data. Each row is identified by a **unique row key** which does not belong to any data type but is stored as a bytearray. **Column families** are further used to group data in rows. Column families define the physical structure of data so they are defined upfront and their modification is difficult. Each row in a table has same column families. Data in a column family is addressed using a column qualifier. It is not necessary to specify column qualifiers in advance and there is no consistency requirement between rows. No data types are specified for **column qualifiers**, as such they are just stored as bytearrays. A unique combination of row key, column family and column qualifier forms a cell. Data contained in a cell is referred to as cell value. There is no concept of data type when referring to cell values and they are stored as bytearrays. **Versioning** happens to cell values using a **timestamp** of when the cell was written.

g) What happens internally when new data gets inserted into HBase table?

Sol: Put command is used to insert data into Hbase. Each Hbase table is divided into regions, where each region will have the sorted row key.

Each region will have store that has column family assigned to each store within the region. Regions are assigned to region servers(storage nodes) fro storage. A master server(master node) is responsible for monitoring the region servers and splitting a table into regions and assigning regions to region servers.

Hbase uses the Apache Zookeeper open source system for services related managing the naming, distribution and synchronization of the hbase data on the distributed hbase server nodes, as well as coordination and replication servers. It uses HDFS for file services. Hbase is built on top of both Zookeeper and HDFS.

<u>TASK2</u>

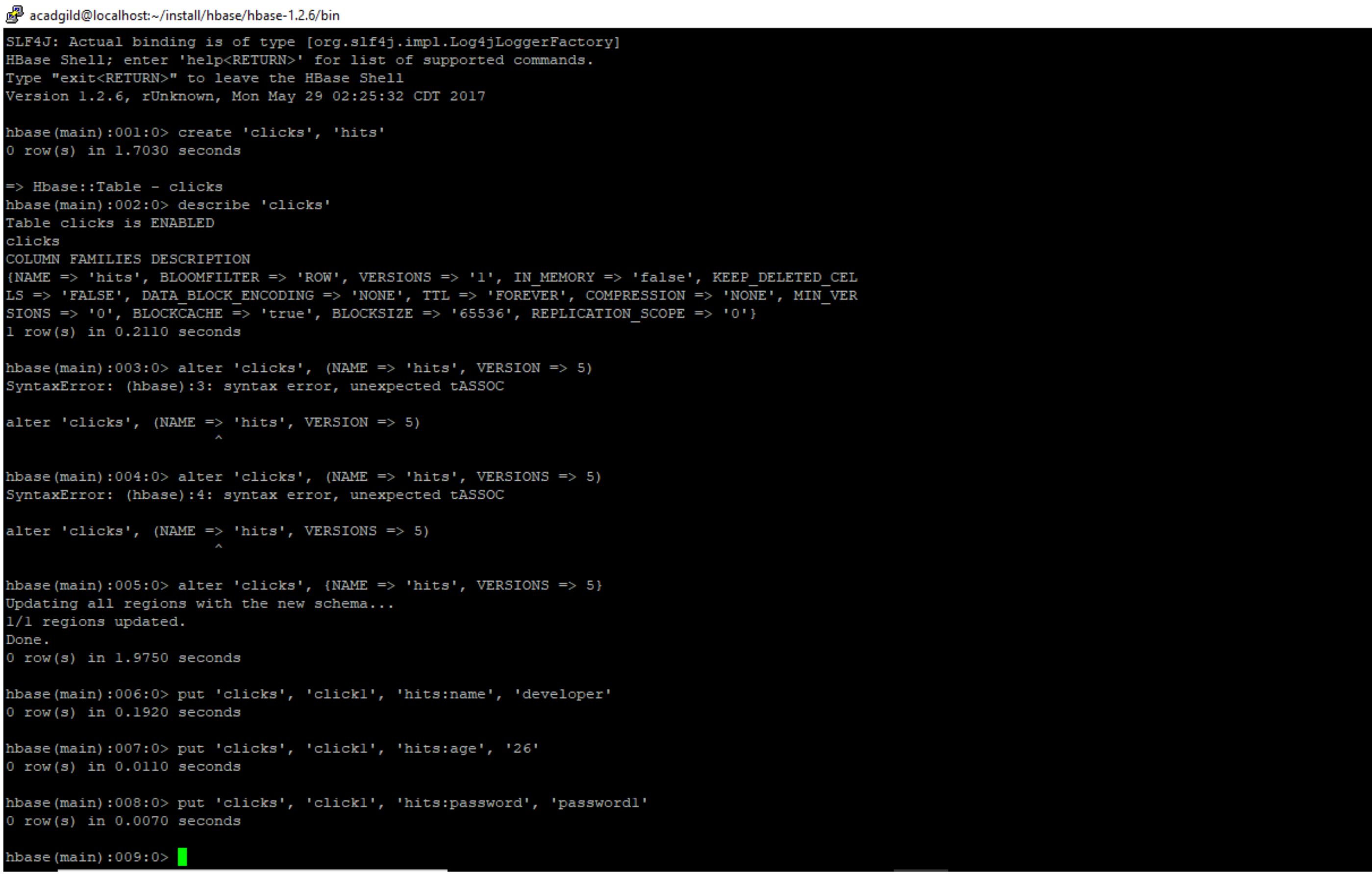
Command used:

