# Hadoop - HDFS Overview

Hadoop File System was developed using distributed file system design. It is run on commodity hardware. Unlike other distributed systems, HDFS is highly faulttolerant and designed using low-cost hardware.

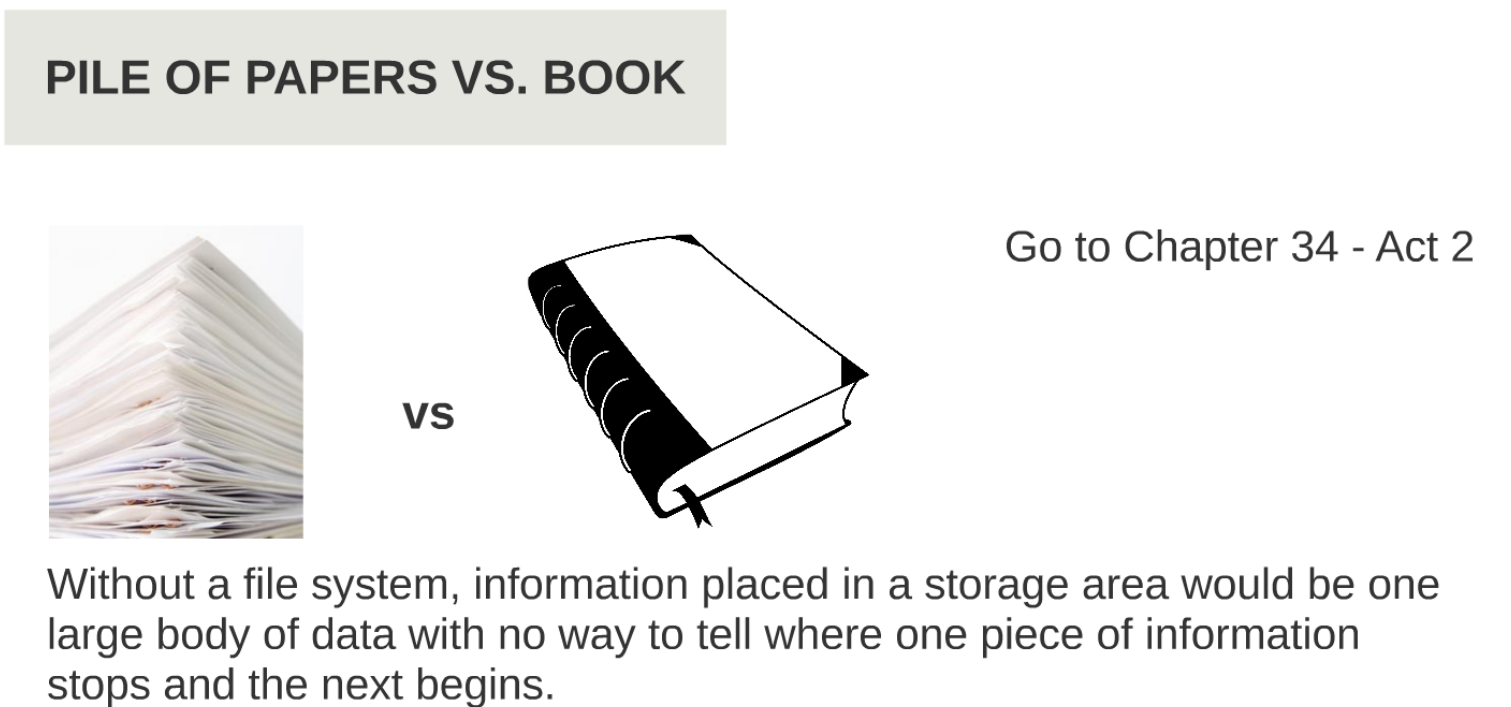
HDFS holds very large amount of data and provides easier access. To store such huge data, the files are stored across multiple machines. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure. HDFS also makes applications available to parallel processing.

# Features of HDFS

* It is suitable for the distributed storage and processing.
* Hadoop provides a command interface to interact with HDFS.
* The built-in servers of namenode and datanode help users to easily check the status of cluster.
* Streaming access to file system data.
* HDFS provides file permissions and authentication.

# Why file system

If we are given a pile of paper and a bok and asked to go to chapter 34, i will be difficult to find the smae in pile of paper, but using a book we cn get the page number of chapter from index and go directly to chapter 34, this make the searching easier



# Functions of File System

* Control how data is stored and retrieved
* Metadata about the files and folder
* Permission and scurity
* Manage storage space efficiently

# Existing File systems

Different OS have different supported file system and their own individual limit on file and logical drives.

