Disclaimer: **These slides are copyrighted and strictly for personal use only**

- This document is reserved for people enrolled into the [Ultimate Big Data Masters Program (Cloud Focused) by Sumit Sir]

- **Please do not share this document,** it is intended for personal use and exam preparation only, thank you.

- If you've obtained these slides for free on a website that is not the course's website, please reach out to legal@trendytech.in . Thanks!

- All the Best and Happy Learning!

HDFS

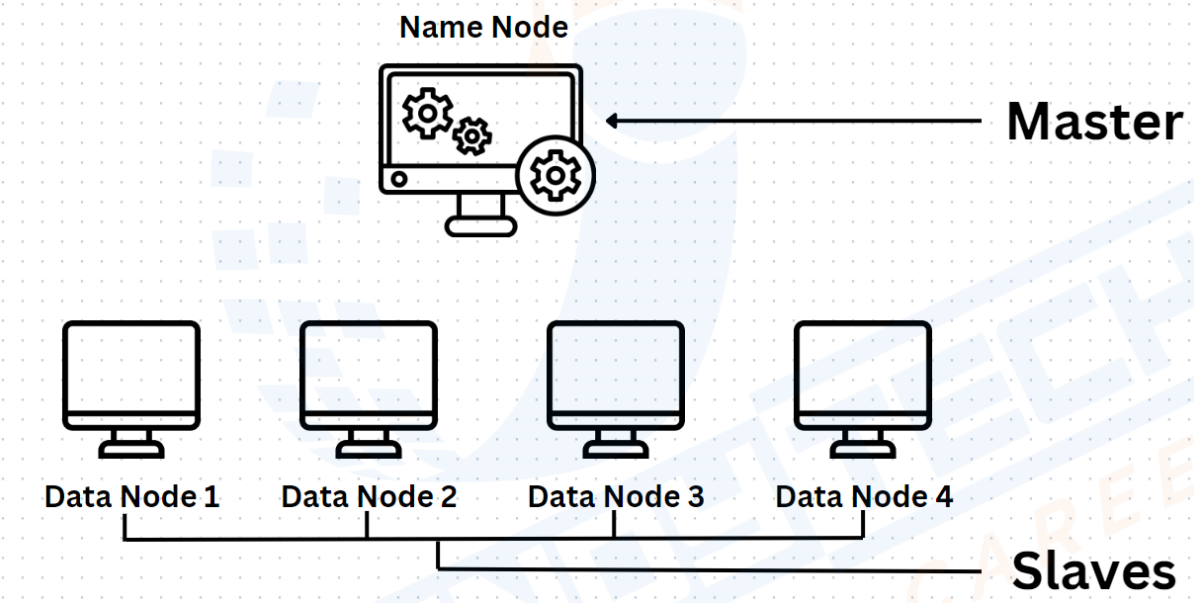
HDFS follows a Master Slave Architecture having one Master (known as Name Node) and multiple Slaves (known as Data Node)

Example :

A cluster, with :

1 Master / Name Node

4 Slaves / Data Nodes (DN1, DN2, DN3, DN4)



Consider a use-case where - file1.txt, a 500mb file(taken just for explanation. In real-time scenarios, files are in petabytes) has to be stored. How to store such a huge file?

Case 1 : Store the complete file in a single Data Node - DN1. This is not the right approach as all the other Data nodes are idle and this is similar to a Monolithic setup which is not a scalable approach.

Case 2 : Divide the file into blocks. By default, the Block Size in the latest version of Hadoop is 128 mb.

