1. **What is NoSQL data base?**

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

**TYPES OF NOSQL DATABASES**

Several different varieties of NoSQL databases have been created to support specific needs and use cases. These fall into four main categories:

**Key-value data stores**

**Document stores**

**Wide-column stores**

**Graph stores**

1. **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.

**Key-value data stores:** Key-value NoSQL databases emphasize simplicity and are very useful in accelerating an application to support high-speed read and write processing of non-transactional data. Stored values can be any type of binary object (text, video, JSON document, etc.) and are accessed via a key. The application has complete control over what is stored in the value, making this the most flexible NoSQL model. Data is partitioned and replicated across a cluster to get scalability and availability. For this reason, key value stores often do not support transactions. However, they are highly effective at scaling applications that deal with high-velocity, non-transactional data.

**Document stores:** Document databases typically store self-describing JSON, XML, and BSON documents. They are similar to key-value stores, but in this case, a value is a single document that stores all data related to a specific key. Popular fields in the document can be indexed to provide fast retrieval without knowing the key. Each document can have the same or a different structure.

**Wide-column stores:** Wide-column NoSQL databases store data in tables with rows and columns similar to RDBMS, but names and formats of columns can vary from row to row across the table. Wide-column databases group columns of related data together. A query can retrieve related data in a single operation because only the columns associated with the query are retrieved. In an RDBMS, the data would be in different rows stored in different places on disk, requiring multiple disk operations for retrieval.

**Graph stores:** A graph database uses graph structures to store, map, and query relationships. They provide index-free adjacency, so that adjacent elements are linked together without using an index.

Multi-modal databases leverage some combination of the four types described above and therefore can support a wider range of applications.

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

HBase tables are organized by column, rather than by row. The columns are organized in groups called column families. When creating a HBase table, we must define the column families before inserting any data. Column families should not be changed often, nor should there be too many of them, so it is important to think carefully about what column families will be useful for our particular data. Each column family, however, can contain a very large number of columns. Columns are named using the format family:qualifier.

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 ( not more than three).

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

Tables are declared up front at schema definition time. Row keys are arrays of bytes and they are lexicographically sorted with the lowest order appearing first.

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

Row-oriented data stores –

* Data is stored and retrieved one row at a time and hence could read unnecessary data if only some of the data in a row is required.
* Easy to read and write records
* Well suited for OLTP systems
* Not efficient in performing operations applicable to the entire dataset and hence aggregation is an expensive operation
* Typical compression mechanisms provide less effective results than those on column-oriented data stores

Column-oriented data stores –

* Data is stored and retrieved in columns and hence can read only relevant data if only some data is required
* Read and Write are typically slower operations
* Well suited for OLAP systems
* Can efficiently perform operations applicable to the entire dataset and hence enables aggregation over many rows and columns
* Permits high compression rates due to few distinct values in columns

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.