a) Creating a table and storing the last 5 values in it.

i) Create ‘clicks’, ‘hits’

ii) alter ‘clicks’, {NAME => ‘hits’, VERSION => 5}

iii) describe ‘clicks’



acadgild@localhost:~/install/hbase/hbase-1.2.6/bin

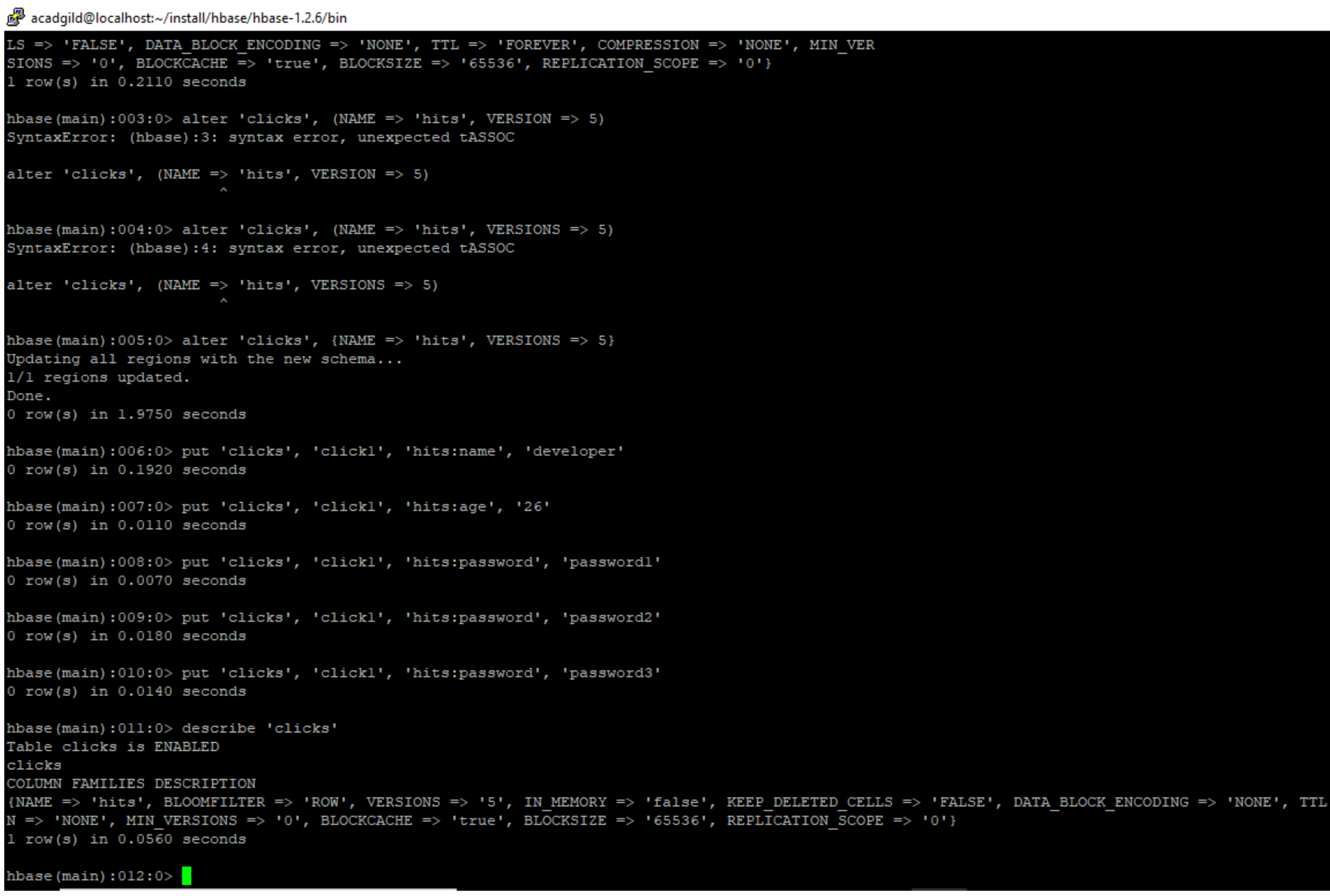
```
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
HBase Shell; enter 'help<RETURN>' for list of supported commands.
Type "exit<RETURN>" to leave the HBase Shell
Version 1.2.6, rUnknown, Mon May 29 02:25:32 CDT 2017

hbase(main):001:0> create 'clicks', 'hits'
0 row(s) in 1.7030 seconds

=> Hbase::Table - clicks
hbase(main):002:0> describe 'clicks'
Table clicks is ENABLED
clicks
COLUMN FAMILIES DESCRIPTION
{NAME => 'hits', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CEL
LS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VER
SIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.2110 seconds

hbase(main):003:0> alter 'clicks', (NAME => 'hits', VERSION => 5)
SyntaxError: (hbase):3: syntax error, unexpected tASSOC

alter 'clicks', (NAME => 'hits', VERSION => 5)
                      ^

