1. What is NoSQL data base?

Non-relational and largely distributed database system that enables rapid, ad-hoc organization and analysis of extremely high-volume, disparate data types.

A very flexible and schema-less data model, horizontal scalability, distributed architectures, and the use of languages and interfaces that are “not only” SQL typically characterize this technology.

1. How does data get stored in NoSQl database?

Types of NoSQL DB:

* **Graph database** – Based on [graph theory](https://web.archive.org/web/20140922065727/http:/en.wikipedia.org/wiki/Graph_theory), these databases are designed for data whose relations are well represented as a graph and has elements which are interconnected, with an undetermined number of relations between them. Examples include: Neo4j and Titan.
* **Key-Value store** – we start with this type of database because these are some of the least complex NoSQL options. These databases are designed for storing data in a schema-less way. In a key-value store, all of the data within consists of an indexed key and a value, hence the name. Examples of this type of database include:[Cassandra](https://web.archive.org/web/20140922065727/http:/www.datastax.com/resources/tutorials/cassandra-overview), DyanmoDB
* **Column store** – (also known as wide-column stores) instead of storing data in rows, these databases are designed for storing data tables as sections of columns of data, rather than as rows of data. Examples include: HBase, BigTable
* **Document databases** - also called document stores, store semi-structured data and descriptions of that data in document format.

Document databases are used for content management and mobile application data handling. Couchbase Server, [CouchDB](http://searchdatamanagement.techtarget.com/definition/CouchDB), DocumentDB, MarkLogic and [MongoDB](http://searchdatamanagement.techtarget.com/definition/MongoDB) are examples of document databases.

1. What is a column family in HBase?

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.

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 and running.

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

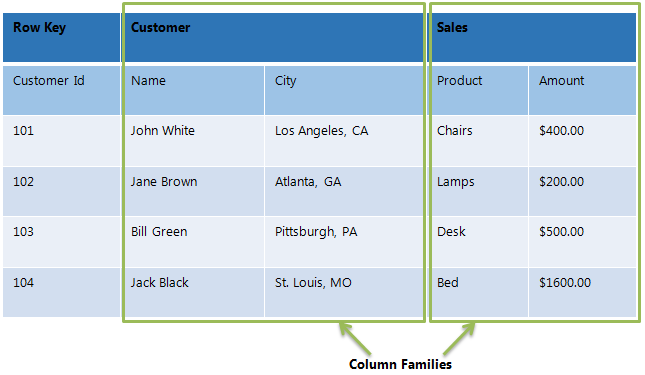
There is no limit to columns but there is limit to column families.

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

?

1. How does data get managed in HBase?

The Data Model in HBase is designed to accommodate semi-structured data that could vary in field size, data type and columns. Additionally, the layout of the data model makes it easier to partition the data and distribute it across the cluster. The Data Model in HBase is made of different logical components such as Tables, Rows, Column Families, Columns, Cells and Versions.



Hbase data model

**Tables** – The HBase Tables are more like logical collection of rows stored in separate partitions called Regions. every Region is then served by exactly one Region Server.

Once specific limit of row key is reached, regions are split.

**Rows** – A row is one instance of data in a table and is identified by a rowkey. Rowkeys are unique in a Table and are always treated as a byte[].

**Column Families** – Data in a row are grouped together as Column Families. Each Column Family has one more Columns and these Columns in a family are stored together in a low level storage file known as HFile. Column Families form the basic unit of physical storage to which certain HBase features like compression are applied. Hence it’s important that proper care be taken when designing Column Families in table.

**Columns** – A Column Family is made of one or more columns. A Column is identified by a Column Qualifier that consists of the Column Family name concatenated with the Column name using a colon – example: columnfamily:columnname. There can be multiple Columns within a Column Family and Rows within a table can have varied number of Columns.

**Cell** – A Cell stores data and is essentially a unique combination of rowkey, Column Family and the Column (Column Qualifier). The data stored in a Cell is called its value and the data type is always treated as byte[].

**Version** – The data stored in a cell is versioned and versions of data are identified by the timestamp. The number of versions of data retained in a column family is configurable and this value by default is 3.

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

When the client issues a Put request, the first step is to write the data to the write- ahead log, the WAL:

- Edits are appended to the end of the WAL file that is stored on disk.

the data is written to the WAL, it is placed in the MemStore. Then, the put request acknowledgement returns to the client.

The MemStore stores updates in memory as sorted KeyValues, the same as it would be stored in an HFile. There is one MemStore per column family. The updates are sorted per column family.

When the MemStore accumulates enough data, the entire sorted set is written to a new HFile in HDFS. HBase uses multiple HFiles per column family, which contain the actual cells, or KeyValue instances. These files are created over time as KeyValue edits sorted in the MemStores are flushed as files to disk.

Note that this is one reason why there is a limit to the number of column families in HBase. There is one MemStore per CF; when one is full, they all flush. It also saves the last written sequence number so the system knows what was persisted so far.

While reading the data:

First, the scanner looks for the Row cells in the Block cache - the read cache. Recently Read Key Values are cached here, and Least Recently Used are evicted when memory is needed.

Next, the scanner looks in the MemStore, the write cache in memory containing the most recent writes.

If the scanner does not find all of the row cells in the MemStore and Block Cache, then HBase will use the Block Cache indexes and bloom filters to load HFiles into memory, which may contain the target row cells.

