**Uses of NoSQL Databases in brief and the current NoSQL databases**

NoSQL encompasses a wide variety of different database technologies that were developed in response to the demands presented in building modern applications:

* Developers are working with applications that create massive volumes of new, rapidly changing data types — structured, semi-structured, unstructured and polymorphic data.
* Long gone is the twelve-to-eighteen month waterfall development cycle. Now small teams work in agile sprints, iterating quickly and pushing code every week or two, some even multiple times every day.
* Applications that once served a finite audience are now delivered as services that must be always-on, accessible from many different devices and scaled globally to millions of users.
* Organizations are now turning to scale-out architectures using open source software, commodity servers and cloud computing instead of large monolithic servers and storage infrastructure.

Relational databases were not designed to cope with the scale and agility challenges that face modern applications, nor were they built to take advantage of the commodity storage and processing power available today.

**NoSQL Database Types**

* **Document databases** pair each key with a complex data structure known as a document. Documents can contain many different key-value pairs, or key-array pairs, or even nested documents.
* **Graph stores** are used to store information about networks of data, such as social connections. Graph stores include Neo4J and Giraph.
* **Key-value stores** are the simplest NoSQL databases. Every single item in the database is stored as an attribute name (or 'key'), together with its value. Examples of key-value stores are Riak and Berkeley DB. Some key-value stores, such as Redis, allow each value to have a type, such as 'integer', which adds functionality.
* **Wide-column stores** such as Cassandra and HBase are optimized for queries over large datasets, and store columns of data together, instead of rows.

**Differences between Hive and HBase**

Hadoop is, essentially, HDFS (Hadoop Distributed File System) and MapReduce. HDFS is meant for storing massive amounts of data across a distributed system. Technically speaking, your question should be on the difference between HBase and HDFS.

HBase is a non-relational database that can run *on top of Hadoop* and provides you random data access/querying capabilities. HDFS, by itself has no support for reads/writes at random location.

Another primary difference would be the way data is stored in the two. HBase stores data as key/value pairs as in a column database (something similar to Cassandra DB) while data, in HDFS is stored as flat files.

To put it simply, HBase is an extension for the Hadoop environment that allows you to quickly read/write data.