hbase(main):004:0> alter 'clicks', (NAME => 'hits', VERSIONS => 5)
SyntaxError: (hbase):4: syntax error, unexpected tASSOC

alter 'clicks', (NAME => 'hits', VERSIONS => 5)
                      ^

hbase(main):005:0> alter 'clicks', {NAME => 'hits', VERSIONS => 5}
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9750 seconds

hbase(main):006:0> put 'clicks', 'click1', 'hits:name', 'developer'
0 row(s) in 0.1920 seconds

hbase(main):007:0> put 'clicks', 'click1', 'hits:age', '26'
0 row(s) in 0.0110 seconds

hbase(main):008:0> put 'clicks', 'click1', 'hits:password', 'password1'
0 row(s) in 0.0070 seconds

hbase(main):009:0>
```

b) **Add few records:**

   a. **put ‘clicks’, ‘click1’, ‘hits:name’, ‘developer’**
   b. **put ‘clicks’, ‘click1’, ‘hits:age’, ‘26’**
   c. **put ‘clicks’, ‘click1’, ‘hits:password’, ‘password1’**

c) **update the record:**

   a. **put ‘clicks’, ‘click1’, ‘hits: password’, ‘password2’**
   b. **put ‘clicks’, ‘click1’, ‘hits: password’, ‘password3’**
   c. **put ‘clicks’, ‘click1’, ‘hits: password’, ‘password4’**

