**ASSIGNMENT-9.1**

1. What is NoSQL database?

A **NoSQL** (originally referring to "non SQL" or "non relational")[database](https://en.wikipedia.org/wiki/Database) provides a mechanism for [storage](https://en.wikipedia.org/wiki/Computer_data_storage) and [retrieval](https://en.wikipedia.org/wiki/Data_retrieval) of data that is modeled in means other than the tabular relations used in [relational databases](https://en.wikipedia.org/wiki/Relational_database). NoSQL databases are increasingly used in [big data](https://en.wikipedia.org/wiki/Big_data) and [real-time web](https://en.wikipedia.org/wiki/Real-time_web)applications. NoSQL systems are also sometimes called "Not only SQL" to emphasize that they may support [SQL](https://en.wikipedia.org/wiki/SQL)-like query languages.

The data structures used by NoSQL databases (e.g. key-value, wide column, graph, or document) are different from those used by default in relational databases, making some operations faster in NoSQL. The particular suitability of a given NoSQL database depends on the problem it must solve. Sometimes the data structures used by NoSQL databases are also viewed as "more flexible" than relational database tables.

Motivations for this approach include: simplicity of design, simpler ["horizontal" scaling](https://en.wikipedia.org/wiki/Horizontal_scaling#Horizontal_and_vertical_scaling) to [clusters](https://en.wikipedia.org/wiki/Cluster_computing) of machines (which is a problem for relational databases and finer control over availability. 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.

1. 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.

A mongo database is broken up into a series of BSON files on disk, with increasing size up to 2GB. BSON is its own format, built specifically forMongoDB. MongoDB stores the data on the disk as BSON in your data path directory, which is usually **/data/db**.

1. What is a column family in HBase?

In the HBase data model columns are grouped into column families, which must be defined up front during table creation. Column families are stored together on disk, which is why HBase is referred to as a column-oriented data store.

Logical View of Customer Contact Information table:

|  |  |
| --- | --- |
| **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’} |

The table shows two column families: CustomerName and ContactInfo. When creating a table in HBase, the developer or administrator is required to define one or more column families using printable characters.

Generally, column families remain fixed throughout the lifetime of an HBase table but new column families can be added by using administrative commands.

1. 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.

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

Column families must be declared up front at schema definition time where as columns do not need to be defined at schema time but can be conjured on the fly while the table is up and running.

1. How does data get managed in HBase?

Hbase is natively supported on Hadoop. 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.

Hbase can run standalone on the local file system but this set up does not guarantee durability. Edits will be lost when daemons are not cleanly started and stopped. Such a set up is not suitable in a production environment but it provides a way of exploring how the database functions.

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

There is a special HBase Catalog table called the META table, which holds the location of the regions in the cluster. ZooKeeper stores the location of the META table.

This is what happens the first time a client reads or writes to HBase:

1. The client gets the Region server that hosts the META table from ZooKeeper.
2. 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.
3. 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.

