**Assignment 31.2**

**Explain in brief**

**● When should we use HBASE, list some of the scenarios for the same in real time.**

Apache HBase is a column oriented database which supports dynamic database schema. It mainly runs on top of the HDFS and supports MapReduce jobs. HBase also supports other high level languages for data processing.

Following are some points which should be considered for selecting a NoSQL DB like HBase.

Volume: The volume of data is the first criteria for selecting a NoSQL DB. You should have endless data (millions or billions of rows) to process and store. If you only have a few thousands or million of rows then traditional RDBMS is the best fit. But if you select HBase for a small amount of data then the data will accumulate in a single node and the other nodes in the cluster will sit idle.

Hardware support: HDFS performs efficiently when there are at least 5 data nodes. As we know that HBase is based on HDFS, so you should have sufficient hardware support for implementing HBase DB.

No need for RDBMS features: Make sure that your application does not require extra features provided by typical RDBMS. The advanced features like transaction, complex query, triggers are not supported by HBase. So this is another important criterion for selection.

Some real life examples of using HBase are as follows:

**Facebook** is a social utility that connects people with friends and others who work and live around them. Facebook uses HBase mainly to power their messages infrastructure. The following are the services where Facebook uses HBase:

* Messages between users
* Chats
* E-mails
* SMS

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

HBase has two run modes: “Standalone HBase” and “Distributed”. Out of the box, HBase runs in standalone mode. To set up a distributed deploy, we will need to configure HBase by editing files in the HBase conf directory**.**

* **Standalone HBase**

This is the default mode. In standalone mode, HBase does not use HDFS -- it uses the local filesystem instead -- and it runs all HBase daemons and a local ZooKeeper all up in the same JVM. Zookeeper binds to a well known port so clients may talk to HBase.

### Distributed

Distributed mode can be subdivided into distributed but all daemons run on a single node --  pseudo-distributed-- and fully-distributed where the daemons are spread across all nodes in the cluster.

Distributed modes require an instance of the Hadoop Distributed File System (HDFS).

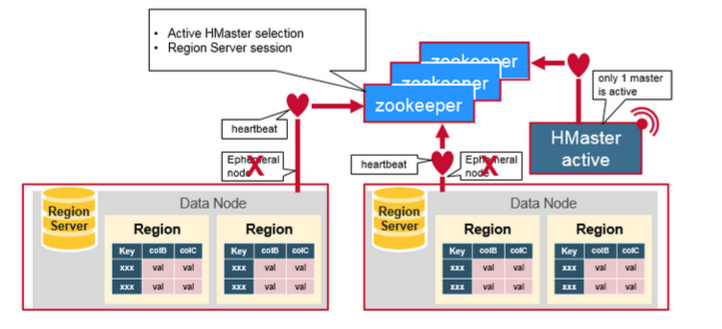
**● Need and working of zookeeper in Hbase?**

Zookeeper

* Manage configuration across nodes – If you have dozens or hundreds of nodes, it becomes hard to keep configuration in sync across nodes and quickly make changes. ZooKeeper helps you to quickly push the configuration changes.
* Implement reliable messaging – With ZooKeeper, you can easily implement a producer/consumer queue that guarantees delivery, even if some consumers or even one of the ZooKeeper servers fails.
* Implement redundant services – With ZooKeeper, a group of identical nodes (e.g. database servers) can elect a leader/master and let ZooKeeper refer all clients to that master server. If the master fails, ZooKeeper will assign a new leader and notify all clients.
* Synchronize process execution – With ZooKeeper, multiple nodes can coordinate the start and end of a process or calculation. This ensures that anyfollow-up processing is done only after all nodes have finished their calculations**.**

**Working of the components of HBase:**

Zookeeper is used to coordinate shared state information for members of distributed systems. Region servers and the active HMaster connect with a session to ZooKeeper. The ZooKeeper maintains ephemeral nodes for active sessions via heartbeats.



Each Region Server creates an ephemeral node. The HMaster monitors these nodes to discover available region servers, and it also monitors these nodes for server failures. HMasters vie to create an ephemeral node. Zookeeper determines the first one and uses it to make sure that only one master is active. The active HMaster sends heartbeats to Zookeeper, and the inactive HMaster listens for notifications of the active HMaster failure.

If a region server or the active HMaster fails to send a heartbeat, the session is expired and the corresponding ephemeral node is deleted. Listeners for updates will be notified of the deleted nodes. The active HMaster listens for region servers, and will recover region servers on failure. The Inactive HMaster listens for active HMaster failure, and if an active HMaster fails, the inactive HMaster becomes active.