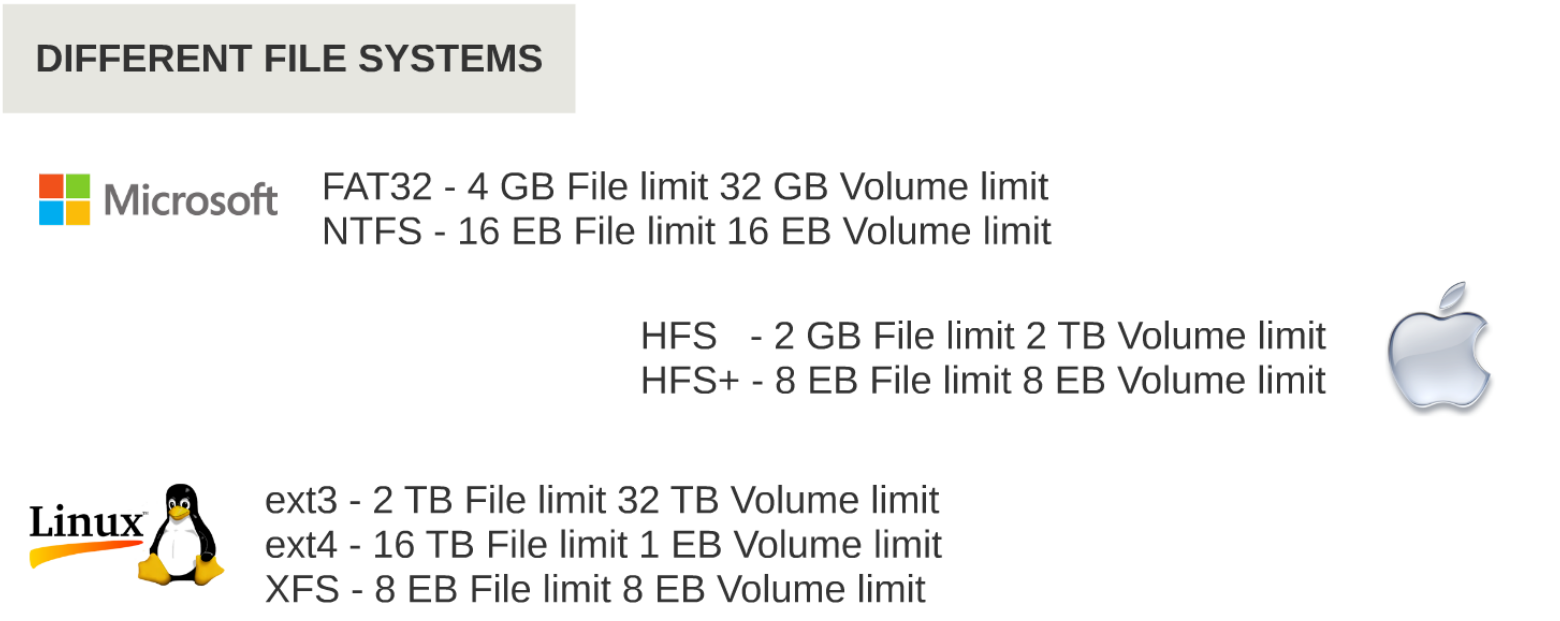
FAT - File Allocation Table

NTFS - new technology file system

HFS - Hierarchical File System

Ext - extended file system

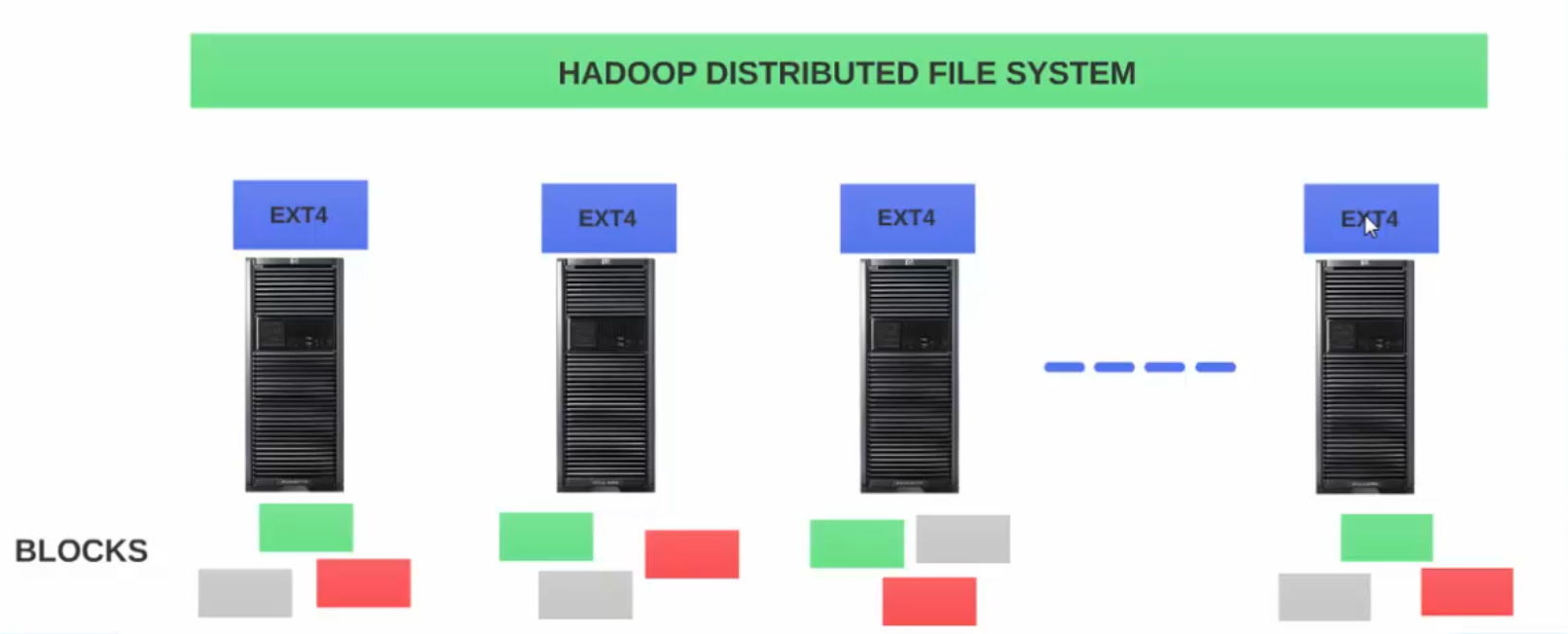
XFS - Extents File System





# HDFS why another File system?

For truly parallel computation, need divide the data in to blocks and for recovery each block need duplicate in more than 1 node.



Suppose we have N node cluster and have ext4 as the local File system in each node.

- when upload the data to file system, it should divide the data to fixed size blocks

- each exiting file system has its own block feature, the concept of block in hdfs is different then concept of block in traditional fle system.

- The file system should have a distributed view of files in cluster, which is not possile with local file system. The local file system in node 1 has no detail what is in node 2

Suppose if we have a file system on top of existing file system, and it spreads across all the node in cluster, this is called HDFS.

When a file is uploaded to HDFS

- the file is automatically split in to **fixed size block (128 MB) in older version (64MB)**

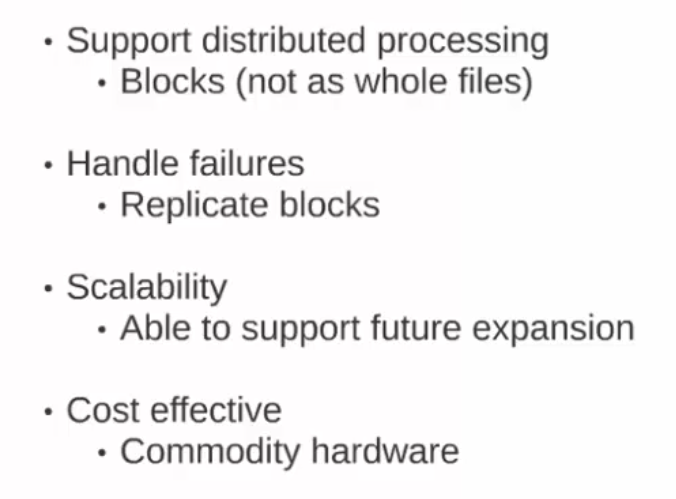
- also replicate each block in to multiple nodes in cluster, **by default HDFS replicate block to 3 node**

- hdfs keeps track of all the block of a file and the node assigned to each block, so when user tries to query the file, HDFS gets all the block and display the file to user.

**Question: what happen to local file system which is on each node.**

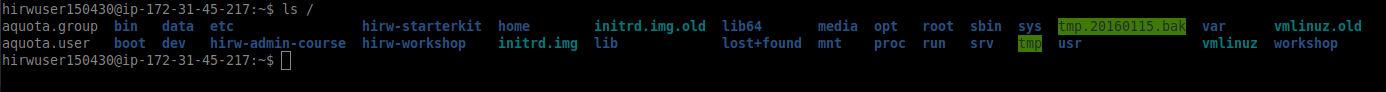
**Solution: Hdfs is not a replacment for local file system. OS on that node still reply on the local file system, HDFS still go through the local file system to store the block in storage. HDFS is placed on top of local file system**

# Benefits of HDFS

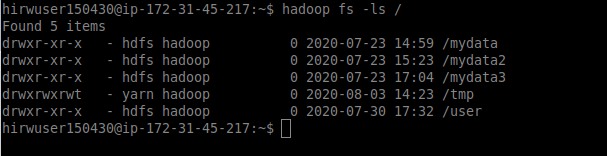


# Working with HDFS

Local file system



HDFS file system



As you can see the listing of root directory in both local file system and hadoop file system are different.