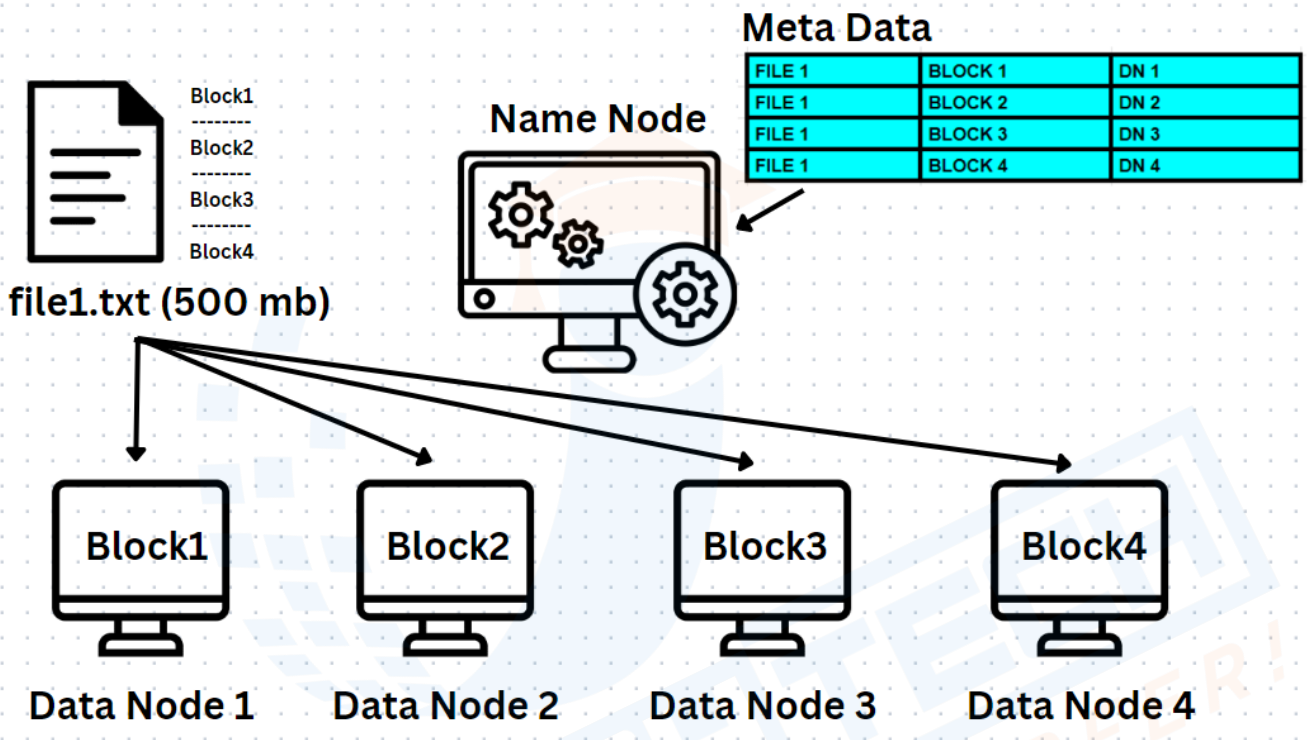
Suppose we take our example file of 500 mb, to how many blocks can this file be divided?

1st block - 128 mb

2nd block - 128 mb

3rd block - 128 mb

4th block - 116 mb



Note :

- Data node stores the actual data.
- Master node stores the mapping information or metadata. It has the information of where the actual data is stored.

