Session 10:

HBASE BASICS

Assignment 1

PROBLEM STATEMENT –

TASK 1 –

Answer in your own words with example.

1. **What is NoSQL data base?**

A **NoSQL** database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases. NoSQL is an approach to database design that can accommodate a wide variety of data models, including **key-value, document, columnar and graph formats**.

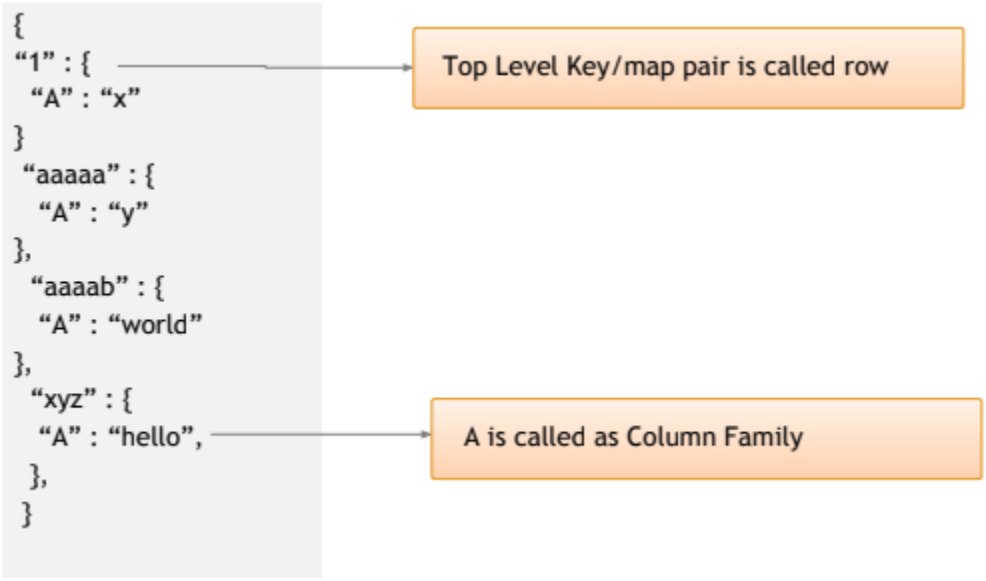
2. **How does data get stored in NoSQl database?**

Graph stores are used to store information about networks of data, such as social connections. Graph stores include Neo4J and Giraph. Key-value stores are the simplest NoSQL databases. Every item in the database is stored as an attribute name together with its value.

3. **What is a column family in HBase?**

Columns in Apache HBase are grouped into column families. All column members of a column family have the same prefix. For example, the columns **sports:football** and **sports:cricket** are both members of the courses column family. The colon character (:) delimits the column family from the column family prefix must be composed of printable characters. The qualifying tail, the column family qualifier, can be made of any arbitrary bytes.

Physically, all column family members are stored together on the filesystem. Because tuning and storage specifications are done at the column family level.



4. **How many maximum numbers of columns can be added to HBase table?**

There is no limit to number of columns in HBase, we can have more than 1 million columns but usually three column families are recommended (not more than three).

5. **Why columns are not defined at the time of table creation in HBase?**

An HBase table is made of column families which are the logical and physical grouping of columns. The columns in one family are stored separately from the columns in another family.

A single column family contains one or more columns, column families must be defined at table creation time but columns can be added dynamically after table creation (if an insert statement states a column that does not exist for a column family it will create it).

6. **How does data get managed in HBase?**

HBase's Region server has the following components which manages the data in the HBASE.

•**WAL**: Write Ahead Log is a file on the distributed file system. The WAL is used to store new data that hasn't yet been persisted to permanent storage; it is used for recovery in the case of failure.

•**BlockCache**: Is the read cache. It stores frequently read data in memory. Least Recently Used data is evicted when full.

•**MemStore**: Is the write cache. It stores new data which has not yet been written to disk. It is sorted before writing to disk. There is one MemStore per column family per region.

•**Hfiles** store the rows as sorted KeyValues on disk.

7. **What happens internally when new data gets inserted into HBase table?**

When the client reads or writes to HBase,

•The client gets the Region server that hosts the META table from ZooKeeper.

•The client will query the .META. Server to get the region server corresponding to the row key it wants to access. The client caches this information along with the META table location.

•It will get the Row from the corresponding Region Server.

For future reads, the client uses the cache to retrieve the META location and previously read row keys. Over time, it does not need to query the META table, unless there is a miss because a region has moved; then it will re-query and update the cache.

TASK 2 –

1. Create an HBase table named 'clicks' with a column family 'hits' such that it should be

able to store last 5 values of qualifiers inside 'hits' column family.

2. Add few records in the table and update some of them. Use IP Address as row-key.

Scan the table to view if all the previous versions are getting displayed.

SOLUTION –

1. **create 'clicks',NAME=>'hits',VERSIONS=>5**

```
hbase(main):006:0> create 'clicks',NAME=>'hits',VERSIONS=>5
0 row(s) in 2.5360 seconds

=> Hbase::Table - clicks
```

2. **Here we are using IP address 192.168.1.7' as row key.**

put 'clicks',' 192.168.1.7','hits:No_Of_Times','12'

scan 'clicks'

```
hbase(main):008:0> scan 'clicks'
ROW                                   COLUMN+CELL
  192.168.1.7                         column=hits:No_Of_Times, timestamp=1538270311476, value=12
1 row(s) in 0.1360 seconds

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

Updating the rows by adding the below values,

put 'clicks','192.168.1.7','hits:No_Of_Times','27'

put 'clicks','192.168.1.7','hits:No_Of_Times','68'

put 'clicks','192.168.1.7','hits:No_Of_Times','15'

put 'clicks','192.168.1.7','hits:No_Of_Times','97'

We are going to see the last 5 qualifiers inside hits column family,

scan 'clicks',{COLUMN=>'hits:No_Of_Times',VERSIONS=>5}

```
hbase(main):014:0> scan 'clicks',{COLUMN=>'hits:No_Of_Times',VERSIONS=>5}
ROW                                   COLUMN+CELL
  192.168.1.7                         column=hits:No_Of_Times, timestamp=1538270311476, value=12
 192.168.1.7                          column=hits:No_Of_Times, timestamp=1538270480173, value=97
 192.168.1.7                          column=hits:No_Of_Times, timestamp=1538270478248, value=15
 192.168.1.7                          column=hits:No_Of_Times, timestamp=1538270478050, value=68
 192.168.1.7                          column=hits:No_Of_Times, timestamp=1538270477746, value=27
2 row(s) in 0.3210 seconds

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

We can see the IP address and how much number of times it was hit in a regular interval manner.