### LOCAL FILE SYSTEM ###

ls

mkdir

cp

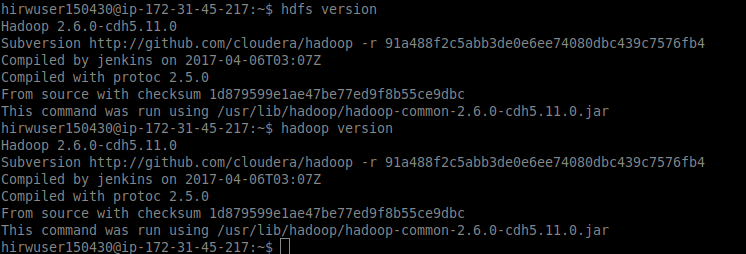
mv

rm

### GET version of HADOOP/HDFS###

hdfs version

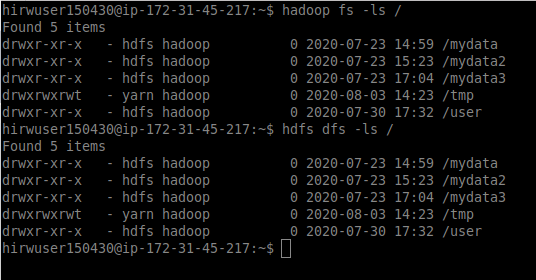
hadoop version



### LISTING ROOT DIRECTORY ###

hadoop fs -ls /

hdfs dfs -ls /



### LISTING DEFAULT TO HOME DIRECTORY ###

hadoop fs -ls

hadoop fs -ls /user/hirwuser150430

hdfs dfs -ls

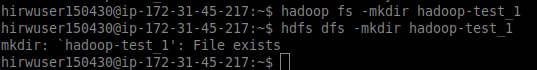
hdfs dfs -ls /user/hirwuser150430

### CREATE A DIRECTORY IN HDFS ###

hadoop fs -mkdir hadoop-test\_1

hdfs dfs -mkdir hadoop-test\_1

Note: /hadoop-test\_1 absolute path without / is relative to home directory



### COPY FROM LOCAL FS TO HDFS ###

hadoop fs -copyFromLocal /hirw-starterkit/hdfs/commands/dwp-payments-april10.csv hadoop-test1

haoop fs -put <localsrc> <dest>

hdfs dfs -copyFromLocal /hirw-starterkit/hdfs/commands/dwp-payments-april10.csv hadoop-test1

hdfs dfs -put <localsrc> <dest>

### COPY TO HDFS TO LOCAL FS ###

hadoop fs -copyToLocal hadoop-test1/dwp-payments-april10.csv .

hadoop fs -get <src> <localdest>

hdfs dfs -copyToLocal hadoop-test1/dwp-payments-april10.csv .

hdfs dfs -get <src> <localdest>

### CREATE 2 MORE DIRECTORIES ###

hadoop fs -mkdir hadoop-test2

hadoop fs -mkdir hadoop-test3

### print file contents ###

hdfs dfs -cat <path>

Hadoop fs -cat <path>

### COPY A FILE FROM ONE FOLDER TO ANOTHER ###

hadoop fs -cp hadoop-test1/dwp-payments-april10.csv hadoop-test2

hdfs dfs -cp hadoop-test1/dwp-payments-april10.csv hadoop-test2

### MOVE A FILE FROM ONE FOLDER TO ANOTHER ###

hadoop fs -mv hadoop-test1/dwp-payments-april10.csv hadoop-test3

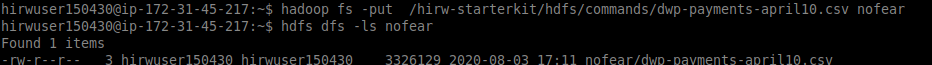
hdfs dfs -mv hadoop-test1/dwp-payments-april10.csv hadoop-test3

### CHECK REPLICATION ###

hadoop fs -ls <fille/folder>

Hdfs dfs -ls <fille/folder>

Here we can see the replication fator is 3, it indicatethe file is replicated in 3 node



### CHANGE OR SET REPLICATION FACTOR ###

hadoop fs -Ddfs.replication=2 -cp hadoop-test2/dwp-payments-april10.csv hadoop-test2/test\_with\_rep2.csv

hdfs dfs -Ddfs.replication=2 -cp hadoop-test2/dwp-payments-april10.csv hadoop-test2/test\_with\_rep2.csv

hdfs dfs -setrep -R -w 6 geeks.txt

**Note:** The **-w** means wait till the replication is completed. And **-R** means recursively, we use it for directories as they may also contain many files and folders inside them.

hadoop fs -ls hadoop-test2

hadoop fs -ls hadoop-test2/test\_with\_rep2.csv

### CHANGING PERMISSIONS ###

hadoop fs -chmod 777 hadoop-test2/test\_with\_rep2.csv

hdfs dfs -chmod 777 hadoop-test2/test\_with\_rep2.csv

### FILE SYSTEM CHECK - REQUIRES ADMIN PREVILEGES ###

sudo -u hdfs hdfs fsck /user/hirwuser150430/hadoop-test2 -files -blocks -locations

sudo -u hdfs hdfs fsck /user/hirwuser150430/hadoop-test3 -files -blocks -locations

sudo -u hdfs hdfs fsck /user/ubuntu/input/yelp/yelp\_academic\_dataset\_review.json -files -blocks -locations

vi /etc/hadoop/conf/hdfs-site.xml

/data/1/dfs/dn/current/BP-2125152513-172.31.45.216-1410037307133/current/finalized