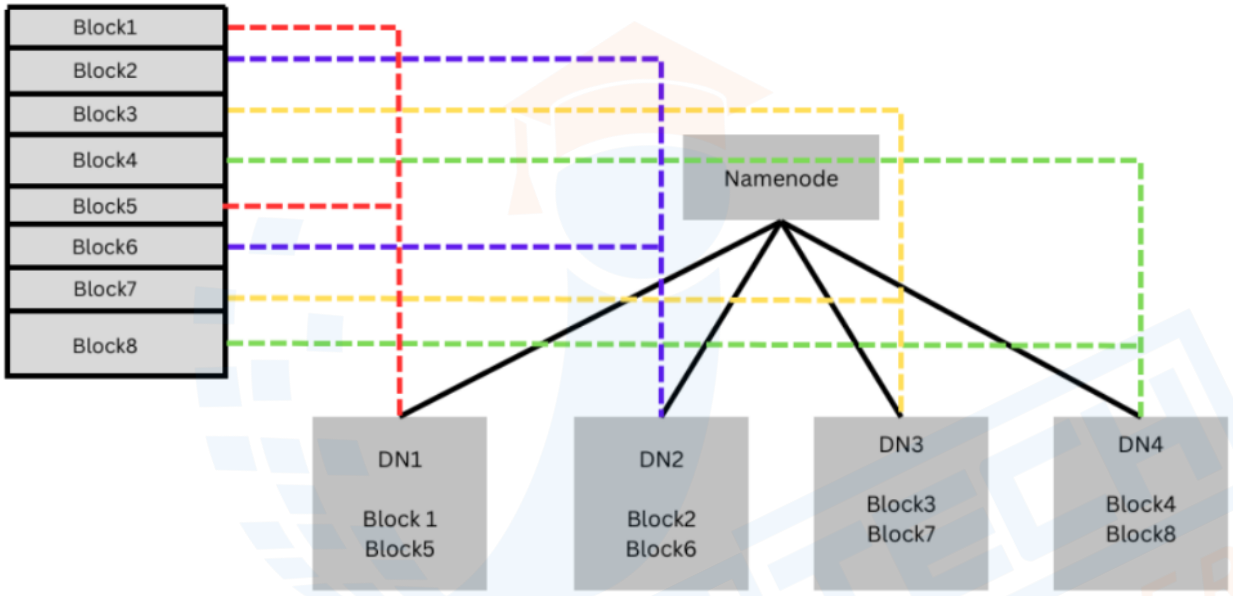
FILE 1	BLOCK 1	DN 1
FILE 1	BLOCK 2	DN 2
FILE 1	BLOCK 3	DN 3
FILE 1	BLOCK 4	DN 4

Another Example:

Suppose file size= 1GB = 1024 MB

No of blocks=1024/128 = 8 blocks.

We are having 4 node cluster, where each node contains multiple blocks as depicted in the diagram below. Note that, each data node can contain multiple blocks.



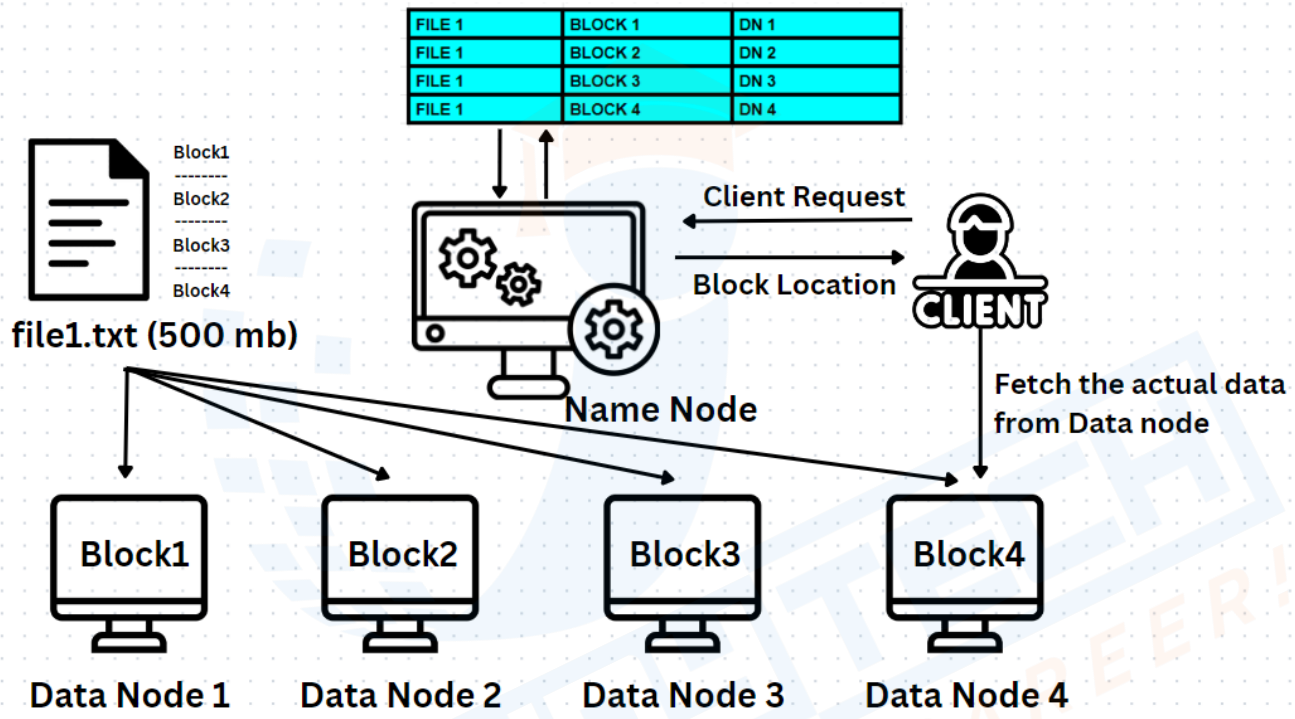
How the Client Request is Processed? Say the client wants to read file1 from HDFS, then