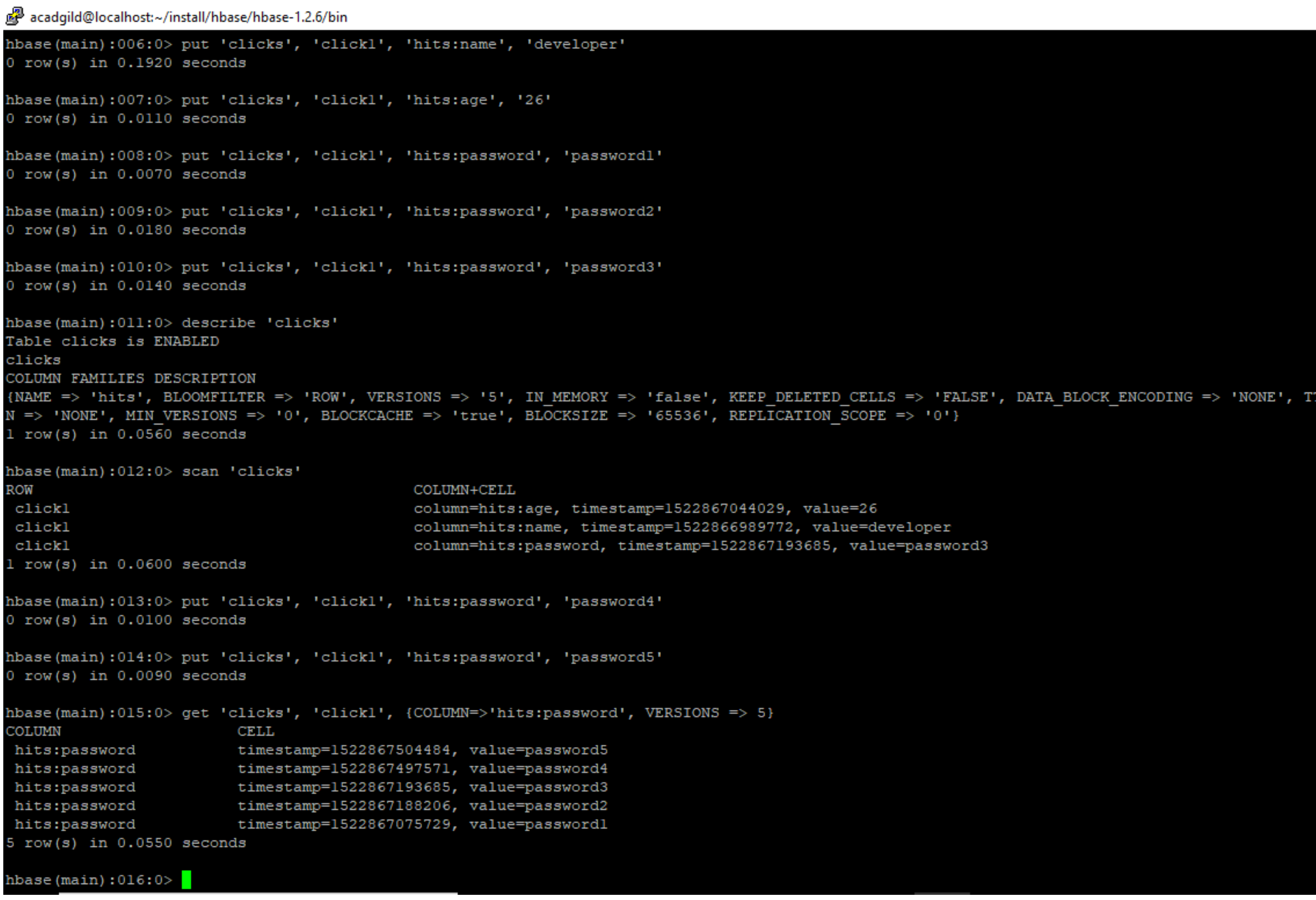


acadgild@localhost:~/install/hbase/hbase-1.2.6/bin

```
LS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VER
SIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.2110 seconds

hbase(main):003:0> alter 'clicks', (NAME => 'hits', VERSION => 5)
SyntaxError: (hbase):3: syntax error, unexpected tASSOC

alter 'clicks', (NAME => 'hits', VERSION => 5)
                       ^

hbase(main):004:0> alter 'clicks', (NAME => 'hits', VERSIONS => 5)
SyntaxError: (hbase):4: syntax error, unexpected tASSOC

alter 'clicks', (NAME => 'hits', VERSIONS => 5)
                       ^

hbase(main):005:0> alter 'clicks', {NAME => 'hits', VERSIONS => 5}
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9750 seconds

hbase(main):006:0> put 'clicks', 'click1', 'hits:name', 'developer'
0 row(s) in 0.1920 seconds

hbase(main):007:0> put 'clicks', 'click1', 'hits:age', '26'
0 row(s) in 0.0110 seconds

hbase(main):008:0> put 'clicks', 'click1', 'hits:password', 'password1'
0 row(s) in 0.0070 seconds

hbase(main):009:0> put 'clicks', 'click1', 'hits:password', 'password2'
0 row(s) in 0.0180 seconds

hbase(main):010:0> put 'clicks', 'click1', 'hits:password', 'password3'
0 row(s) in 0.0140 seconds

hbase(main):011:0> describe 'clicks'
Table clicks is ENABLED
clicks
COLUMN FAMILIES DESCRIPTION
{NAME => 'hits', BLOOMFILTER => 'ROW', VERSIONS => '5', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL
N => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.0560 seconds

hbase(main):012:0>
```

d) **scan it to see the previous values:**

   a. **get ‘clicks’, ‘click1’, {COLUMN => ‘hits:password’, VERSION => 5}**


acadgild@localhost:~/install/hbase/hbase-1.2.6/bin

```
hbase(main):006:0> put 'clicks', 'click1', 'hits:name', 'developer'