### DELETE DIR/FILES IN HDFS ###

hadoop fs -rm hadoop-test2/test\_with\_rep5.csv

hadoop fs -rm -r hadoop-test1

hadoop fs -rm -r hadoop-test2

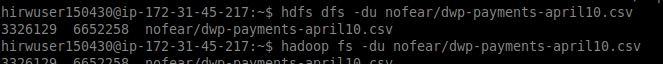
hadoop fs -rm -r hadoop-test3

hdfs dfs -rmr <filename/directoryName>

### size of each file in directory ###

hdfs dfs -du <dirName>

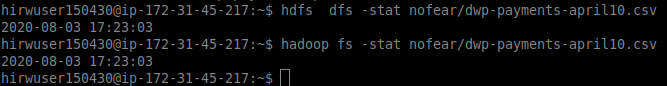
hadoop fs -du <dirName>



### last modified time of file or folder###

hdfs dfs -stat <hdfs file>

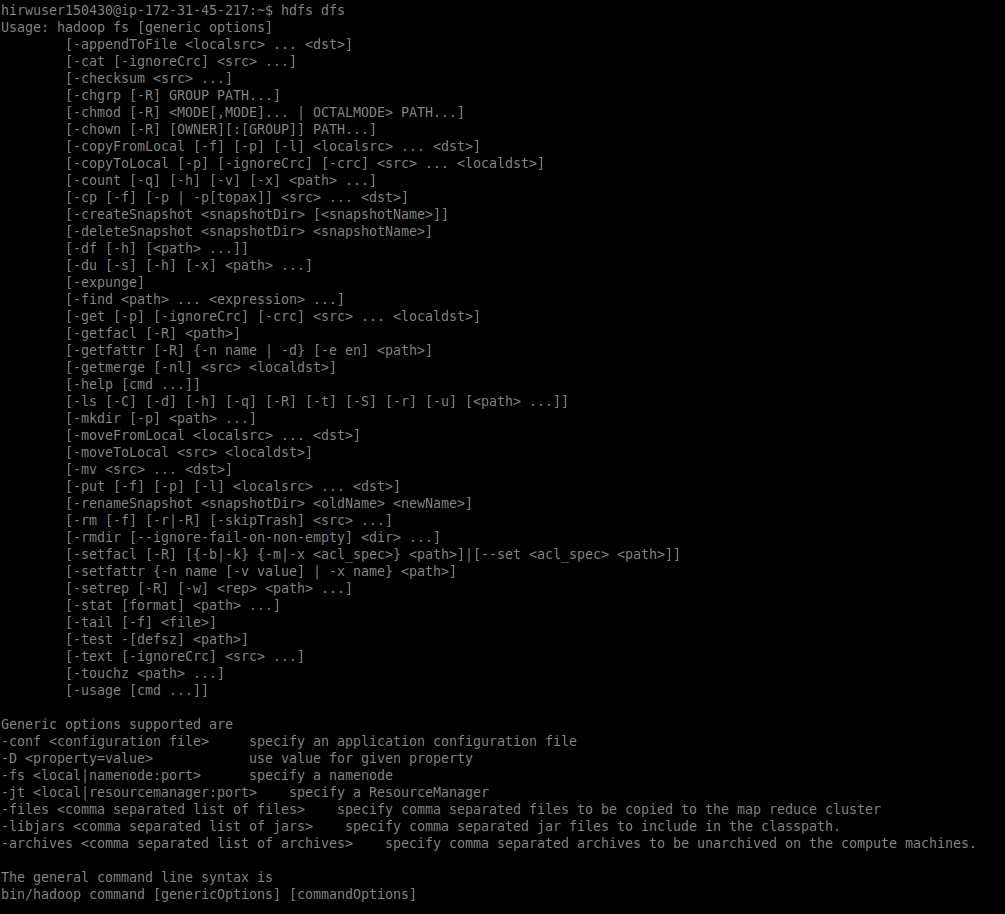
Hadoop fs -stat <hdfs file>