1. When a client requests for file read, the request will first go to the name node.
2. The name checks its meta data table to find the location where the actual data is stored.
3. The name node passes the location information to the client from where the client can read the blocks.

A real-time example to explain the process -

Consider you have a 1000 page book with an index page.

- **The Index Page is like the metadata table in the name node**
- **The 1000 pages are the data nodes that store the actual data.**



Reason for having 128 mb as the default block size :

- **If the block size is decreased, i.e., < 128 mb**

Consider 1 GB file	
Default Block Size	Number of Blocks
128 MB	8 Blocks
64 MB	16 Blocks
32 MB	32 Blocks

Advantage : leads to an increase in the number of blocks and thereby increases parallelism.

Drawback : The Name Node gets overburdened with having to store many entries in its table.

- **If the block size is increased, i.e., > 128 mb**

Consider 1 GB file	
Default Block Size	Number of Blocks
128 MB	8 Blocks
256 MB	4 Blocks
512 MB	2 Blocks

Advantage : Name node is less burdened as it needn't have to store too many entries in the metadata table.

Drawback : Compromising on Parallelism. Less number of nodes implies less parallelism.

- **Therefore, a balanced approach of 128 mb is considered to get the best performance.**

Name Node Federation :

- In the newer versions of Hadoop, Name Node Federation is introduced where there can be more than one name node to handle the growing metadata.

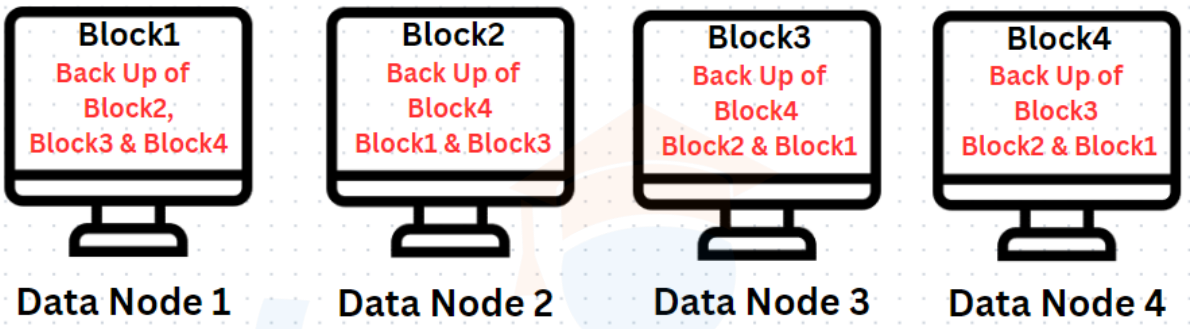
In order to overcome the bottleneck and avoid performance issues that might arise due to growing metadata information, the metadata is distributed across multiple name nodes.

