Assignment 3.7

Problem Statement

1. What is NoSQL data base?

2. How does data get stored in NoSQl database?

3. What is a column family in HBase?

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

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

6. How does data get managed in HBase?

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

**WHAT IS NOSQL?**

NoSQL is an approach to databases that represents a shift away from traditional relational database management systems (RDBMS). To define NoSQL, it is helpful to start by describing SQL, which is a query language used by RDBMS. Relational databases rely on tables, columns, rows, or schemas to organize and retrieve data. In contrast, NoSQL databases do not rely on these structures 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 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.

**How 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.

**In the in-memory databases** like Redis/CouchBase/Tarantool/Aerospike everything is stored in RAM in balanced trees like RB-Tree or in hash tables. All the writes are applied on both RAM and disk, but on disk it goes in an append-only way. A file append can be done as fast as 100Mbytes per second on a normal magnetic disk. If a record size is, say, 1K, then the data will be written at 100krps.

**In the on-disk NoSQL databases and db-engines** like Cassandra/HBase/RocksDB/LevelDB/Sophia the main idea is that you have a snapshot file and a write ahead log (WAL) file. Snapshot contains already prepared data in a form of B-Tree with upper levels of that tree being permanently in RAM, that can be accesses for reading by doing only one disk seek. A WAL contains all the new changes on top of a current snapshot. A snapshot file is being totally rebuilt on a regular basis using current snapshot and a WAL. All the writes are done nearly as fast as with in-memory databases. "Nearly" because disk is partially busy by doing regular snapshot converting that was described earlier. Reads are significantly slower than that are in in-memory databases, because they take at least one disk seek, but good news is that they can be cached in optimized in-memory structures like RB-Trees/hash tables.

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

Column families are the base storage mechanism 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.    A HBase table would then be a view which does a full outer join on a set of RDBMS tables which all share the same primary key (thus having a 1:1 relationship).    In this analogy, HBase region files map to pages in an RDBMS.

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 . 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 an 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.

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

There is no limit on number of column families in HBase, in theory. In reality, there are several factors which can limit useable number of column families in HBase:

HBase Admin web UI usability. It will be very hard to show even 100s of column families in a table configuration page.

HDFS practical limit of maximum number of files. Say, 100m. If your table has N regions, M column families you will need NxM directories to support this configuration. Every region/column family, in turn, can contain up to K store files (depends on write load and many other configuration options). With very modest N = 100 and K = 10 we can say practical limit of maximum number of column families is less than 100K. Usually, much less than 100K.

Each column family has its own directory in HDFS and set of store files and, from performance point of view, the fewer directories (column families) you have the better performance for scan operations you get.