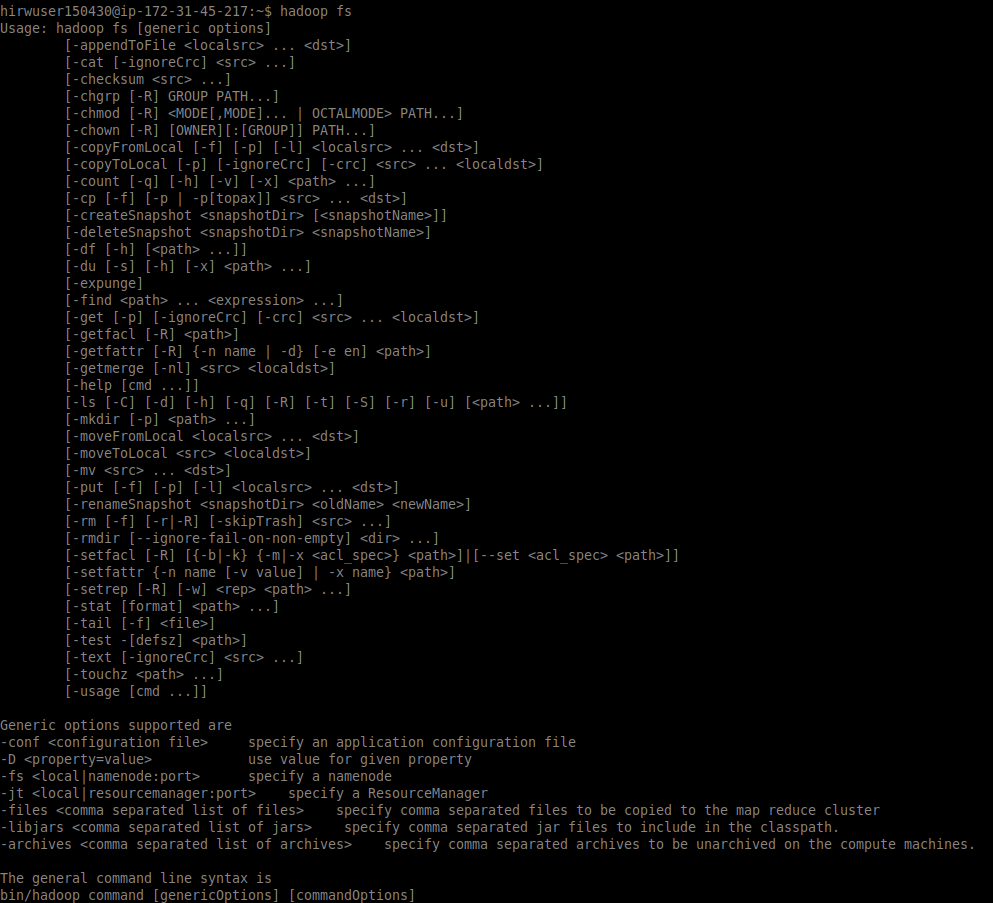


### list all hdfs and hadoop command###

Hadoop fs

Hdfs dfs





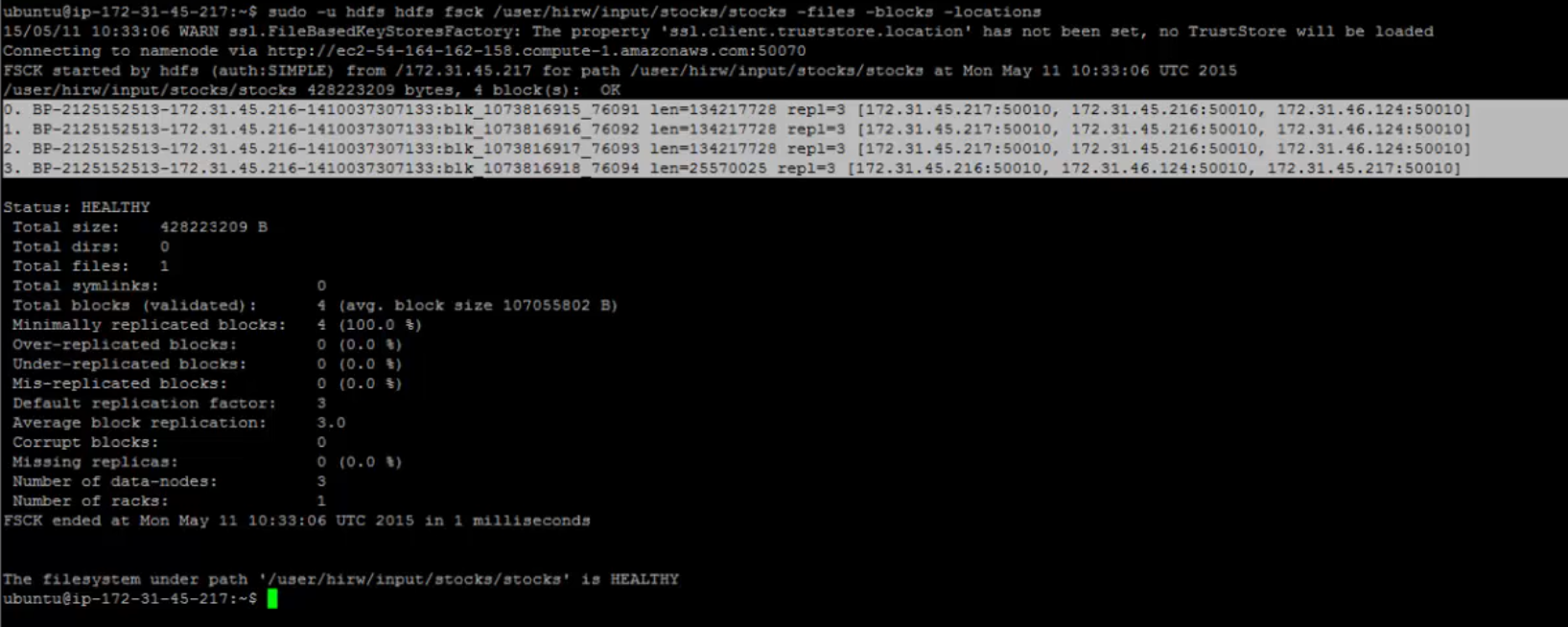
# HDFS Architecture

When use the fsck command

- gives the status of file healthy or not

- it will list ho many block the file is made of

- the location of block and in wich node the blocks are replicated

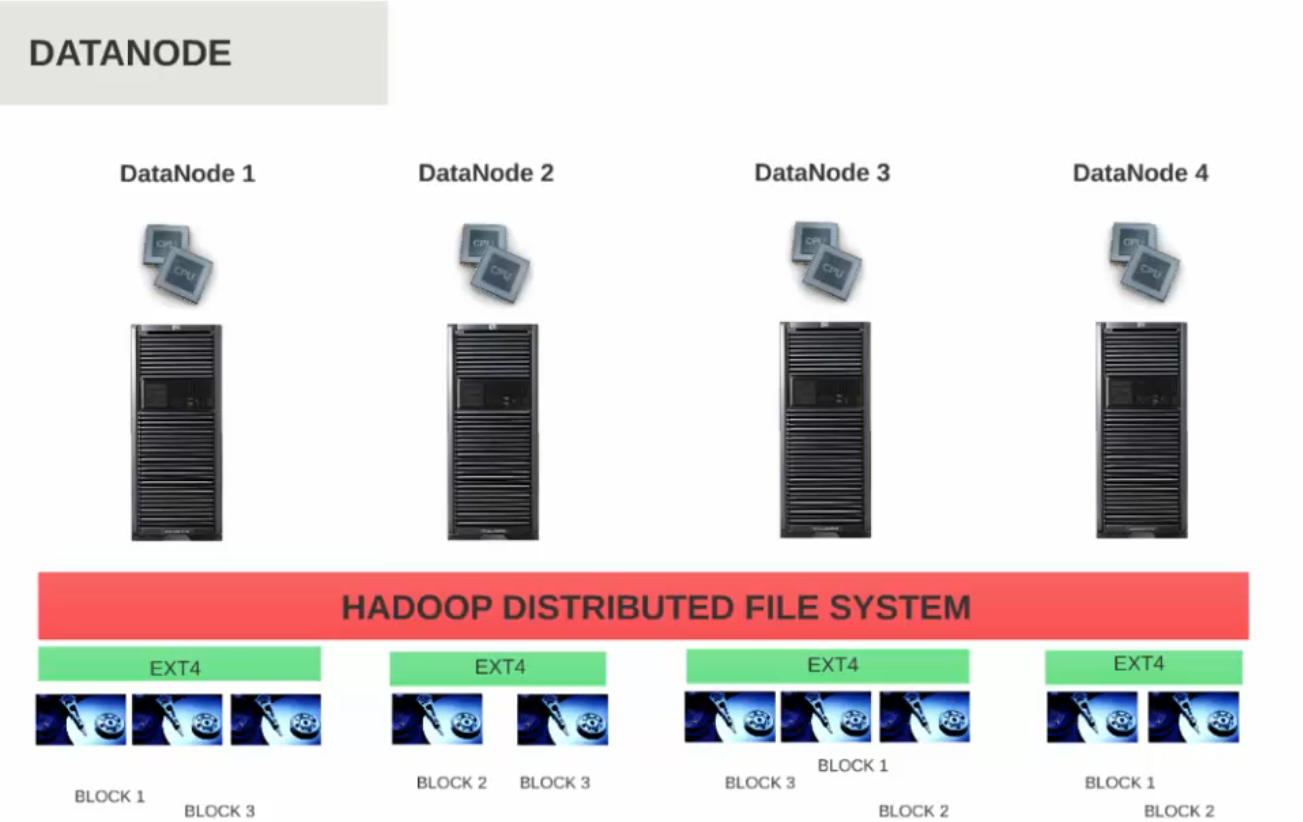


**Data node (slave node):**

Each node where the block ar physically located are called Data node.

Each data node knows the block which it holds.

