HDFS

by – viraj.hadoop@gmail.com

Evolution of HDFS:

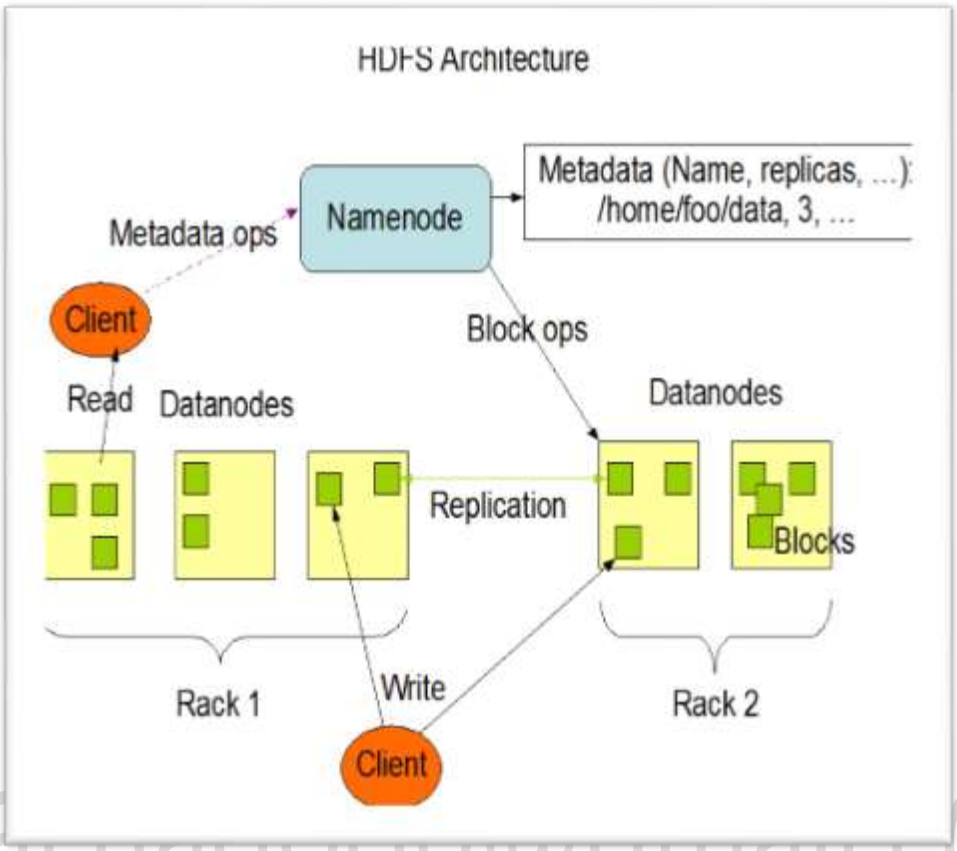
Once Google gave the concept of GFS (Google file system) and MapReduce on paper, another search engine giant Yahoo, took the idea and started working. The main idea behind GFS was to distribute the large files into small files and then process and to process they introduced MapReduce which will process the file parallels.

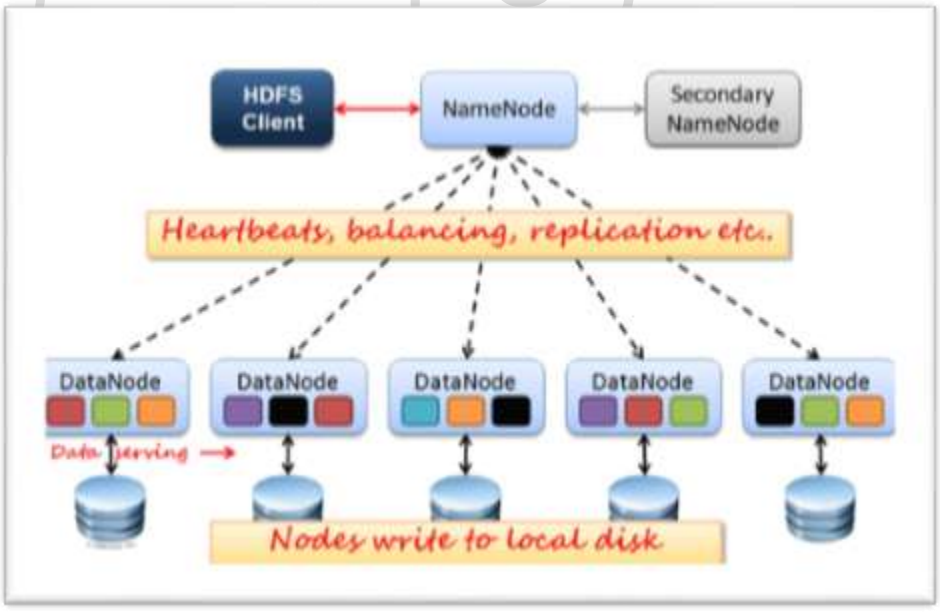
Here large files get divided into small blocks and then getting processed. These block size can be anywhere between 64-128 MB.