- Helps to achieve Scalability

Fault Tolerance :

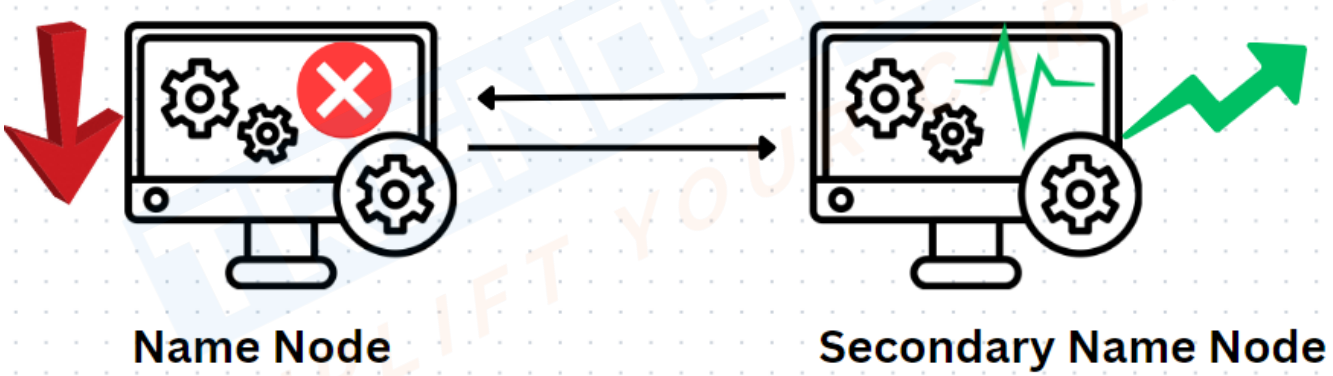
What if Data Node Fails?

Replication factor helps in having a backup stored on other Data Nodes and when the current data node goes down, the data can be accessed from the backup nodes.



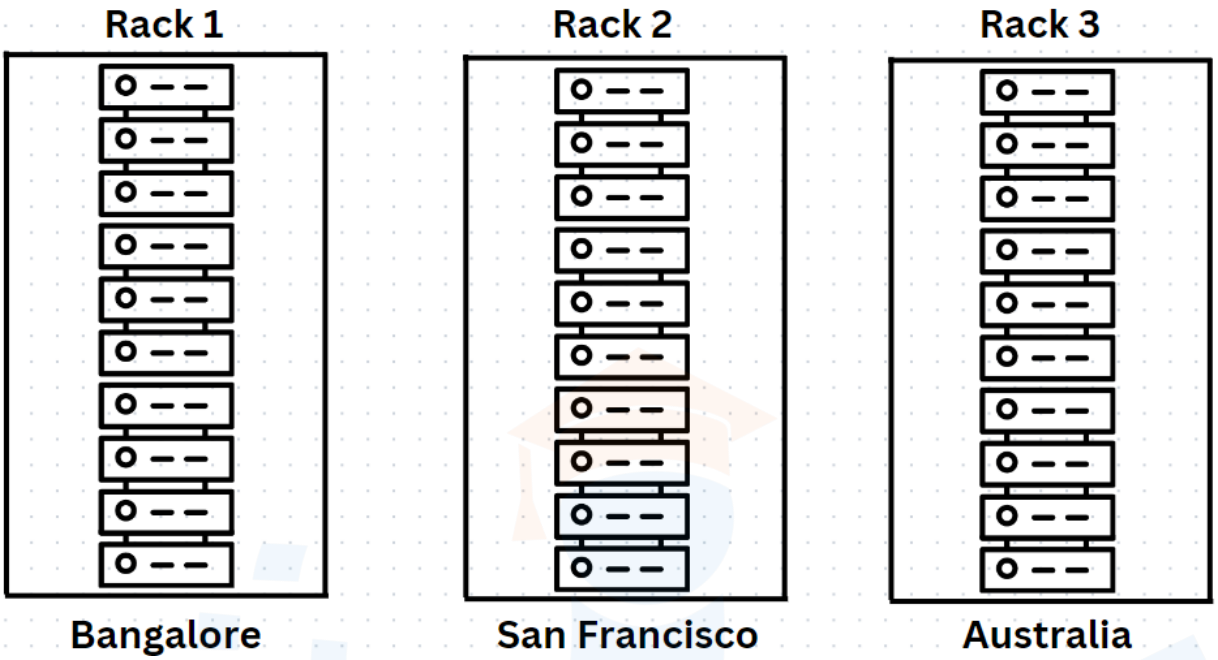
What if the Name Node Fails?

