

NoSQL is an approach to database design that can accomodate a wide variety of data models, including ***key-value, document, columnar and graph format***s. NoSQL, which stand for "not only SQL," is an alternative to traditional relational databases in which data is placed in tables and data schemais carefully designed before the database is built. NoSQL databases are especially useful for working with large sets of distributed data.



Before putting data in NO-SQL for data modeling, for example, a programmer does not need consider the role that each table represents and also what the inside columns represent, as well as how to join each by determining a common element/column that will be used as a primary key.This saves lot of time because there are so many kinds of data sources.

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.

Foe example MongoDB is an document store, where data is stored as ***Key: Value*** pairs in JSON format.

1. {
2. name: "sid",
3. phone: 1234567890,
4. address:
5. {
6. street: "1234 Some\_XYZ Pkwy" ,
7. Apt: 1001,
8. City: "Richardson",
9. State: "Texas"
10. }
11. }



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.

*column families*, are the logical and physical grouping of columns. There are *column qualifiers* inside of a column family, which are the columns. Column families contain columns with time stamped versions. Columns only exist when they are inserted, which makes HBase a sparse database. All column members of the same column family have the same column family prefix. Each column value is identified by a key. The *row key* is the implicit primary key. Rows are sorted by the *row key*. An HBase column can be specified by using the following format:

hbase-family:hbase-column-name



Two or three column families can be added.

There is a limit to the number of column families in HBase. There is one MemStore(Its a write cache which stores new data before writing it into Hfiles) per Column Family, when one is full, they all flush.

The more you add column families there will be more MemStore created and Memstore flush will be more frequent. It will degrade the performance.



***Columns*** are arbitrary names (or labels) assigned by the application.

Columns families are stored on Hfile. Hfile are stored on Memstore. Memstore is flushed every time when it is full which degrades the performance of hbase.

This is reason why columns are defined on fly trying to keep it limited to improve the performance.



Puts and Deletes are collected into an in-memory structure called the MemStore. Before the MemStore is update the changes are written to a Write Ahead Log (WAL) to enable recovery in case a server crashes.  
When it reaches a certain size the MemStore is flushed to disk into StoreFile.  
  
Periodically StoreFiles are compacted into fewer StoreFiles.  
  
For reading and writing HBase employs Log Structured Mergetrees, which is just a fancy way of saying that reading and compacting in HBase is performing a merge sort (a scan looks at the heads of all StoreFiles and the Memstore and picks the smallest element first, in case of a Scan it is returned to the client, in case of a compacted it is written to the new StoreFile).



Internal process after inserting data is based on schema definition.

The decision in made based on parameter provided such as versions, block size.

Columns families are stored on Hfile. Hfile are stored on Memstore. Memstore is flushed every time when it is full

HBase sorts the versions of a cell from newest to oldest, by sorting the timestamps lexicographically. When a version needs to be deleted because a threshold has been reached, HBase always chooses the "oldest" version, even if it is in fact the most recent version to be inserted. Keep this in mind when designing your timestamps. Consider using the default generated timestamps and storing other version-specific data elsewhere in the row, such as in the row key. If MIN\_VERSIONS and TTL conflict, MIN\_VERSIONS takes precedence.