Task 1

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. NoSQL, which stand for "not only SQL," is an alternative to traditional relational databases in which data is placed in tables and data schema is carefully designed before the database is built. NoSQL databases are especially useful for working with large sets of distributed data.

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 single item in the database is stored as an attribute name (or 'key'), 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 courses:history and courses:math 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. Column families must be declared up front at schema definition time whereas columns do not need to be defined at schema time but can be conjured on the fly while the table is up and running.

Physically, all column family members are stored together on the filesystem. Because tunings and storage specifications are done at the column family level, it is advised that all column family members have the same general access pattern and size characteristics.

Row Key	Column Family: {Column Qualifier:Version:Value}
00001	***CustomerName***: {'FN': 1383859182496:'John', 'LN': 1383859182858:'Smith', 'MN': 1383859183001:'Timothy', 'MN': 1383859182915:'T'} ***ContactInfo***: {'EA': 1383859183030:'John.Smith@xyz.com', 'SA': 1383859183073:'1 Hadoop Lane, NY 11111'}
00002	***CustomerName***: {'FN': 1383859183103:'Jane', 'LN': 1383859183163:'Doe', ***ContactInfo***: { 'SA': 1383859185577:'7 HBase Ave, CA 22222'}

Note:-Column families are grouped together on disk, so grouping data with similar access patterns reduces overall disk access and increases performance.

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

There is no hard 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?

The data in the HBASE is managed in such a way that,

•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.

All the above are sub components of Region server which manages the data in the HBASE.

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

This is what happens the first time a 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.

Table name: ***clicks***

Column Family: ***hits***

Versions: 5

Command format:

Create <'table name'>,<'column family=>value'>,VERSIONS=>5

hbase(main):007:0* create 'clicks',NAME=>'hits',VERSIONS=>5

```
hbase(main):006:0>
hbase(main):007:0* create 'clicks',NAME=>'hits',VERSIONS=>5
0 row(s) in 1.0230 seconds
```

```
hbase(main):010:0* list
TABLE
clicks
customer
studentAcad
3 row(s) in 0.0860 seconds
```

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.

Here we are using ip address ***'27.59.2.31'*** as a row key,

put 'clicks','27.59.2.31','hits:No_Of_Times','12'

scan 'clicks'

```
hbase(main):014:0* put 'clicks','27.59.2.31','hits:No_Of_Times','12'
0 row(s) in 0.2190 seconds

hbase(main):015:0> scan 'clicks'
ROW                                   COLUMN+CELL
 27.59.2.31                           column=hits:No_Of_Times, timestamp=1511876988874, value=12
1 row(s) in 0.1310 seconds
```

Update the row by adding some values,

put 'clicks','27.59.2.31','hits:No_Of_Times','32'

put 'clicks','27.59.2.31','hits:No_Of_Times','8'

put 'clicks','27.59.2.31','hits:No_Of_Times','44'

put 'clicks','27.59.2.31','hits:No_Of_Times','99'

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

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

```
hbase(main):023:0> scan 'clicks',{COLUMN=>'hits:No_Of_Times',VERSIONS=>5}
ROW                                   COLUMN+CELL
 27.59.2.31                           column=hits:No_Of_Times, timestamp=1511877052223, value=99
 27.59.2.31                           column=hits:No_Of_Times, timestamp=1511877050801, value=44
 27.59.2.31                           column=hits:No_Of_Times, timestamp=1511877050655, value=8
 27.59.2.31                           column=hits:No_Of_Times, timestamp=1511877050455, value=32
 27.59.2.31                           column=hits:No_Of_Times, timestamp=1511876988874, value=12
1 row(s) in 0.1910 seconds
```

We can see the Ip address that how many number of times it was hits in a regular interval manner.