Secondary Name node will come into picture to keep the system up and running

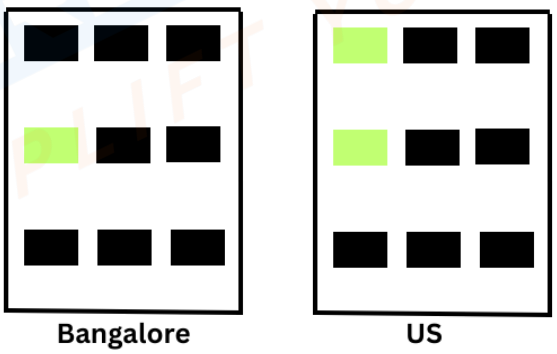


What is a Rack?

Rack is a group of servers in different geographical locations.



- ☐ It is the group of servers placed in different geographical locations.
- ☐ It is spread across different geographic locations, so that if there is a natural calamity , you don't lose your data.
- ☐ The balanced approach is to place replicas(copies) in two different racks, that is, one rack will have 1 copy and the other rack will have 2 copies of the data.



Practice Labs

Options for practising :

1. You can manually install all the hadoop services like Sqoop, Hive, Spark, Kafka, …

However, this is a very tedious and time consuming process that needs to be taken care of by Admins. Therefore, this option is ruled out from our practice environment list

2. **Pseudo Distributed Mode** - Ready to work Virtual Machines that have all the services preinstalled like Cloudera Quickstart VM and requires a Virtual box that manages the VM. This is called a Pseudo Distributed Setup where both the Name node and the Data node run on the same VM.

However, the system performance will be very slow as more than 50% of the system resources are allotted to the VM. Moreover, it doesn't mimic the production environment, therefore not a best option to practise

3. **Fully Distributed Mode** - A Multi-Node cluster by External third party lab provider. This is an ideal option to understand how things function in the production environment.

The different services that can be practised in the External Lab :

Linux Commands

HDFS Commands

Python

Data Structure and Algorithms

PySpark

Kafka

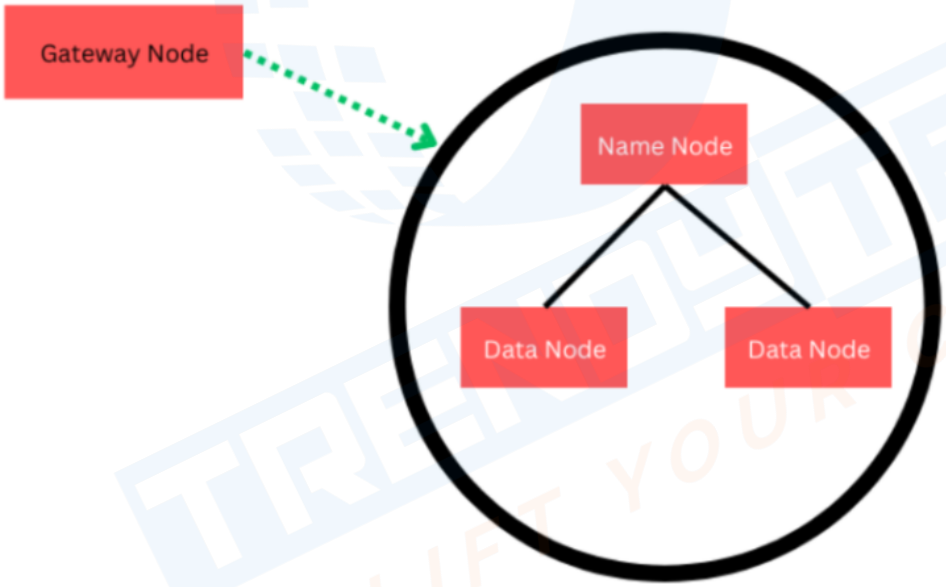
Structured Streaming

Hive

4. **Cloud Providers** - For all Cloud related services practice, you are required to create your own personal cloud accounts on the respective cloud providers like Azure | AWS

Process of logging in and accessing your External Lab account -

Users login to practise on the cluster through the Gateway node (acts as an entry to the cluster).