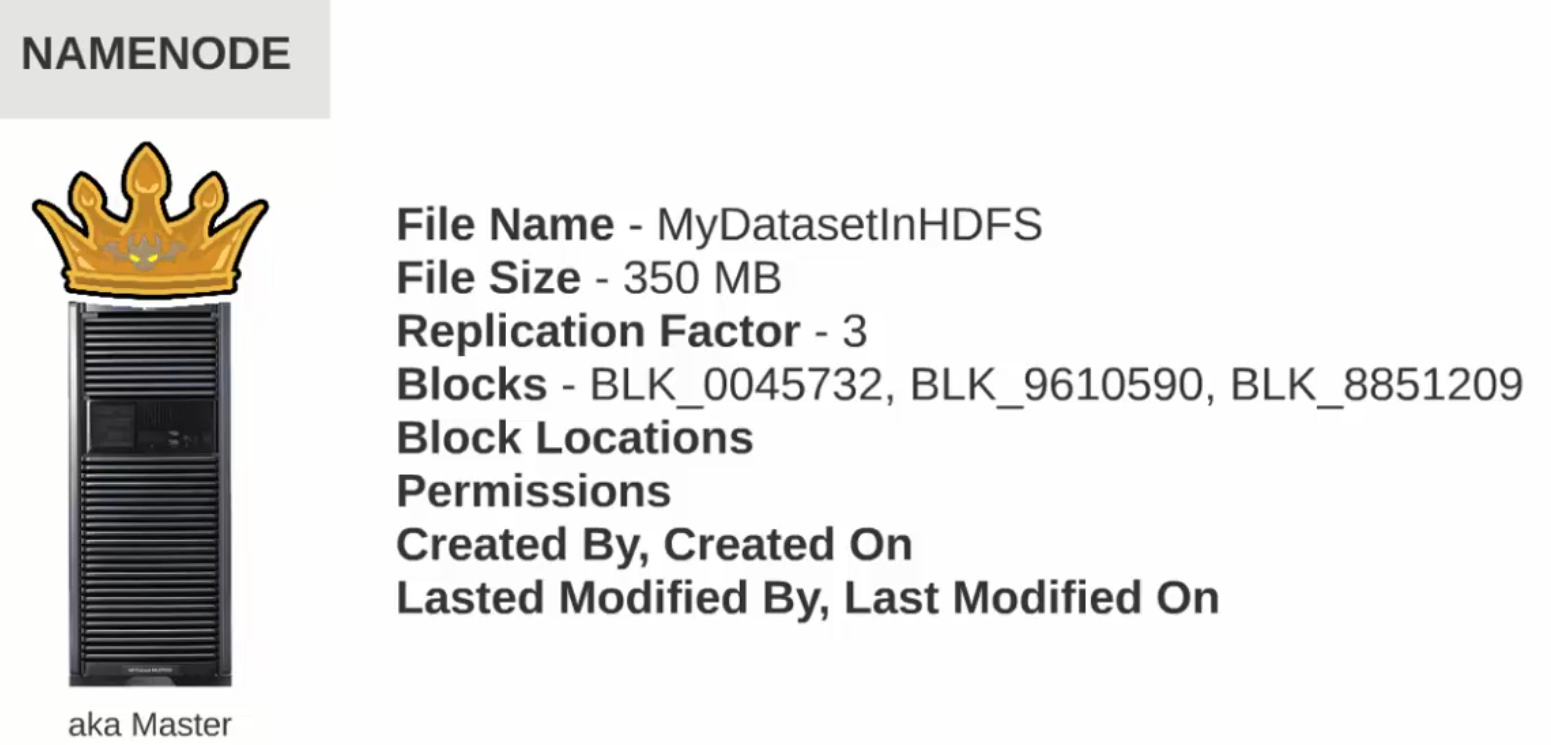
Data node does not know which blcok belongs to which file



**Name Node (master node):**

Is the master node which knows the detail of which block belongs to which file and where it is located

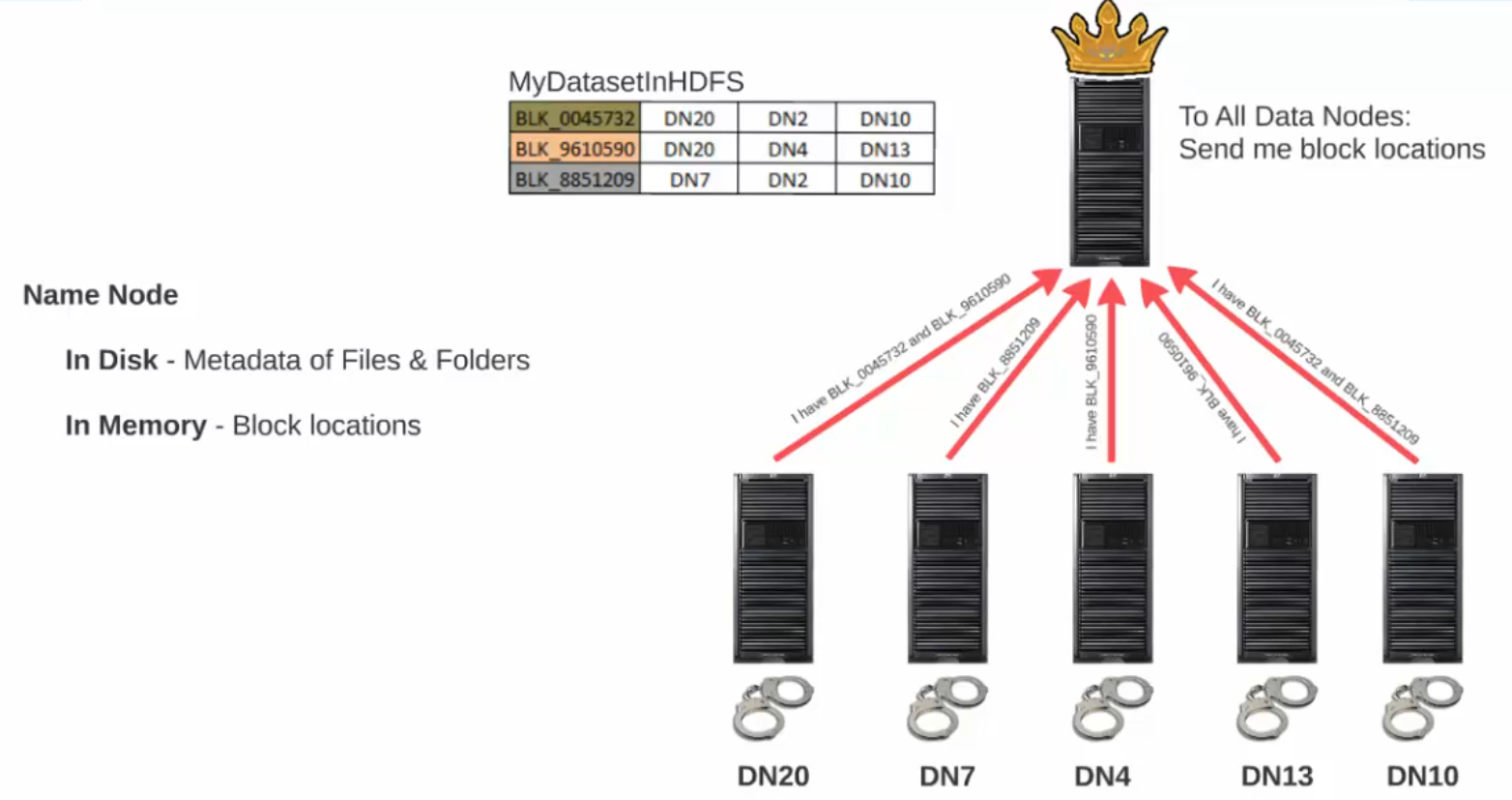
Keeps track of all files in HDFS.



Name node does not keep the block location in disk, it keeps it in meory.

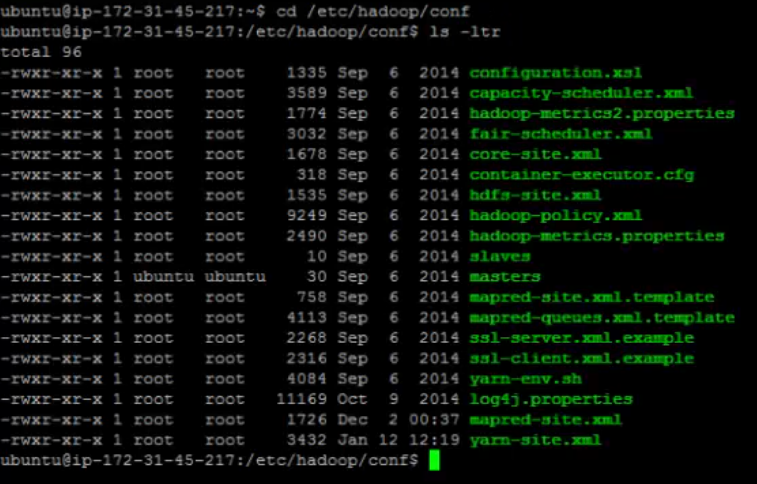
When the data node startup, it broadcast tge list of block it holds to name node, name node keep that information in memory, this is to make the processing faster by name node,

