1. **Give a brief difference between HBASE and HDFS**

HDFS allows you store huge amounts of data in a distributed (provides faster read/write access) and redundant (provides better availability) manner. And MapReduce allows you to process this huge data in a distributed and parallel manner. But MapReduce is not limited to just HDFS. Being a FS, HDFS lacks the random read/write capability. It is good for sequential data access. And this is where HBase comes into picture. It is a NoSQL database that runs on top your Hadoop cluster and provides you random real-time read/write access to your data.

You can store both structured and unstructured data in Hadoop, and HBase as well. Both of them provide you multiple mechanisms to access the data, like the shell and other APIs. And, HBase stores data as key/value pairs in a columnar fashion while HDFS stores data as flat files. Some of the salient features of both the systems are :

**Hadoop**

1. Optimized for streaming access of large files.
2. Follows write-once read-many ideology.
3. Doesn't support random read/write.

**HBase**

1. Stores key/value pairs in columnar fashion (columns are clubbed together as column families).
2. Provides low latency access to small amounts of data from within a large data set.
3. Provides flexible data model.

Hadoop is most suited for offline batch-processing kinda stuff while HBase is used when you have real-time needs.

1. **List the main components of HBASE.**

HBase has three major components: the **client library**, **a master server**, and **region servers**.

**MasterServer**

The master server -

* Assigns regions to the region servers and takes the help of Apache ZooKeeper for this task.
* Handles load balancing of the regions across region servers. It unloads the busy servers and shifts the regions to less occupied servers.
* Maintains the state of the cluster by negotiating the load balancing.
* Is responsible for schema changes and other metadata operations such as creation of tables and column families.

## Regions

Regions are nothing but tables that are split up and spread across the region servers.

### Region server

The region servers have regions that -

* Communicate with the client and handle data-related operations.
* Handle read and write requests for all the regions under it.
* Decide the size of the region by following the region size thresholds.

## Zookeeper

* Zookeeper is an open-source project that provides services like maintaining configuration information, naming, providing distributed synchronization, etc.
* Zookeeper has ephemeral nodes representing different region servers. Master servers use these nodes to discover available servers.
* In addition to availability, the nodes are also used to track server failures or network partitions.
* Clients communicate with region servers via zookeeper.
* In pseudo and standalone modes, HBase itself will take care of zookeeper.

1. **Does Hbase support sql?**

HBase does not support a structured query language like SQL; in fact, HBase isn't a relational data store at all. HBase applications are written in Java much like a typical MapReduce application.

1. **When should we use HBASE, list some of the scenarios for the same**

It is good for lookup use cases but not good for queries/use cases that needs to access all the data. It also needs a careful key/ schema design based on access patterns and use cases.  
Hive on the other hand provides Sql layer on top of HDFS and map reduce. Initially developed and open sourced by Facebook to give analyst ability to write simple sql queries instead of writing map reduce code to do heavy data crunching on big datasets lying on HDFS. It supports most of SQL constructs and provides their efficient map-reduce implementation. It supports different file formats including HFile of HBase.  
 It is highly used batch query language on HDFS and map reduce.

1. **What are the different modes in which Hbase can be run?**

Hbase can run in three different modes

* Standalone mode.
* Pseudo-distributed mode.
* Distributed mode.

1. **Why is zookeeper needed in Hbase?**

The zookeeper maintains configuration information, provides distributed synchronization, and also maintains the communication between clients and region servers.

1. **Hbase is a schema less database, what does it mean?**

What this really means is that the "schema" is stored with the record, not the table. In a RDBMS, the schema is defined and that table has the schema. In HBase (and other BigTable implementations) data is labeled with its types.

1. **What is the minimum number of column family every Hbase table should have?**

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.

1. **What is the benefit of using connection pool in Hbase?**

For applications which require high-end multithreaded access (e.g., web-servers or application servers that may serve many application threads in a single JVM), you can pre-create an HConnection, as shown in the following example.

// create a connection to the cluster.

HConnection connection = HConnectionManager.createConnection (Configuration);

HTableInterface table = connection.getTable ("myTable");

// use table as needed, the table returned is lightweight

table.close ();

// use the connection for other access to the cluster

connection.close ();