0 row(s) in 0.1920 seconds

hbase(main):007:0> put 'clicks', 'click1', 'hits:age', '26'
0 row(s) in 0.0110 seconds

hbase(main):008:0> put 'clicks', 'click1', 'hits:password', 'password1'
0 row(s) in 0.0070 seconds

hbase(main):009:0> put 'clicks', 'click1', 'hits:password', 'password2'
0 row(s) in 0.0180 seconds

hbase(main):010:0> put 'clicks', 'click1', 'hits:password', 'password3'
0 row(s) in 0.0140 seconds

hbase(main):011:0> describe 'clicks'
Table clicks is ENABLED
clicks
COLUMN FAMILIES DESCRIPTION
{NAME => 'hits', BLOOMFILTER => 'ROW', VERSIONS => '5', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', T
N => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.0560 seconds

hbase(main):012:0> scan 'clicks'
ROW                                        COLUMN+CELL
 click1                                    column=hits:age, timestamp=1522867044029, value=26
 click1                                    column=hits:name, timestamp=1522866989772, value=developer
 click1                                    column=hits:password, timestamp=1522867193685, value=password3
1 row(s) in 0.0600 seconds

hbase(main):013:0> put 'clicks', 'click1', 'hits:password', 'password4'
0 row(s) in 0.0100 seconds

hbase(main):014:0> put 'clicks', 'click1', 'hits:password', 'password5'
0 row(s) in 0.0090 seconds

hbase(main):015:0> get 'clicks', 'click1', {COLUMN=>'hits:password', VERSIONS => 5}
COLUMN                   CELL
 hits:password           timestamp=1522867504484, value=password5
 hits:password           timestamp=1522867497571, value=password4
 hits:password           timestamp=1522867193685, value=password3
 hits:password           timestamp=1522867188206, value=password2
 hits:password           timestamp=1522867075729, value=password1
5 row(s) in 0.0550 seconds

hbase(main):016:0>
```