Hdfs has lot of block to process, if name node keep the detail in disk, then getting the information of block becomes bottle neck and reduce the performance



**How does the data node knows the location of name node?**

**The hadoop configuration files are prsent in /etc/hadoop/conf**



In the file **core-site.xml** we can find the detal of the Name node

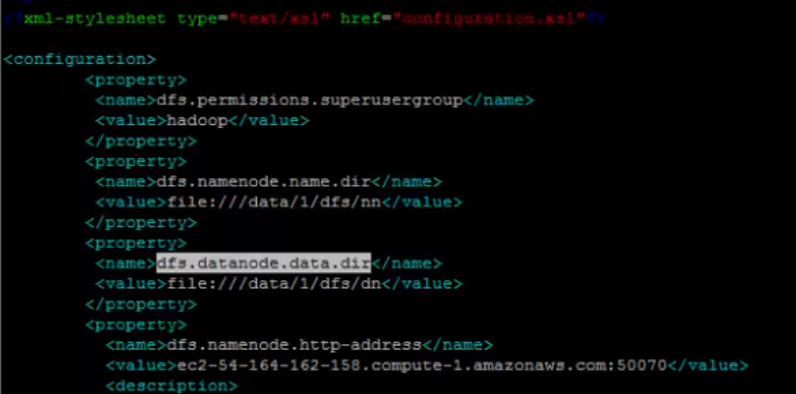


Another important file is **hdfs-site.xml**

dfs.namenode.name.dir -- location where name node store the data/block

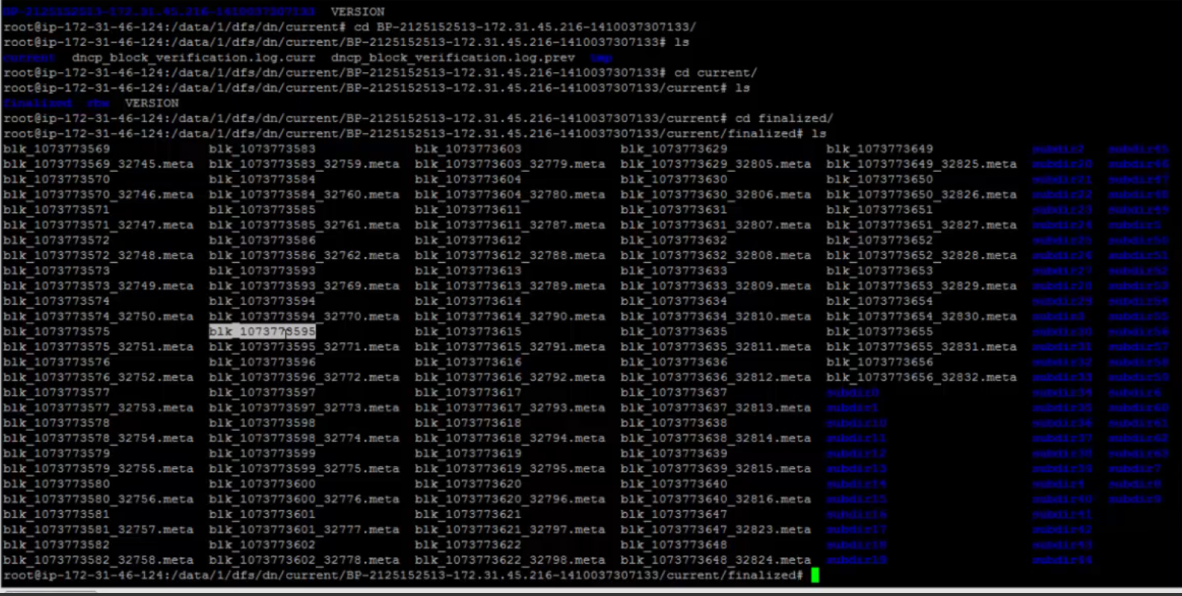
dfs.datanode.data.dir -- location where data node store the data/block

This file also contain the default replication factor of a file.

