**Assignment 9.3:**

Explain the below concepts with an example in brief.

* Nosql Databases
* Types of Nosql Databases
* CAP Theorem
* HBase Architecture
* HBase vs RDBMS

1. NoSQL Databases:

NoSQL is an approach to database design that can accommodate a wide variety of data models, including key-value, document, columnar and graph formats. NoSQL, which stand for "not only SQL," is an alternative to traditional relational databases in which data is placed in tables and data schema is carefully designed before the database is built. NoSQL databases are especially useful for working with large sets of distributed data. 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.

1. Types of Nosql Databases.

* **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.

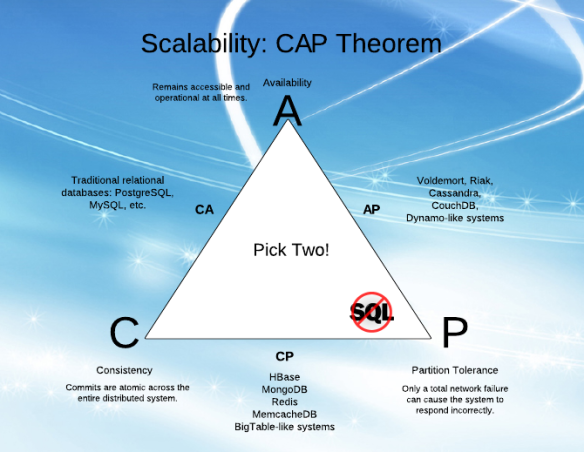
1. CAP Theorem:

One of the first things that you realize, when examining NoSQL distributed databases (and how could you not)is that these days databases are like cars: they're all good. Old fashioned SQL databases can scale in and out, horizontally sharded over several machines to achieve high availability. NoSQL systems claim to be consistent. What difference then does it make what database would you choose?

The Availability and Consistency that I mentioned comes, of course, from the misunderstood  CAP theorem, that - so people say - states that you can only choose 2 out of the 3

* **Consistency:** every read would get you the most recent write
* **Availability:** every node (if not failed) always executes queries
* **Partition-tolerance:** even if the connections between nodes are down, the other two (A & C) promises, are kept.

Usually its depicted in a nicely equilateral triangle, as this one from Ofirm:



CAP theorem: says that in a distributed environment it is not possible for a database to have all the features as C, A, P.

NoSQL databases lies in AP (availability, partition tolerance) & CP (availability, partition tolerance) domains.

NoSQL databases are generally schema less or having dynamic schema.

AP: DynamoDB, Cassandra, CouchDB etc.

CP: mongoDB, Redis, memcache, HBase etc.

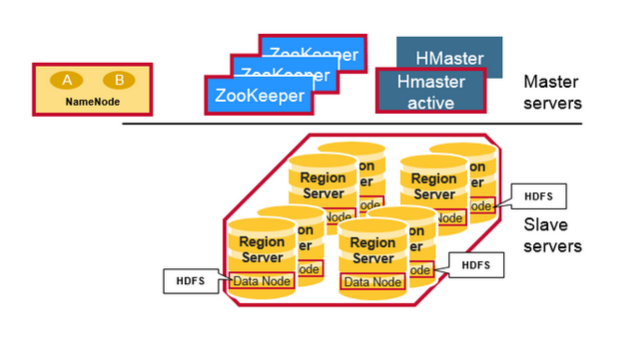
1. HBase Architecture:

HBase is composed of three types of servers in a master slave type of architecture. Region servers serve data for reads and writes. When accessing data, clients communicate with HBase Region Servers directly. Region assignment, DDL (creates, delete tables) operations are handled by the HBase Master process.

Zookeeper, which is part of HDFS, maintains a live cluster state.