-We login to the gateway node (also called as edge node).This node has the connectivity to the Big Data cluster.

-Gateway node provides you an interface to talk to Big Data cluster. Users will be provided with the credentials of the edge node, so that they can login to the cluster.

User login credentials are provided by the third party external lab service providers. You will receive an email with the URL, Username and Password to login.

Logging in to the practice lab :

Gateway Node URL , Username & Password - You would have received an email with these details.

-After login in to the Gateway node with your login credentials, you can start your practice by opening a new terminal (File -> New Terminal)

HDFS vs non-HDFS

Let's assume you have a 10 TB hard disk in each of the three data nodes. Not all the 10 TB will be used for HDFS, some will be allocated for local as well.

10 TB of hard disk

3 data nodes = 30 TB

Each data node :

- 9 TB for HDFS
- 1 TB for non-HDFS / Local

So in total for all 3 data nodes

- for HDFS : 27 TB
- For non-HDFS : 3 TB

Linux Commands

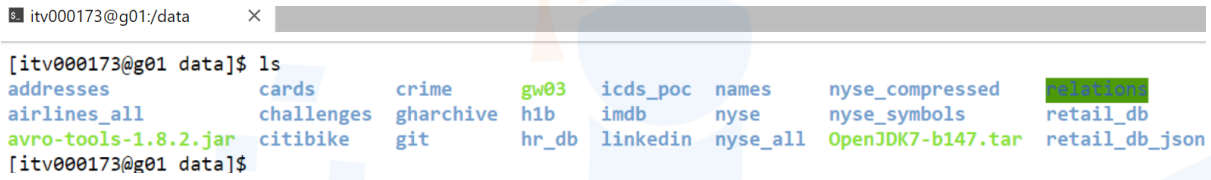
Linux follows a Tree structure -

“ / ” is the root directory of a base folder, considered to be the parent directory of all the directories. It is the topmost directory.

Here are some of the important Linux commands

cd : Change Directory is used to change the current directory and to move inside sub-directories.

ls : It is used to display the list of files and sub-directories in the current directory



In the above image,

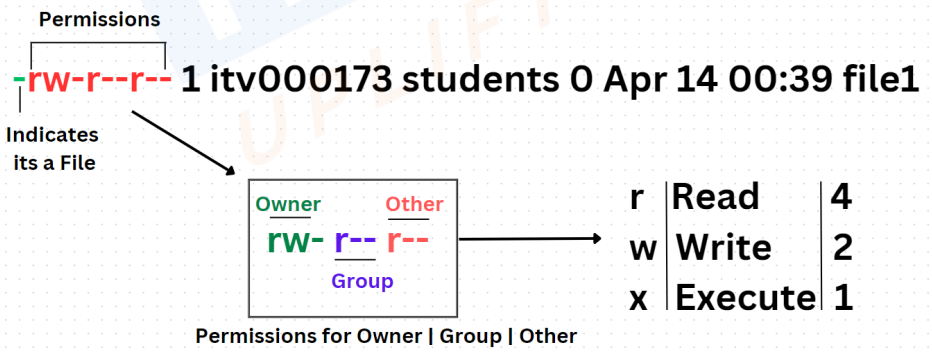
-blue represents directories

-green represents executables

-black denotes normal files

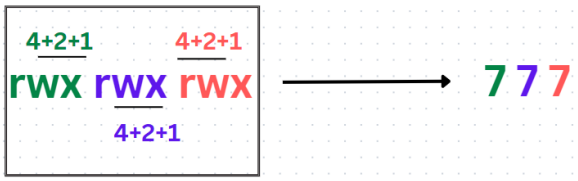
touch : Is used to create an empty file.