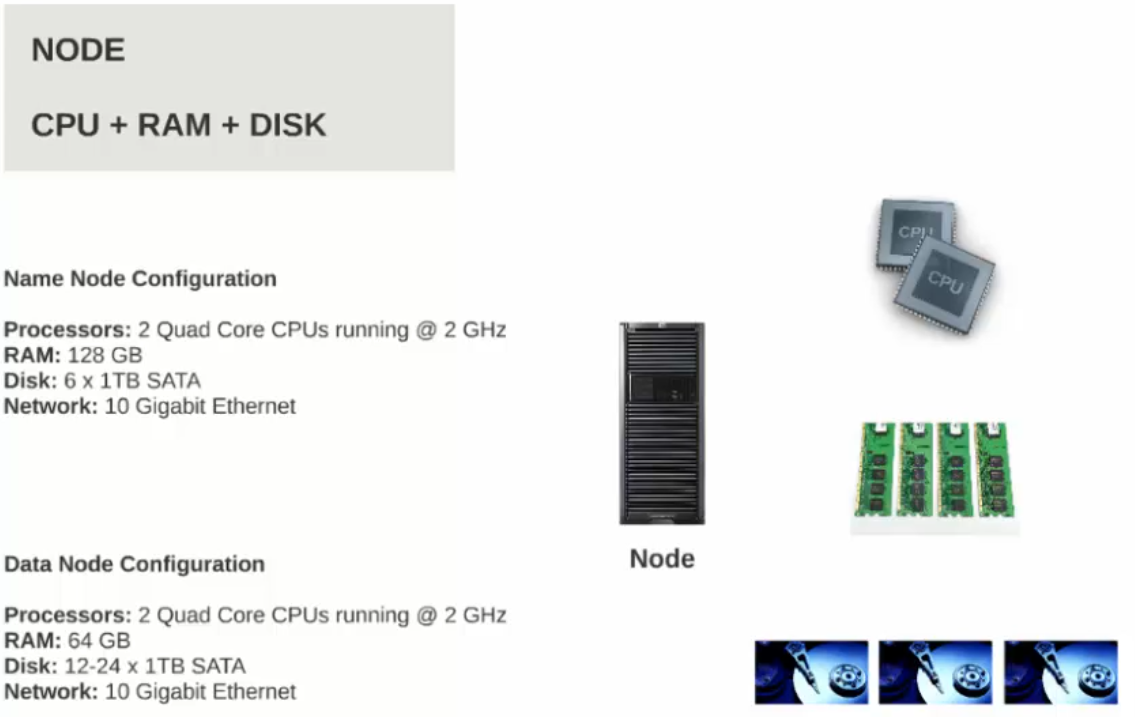


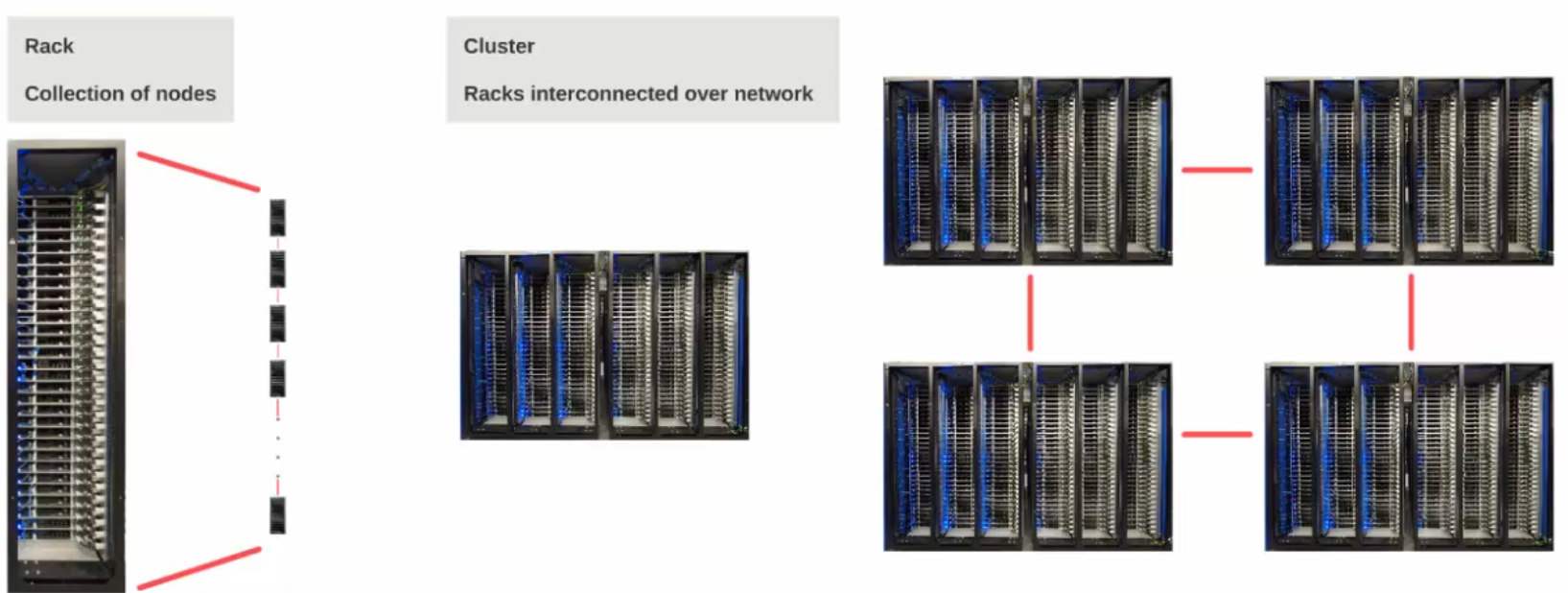
Location in data node where the block are stored

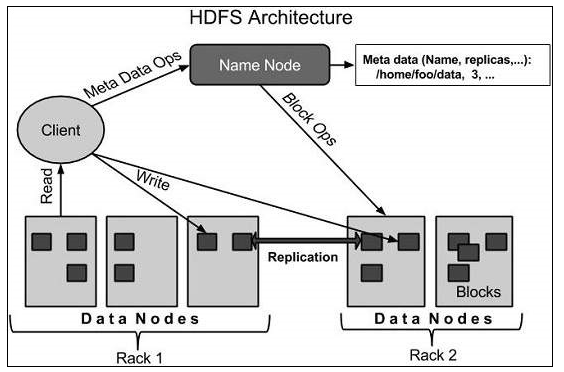
/data/1/dfs/dn



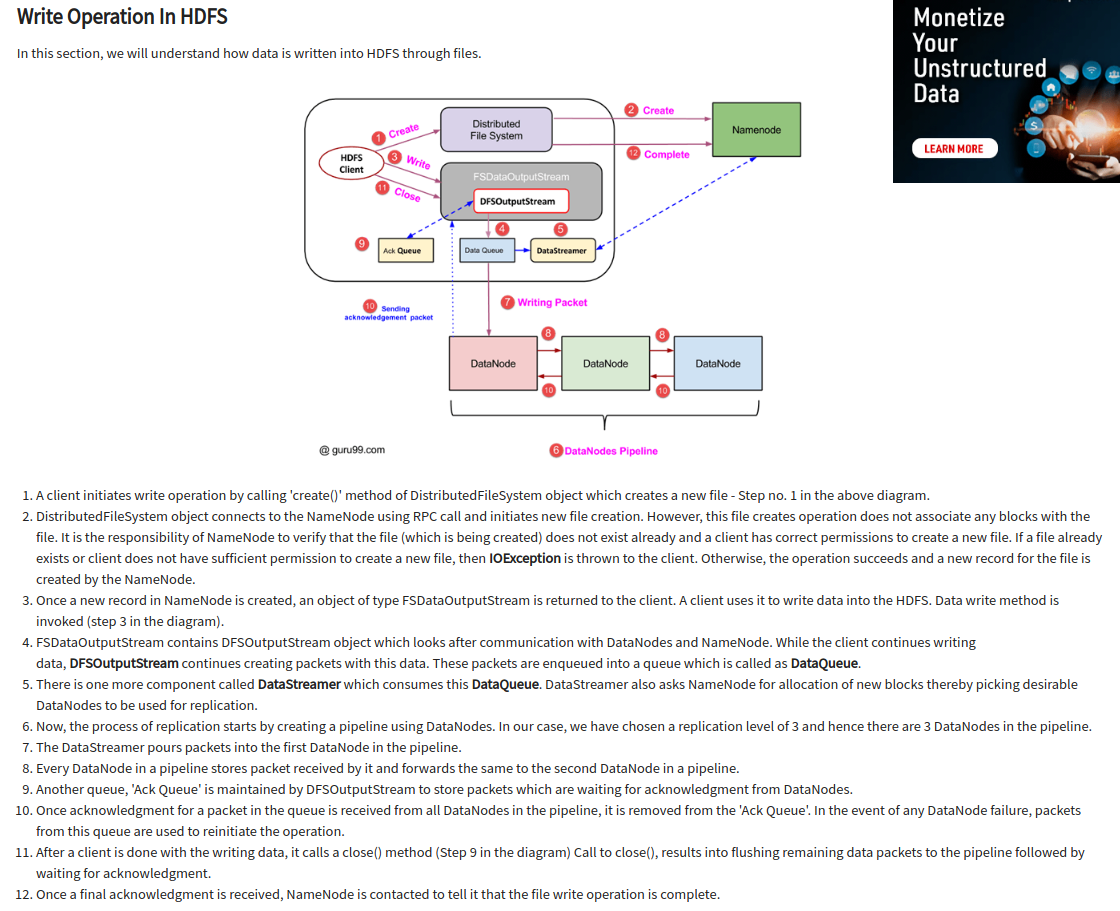
The name node is usually high in memory, the data node is high in hard disk space











Links:

<https://www.guru99.com/learn-hdfs-a-beginners-guide.html>

<https://data-flair.training/blogs/hadoop-hdfs-architecture/>

<https://www.tutorialspoint.com/hadoop/index.htm>