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

**Application Types:** HBase is not suitable for transactional applications, large volume MapReduce jobs, relational analytics, etc. It is preferred when you have a variable schema with slightly different rows. It is also suitable when you are going for a key dependent access to your stored data.

**Hardware environment:** HBase runs on top of HDFS. And HDFS works efficiently with a large number of nodes (minimum 5). So, if you have good hardware support, then HBase can be a good selection.

**No requirement of relational features:** Your application should not have any requirement for RDBMS features like transaction, triggers, complex query, complex joins etc. If you can build your application without these features, then go for HBase.

**Quick access to data:**If you need a random and real time access to your data, then HBase is a suitable candidate. It is also a perfect fit for storing large tables with multi structured data. It gives ‘flashback’ support to queries, which makes it more suitable for fetching data in a particular instance of time.

Apart from the above points, HBase is also suitable when you need fault tolerant, fast and usable data management in a non-relational environment.

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

It runs on two modes

* Standalone mode
* Distributed mode

Standalone mode:

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 mode:

Distributed mode can be subdivided into distributed but all daemons run on a single node -- a.k.a 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).

a)Pseudo distributed:

A pseudo-distributed mode is simply a distributed mode run on a single host. Use this configuration testing and prototyping on HBase. Do not use this configuration for production nor for evaluating HBase performance.

b)Fully distributed:

For running a fully-distributed operation on more than one host, make the following configurations. In hbase-site.xml, add the property hbase.cluster.distributed and set it to true and point the HBase hbase.rootdir at the appropriate HDFS NameNode and location in HDFS where you would like HBase to write data.

3. Need and working of zookeeper in Hbase?

ZooKeeper, while being a coordination service for distributed systems, is a distributed application on its own. ZooKeeper follows a simple client-server model where clients are nodes (i.e., machines) that make use of the service, and servers are nodes that provide the service.

A collection of ZooKeeper servers forms a ZooKeeper ensemble. At any given time, one ZooKeeper client is connected to one ZooKeeper server. Each ZooKeeper server can handle a large number of client connections at the same time. Each client periodically sends pings to the ZooKeeper server it is connected to let it know that it is alive and connected.

The ZooKeeper server in question responds with an acknowledgment of the ping, indicating the server is alive as well. When the client doesn't receive an acknowledgment from the server within the specified time, the client connects to another server in the ensemble, and the client session is transparently transferred over to the new ZooKeeper server.