NOTE: In Hadoop 1, it was maximum up to 64 MB but in the next version it has been increased up to 128 MB but the default is 64 MB. If you want to increase it, you will have to do the changes from the Hadoop configuration file his-site.xml.

HDFS Architecture?

In HDFS Master Node, a daemon called **namenode** run for HDFS. On all the Slave Node a daemon called **datanode** run for HDFS





NAME NODE:

1. Name node is master daemon.

2. Mangers the data node.
3. Name node contain metadata information (block stored, permission, ip addresses, size of file)
4. If any file is deleted or made change it will affect in edit log.
5. Name node receives Heartbeat intervals every 3 sec. from data node. If name node not get any heartbeat interval from data node. It will be treat as dead node.

DATA NODE:

1. Data node is slave daemon.
2. Actual data is stored in data node.
3. It send heartbeat interval to name node every 3 sec.

Secondary NameNode:

1. Secondary NameNode in hadoop is to take checkpoints of the file system metadata present on namenode.
2. It is not a backup namenode. It just checkpoints namenode's file system namespace.
3. The Secondary NameNode is a helper to the primary NameNode but not replace for primary namenode.

Active and Passive "NameNodes":

Active: Means running on cluster

Passive: Means stand by name node

Data Locality:

In our traditional system, we used to bring the data to the application layer and then process it. But now, because of the architecture and huge volume of the data, bringing the data to the application layer will reduce the network performance to a noticeable extent.

In HDFS, we bring the computation part to the data nodes where the data is residing. Hence, you are not moving the data, you are bringing the program or processing part to the data.

There are two types of scaling: vertical and horizontal.

In **vertical scaling (scale up),** you increase the hardware capacity of your system. In other words, you procure more RAM or CPU and add it to your existing system to make it more robust and powerful.

In **horizontal scaling (scale out),** you add more nodes to existing cluster instead of increasing the hardware capacity of individual machines. And most importantly, you can add more machines on the go i.e. without stopping the system

Data Integrity:

Data Integrity talks about whether the data stored in my HDFS are correct or not. HDFS constantly checks the integrity of data stored against its checksum. If it finds any fault, it reports to the name node about it. Then, the name node creates additional new replicas and therefore deletes the corrupted copies.

High Throughput:

Throughput is the amount of work done in a unit time. It talks about how fast you can access the data from the file system.

Rack Awareness:

Rack Awareness is the algorithm in which the “NameNode” decides how blocks and their replicas are placed, based on rack definitions to minimize network traffic between “DataNodes” within the same rack. Let’s say we consider replication factor 3 (default), the policy is that “for every block of data, two copies will exist in one rack, third copy in a different rack”. This rule is known as the “Replica Placement Policy”.

Speculative Execution:

If a node appears to be executing a task slower, the master node can redundantly execute another instance of the same task on another node. Then, the task which finishes first will be accepted and the other one is killed. This process is called “speculative execution”.

HDFS Block is the physical division of the data.

HDFS divides data in blocks for storing the blocks together, whereas for processing,

Input Split is the logical division of the data.

MapReduce divides the data into the input split and assign it to mapper function.

	RDBMS	Hadoop
Data Types	RDBMS relies on the structured data and the schema of the data is always known.	Any kind of data can be stored into Hadoop i.e. Be it structured, unstructured or semi-structured.
Processing	RDBMS provides limited or no processing capabilities.	Hadoop allows us to process the data which is distributed across the cluster in a parallel fashion.
Schema on Read Vs. Write	RDBMS is based on 'schema on write' where schema validation is done before loading the data.	On the contrary, Hadoop follows the schema on read policy.
Read/Write Speed	In RDBMS, reads are fast because the schema of the data is already known.	The writes are fast in HDFS because no schema validation happens during HDFS write.
Cost	Licensed software, therefore, I have to pay for the software.	Hadoop is an open source framework. So, I don't need to pay for the software.
Best Fit Use Case	RDBMS is used for OLTP (Online Trasanctional Processing) system.	Hadoop is used for Data discovery, data analytics or OLAP system.