**Assignment 9.1**

Ques -1) What is NoSQL data base?

A **NoSQL** (originally referring to "non SQL" or "non relational") **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 databases** are increasingly used in big **data** and real-time web applications.

Example of NOSQL is HBASE ,mongoDB

Question 2How does data get stored in NoSQl database

There are various NoSQL Databases. Each one uses a different method to store data. Some might use column store, some document, some graph, etc., Each database has its own unique characteristics.  HBase is a columnar database, so all data is stored into tables with rows and columns similar to relational database management systems (RDBMSs). and these Tables are stored in Regions

Question 3) What is a column family in HBase?

A **HBase** table is comprised of one or more **column families**, each of which is stored in a separate set of regionfiles sharing a common key. To express it in terms of an RDBMS, a **column family** is roughly analogous to a RDBMS table with the rowkey as a clustered primary key index

e.g.

create ‘Customer’ ,’Address’,’Order’

here customer is the name of table while Address and order are column families. If we see it in terms of RDMS Address and order are tables in RDBMS with its ownschema like in address we have columns like state and city and in order we have column like order no amount etc.

Question 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).

Question 5) Why columns are not defined at the time of table creation in HBase

HBase data model consists of tables containing rows. Data is organized into column families grouping columns in each row

Ques 6) How does data get managed in HBase

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.

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

The data is stored in HFiles, which are ordered immutable key/value maps. Internally, the HFiles are sequences of blocks with a block index stored at the end. The block index is loaded when the HFile is opened and kept in memory.  The default block size is 64 KB but it can be changed since it is configurable. HBase API can be used to access specific values and also scan ranges of values given a start and end key.

Since every HFile has a block index, lookups can be performed with a single disk seek. First, HBase does a binary search in the in-memory block index to find a block containing the given key and then the block is read from disk.

When data is updated it is first written to a commit log, called a write-ahead log (WAL) and then it is stored in the in-memory memstore.

When the data in memory exceeds a given maximum value, it is flushed as an HFile to disk and after that the commit logs are discarded up to the last unflushed modification. The system can continue to serve readers and writers without blocking them while it is flushing the memstore to disk. This is done by rolling the memstore in memory where the new empty one is taking the updates and the old full one is transferred into an HFile. At the same time, no sorting or other special processing has to be performed since the data in the memstores is already sorted by keys matching what HFiles represent on disk.

The write-ahead log (WAL) is used for recovery purposes only. Since flushing memstores to disk causes creation of HFiles, HBase has a housekeeping job that merges the HFiles into larger ones using compaction. Various compaction algorithms are supported.