**1.What is NoSQL Database ?**

**Sol:** NoSQL means Not Only SQL,describe an approach to database design that implements a key-value store, document store, column store or graph format for data.

NoSQL contrasts to databases that adhere to SQL's relational methods, where data are placed in tables and data schema  are carefully designed before the database is built.

NoSQL databases are increasingly used in **[big data](https://en.wikipedia.org/wiki/Big_data" \o "Big data)** and**[real-time web](https://en.wikipedia.org/wiki/Real-time_web" \o "Real-time web)applications**.

NoSQL Databases are preferred for their simplicity in design and horizontal scaling of cluster nodes to accommodate more capacity and faster execution at comparatively **cheaper** cost.

NoSQL databases especially target **large set**s of distributed data.

NoSQL databases have become the first alternative to relational databases, with **scalability,** **availability**, and **fault tolerance** being key deciding factors.

 Large-scale web organizations such as Google and Amazon used NoSQL databases to focus on narrow operational goals and employ relational databases as adjuncts where high-grade **data consistency**is necessary.

**2.How Does data get stored in NoSQL Database ?**

**Sol:** Data can be stored in NoSQL Database in one of the following ways :

* **Document databases**pair each key with a complex data structure known as a document. Documents can contain many different key-value pairs, or key-array pairs, or even nested documents. Eg : MongoDB,CouchDB
* **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. Examples of key-value stores are Riak and Berkeley DB. Some key-value stores, such as Redis, allow each value to have a type, such as 'integer', which adds functionality.Eg: Voldemort, Redis, Scalaris.
* **Wide-column stores** such as Cassandra and HBase are optimized for queries over large data sets, and store columns of data together, instead of rows.

**3.What is a Column family in Hbase?**

**Sol:** 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 qualifier.

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 **file system**. Because t**unings 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.

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

**Sol:**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.

Depending on your data access patterns, you should consider **wide table**

( more rows with less columns) vs **tall table** (more columns with less rows ) **layout**.

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

**Sol:**HBase tables are created with column families which generally is not changed. The column keys are specified after the table is up and running.

Operations such as tunings and storage specifications are done at the column family

level and are hence stored in separate HFiles.

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

**Sol:** 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 byte array. 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 **byte arrays**. 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 byte arrays. **Versioning** happens to cell values using a time stamp of when the cell was written.

Tables in Hbase have several properties that need to be understood for one to come up with an effective data model. Indexing and sorting only happens on the row key. The concept of data types is absent and everything is stored as byte array. Only row level **atomicity** is enforced so multi row transactions are not supported.

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

**Sol:**Within HBase, as row key is the only index and when you try to use the same row key and try to insert the data in the column which is already present, it will try to override the data. You can verify it with the time-stamp associated with it. If you specify a new column, the data won't be overridden.