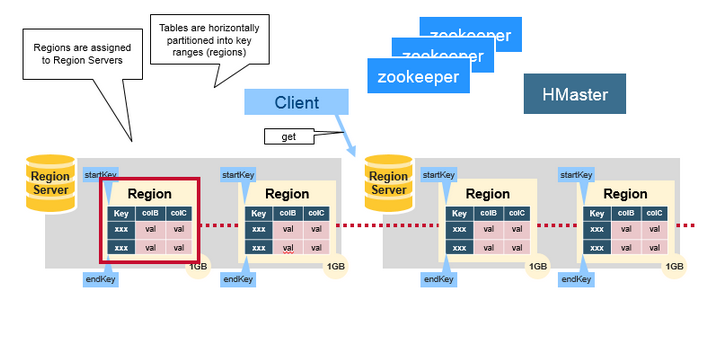
The DataNode stores the data that the Region Server is managing. All HBase data is stored in HDFS files. Region Servers are collocated with the HDFS DataNodes, which enable data locality (putting the data close to where it is needed) for the data served by the Region Servers. HBase data is local when it is written, but when a region is moved, it is not local until compaction.

The NameNode maintains metadata information for all the physical data blocks that comprise the files.



**Regions**

HBase Tables are divided horizontally by row key range into “Regions.” A region contains all rows in the table between the region’s start key and end key. Regions are assigned to the nodes in the cluster, called “Region Servers,” and these serve data for reads and writes. A region server can serve about 1,000 regions.

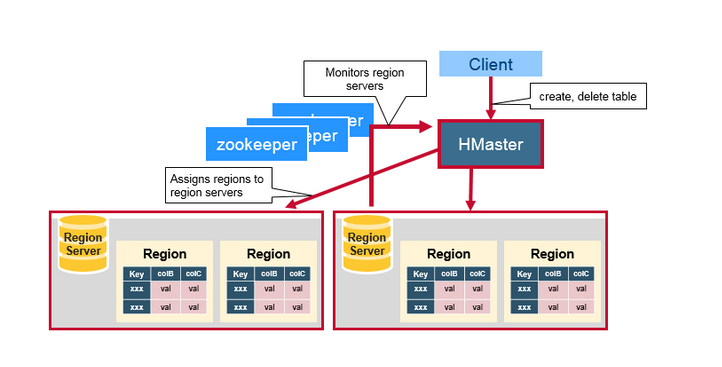


**HBase HMaster**

Region assignment, DDL (create, delete tables) operations are handled by the HBase Master.

A master is responsible for:

* Coordinating the region servers: Assigning regions on start-up, re-assigning regions for recovery or load balancing. Monitoring all RegionServer instances in the cluster (listens for notifications from zookeeper)
* Admin functions: Interface for creating, deleting, updating tables



**ZooKeeper: The Coordinator**

HBase uses ZooKeeper as a distributed coordination service to maintain server state in the cluster. Zookeeper maintains which servers are alive and available, and provides server failure notification. Zookeeper uses consensus to guarantee common shared state. Note that there should be three or five machines for consensus.

1. **HBase vs RDBMS:**

|  |  |
| --- | --- |
| H Base | RDBMS |
| Column-oriented | Row-oriented |
| Flexible schema, add columns on the Fly | Fixed schema |
| Good with sparse tables. | Not optimized for sparse tables. |
| No query language | SQL |
| Wide tables | Narrow tables |
| Joins using MR – not optimized | optimized for Joins(small, fast ones) |
| Tight – Integration with MR | Not really |
| De-normalize your data. | Normalize as you can |
| Horizontal scalability-just add hard war. | Hard to share and scale. |
| Consistent | Consistent |
| No transactions. | transactional |
| Good for semi-structured data as well as structured data. | Good for structured data. |

Hadoop and RDBMS are varying concepts of processing, retrieving and storing the data or information. While Hadoop is an open-source Apache project, RDBMS stands for Relational Database Management System. Hadoop framework has been written in Java which makes it scalable and makes it able to support applications that call for high performance standards. Hadoop framework enables the storage of large amounts of data on files systems of multiple computers. Hadoop is configured to allow scalability from a single computer node to several thousands of nodes or independent workstations in a manner that the individual nodes utilize local computer storage CPU processing power and memory.