Explain in brief

● Differences between HBASE and HDFS.

**Solution**:

**HDFS:**

HDFS is a distributed file system and has the following properties:  
1. It is optimized for streaming access of large files.

2. We would typically store files that are in the 100s of MB upwards on HDFS and access them through MapReduce to process them in batch mode.  
3. HDFS is optimized for use cases where we write once and read many times like in the case of production logs.

4. We can append to files in some of the recent versions but that is not a feature that is very commonly used. There is no concept of random writes.  
5. HDFS doesn’t do random reads very well.

**HBASE:**

HBase on the other hand is a distributed column oriented database.

The filesystem of choice typically is HDFS owing to the tight integration between HBase and HDFS.

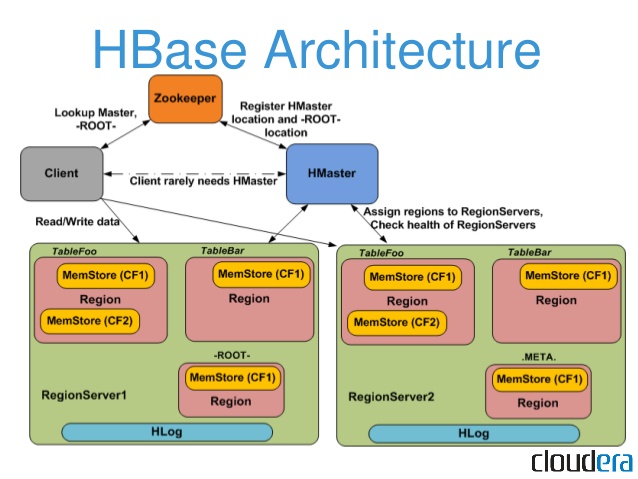
It doesn’t mean that HBase can’t work on any other filesystem.

It’s just not proven in production and at scale to work with anything except HDFS.  
HBase provides you with the following:  
1. It gives us the ability to do random read/writes on your data which HDFS doesn’t allow us to.  
2. HBase stores data in the form of key value pairs in a columnar fashion.

3. HBase provides a flexible data model.  
4. Fast scans across tables.  
5. Scale in terms of writes as well as total volume of data.

* List and explain the main components of HBASE

**Solution:**



HBase has three major components:

1) Client library

2) A master server

3) Region servers

Region servers can be added or removed as per requirement.

**Master Server:**

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 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.



The store contains memory store and HFiles. Memstore is just like a cache memory. Anything that is entered into the HBase is stored here initially. Later, the data is transferred and saved in Hfiles as blocks and the memstore is flushed.

**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.

● Does Hbase support sql?

**Solution:**

HBase is a column-oriented database management system that runs on top of [HDFS](http://www.ibm.com/software/data/infosphere/hadoop/hdfs/).

It is well suited for sparse data sets, which are common in many big data use cases.

Unlike relational database systems, **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](http://www.ibm.com/software/data/infosphere/hadoop/mapreduce/) application.

An HBase system comprises a set of tables.

Each table contains rows and columns, much like a traditional database.

Each table must have an element defined as a Primary Key and all access attempts to HBase tables must use this Primary Key.

An HBase column represents an attribute of an object.

Just as HDFS has a NameNode and slave nodes, and MapReduce has JobTracker and TaskTracker slaves, HBase is built on similar concepts.

In HBase a master node manages the cluster and region servers store portions of the tables and perform the work on the data.