ls -ltr : to view the file created



chmod : Command to change the permissions

chmod 777 file1 - This command is used to change the permissions of a file, so that it is readable, writable, and executable (rwx) by all users. It will give all the permission to all as -rwxrwxrwx



mkdir : to create a new directory.

rmdir: deletes an empty directory. It will throw an error if the directory has some content in it

rm : command to delete files.

rm -R: to delete a directory, we need to remove it recursively.

cp : to copy files or directories.

mv : to move a file from one directory to another (Cut Paste**)** or to rename a file

vi : command to create a file with some content.

cat : command to see the contents of the file.

head : To display the first 10 lines of a file.

tail : To display the last 10 lines of a file.

du: It is used to display the amount of disk space used by files and directories.

grep : It searches for lines which contains a search pattern.

exit : To close the terminal window and end the execution of shell script.

HDFS Commands

Home Directory in local system: /home/username
Home Directory in HDFS: /user/username

HDFS commands are linux based commands. If you want to execute a Linux command on a distributed environment, all you have to do is prefix it with :

hadoop fs -

Or

hdfs dfs -

Some Examples of HDFS commands

hadoop fs -ls / : Lists all files are directories in the root directory of HDFS.

hadoop fs -mkdir -p /user/<username>/dir1/dir2 : Creating hierarchy of directories. It will create dir1, then inside dir1, it will create dir2.

Copying files/directories from one HDFS location to other HDFS location:

hadoop fs -cp <hdfs file path> <hdfs location>

hadoop fs -cp data/bigLog.txt datanew

Cut-Paste files/directories from one HDFS location to other HDFS location:

hadoop fs -mv <hdfs file path> <hdfs location>

hadoop fs -mv data/bigLog.txt newfolder

HDFS specific commands

Copying files/directories from local to HDFS:

put command:

hadoop fs -put <local file path> <hdfs file path>

hadoop fs -put /data/trendytech/bigLog.txt /user/itv005357/data

copyFromLocal command:

hadoop fs -copyFromLocal /data/trendytech/bigLog.txt /user/itv005357/data

Copying files/directories from HDFS to local:

get command:

hadoop fs -get <hdfs file path> <local file path>

hadoop fs -get /user/itv005357/data/bigLog.txt .

copyToLocal :

hadoop fs -copyToLocal <hdfs file path> <local file path>

hadoop fs -copyToLocal /user/itv005357/data/bigLog.txt .

HDFS Vs Cloud Data Lake

Cloud Alternatives of HDFS :

AWS – Amazon S3

Azure – ADLS Gen2 (azure data lake storage generation2)

Amazon S3 and ADLS Gen2 are data lakes in the cloud.

The Combinations which we are going to learn and practice in the course:

1. HDFS + Pyspark
2. ADLS gen2 + Databricks

Difference between HDFS and data lake in cloud:

HDFS – It is a distributed file system , and it stores data in block form.

ADLS Gen2 /amazon S3 – Is an object based storage.

Data is in the form of Object with the following associated parameters**:**
id – a unique identifier

Value – the content

Metadata – who can access the file, what kind of data is stored.

Distributed file system vs Object based Storage.

Or

HDFS vs ADLS gen2 /amazon S3

Some point to remember:

HDFS is not persistent but Amazon S3/ ADLS gen2 are persistent.

- What do you mean by Persistent??

  Say we have 4 node cluster

In HDFS

What if you shut down the cluster?

-Data will be lost.
-Since, storage is tightly coupled with compute. If you just want to increase storage, compute also increases as storage and compute are tightly coupled. Now, you will have to pay for compute also in HDFS.

-In hadoop we cannot access data from one HDFS cluster to another.

In ADLS gen2/ amazon S3

-Storage is not coupled with compute that is why they are cost effective.(decoupled compute and storage)

-In case of cloud like Amazon S3 / ADLS Gen2, any number of clusters can access the data.

HDFS	AMAZON S3 / ADLS GEN2
It is a distributed file system where data is internally stored as blocks.	It is object-based storage that stores data as "objects." Each object has a unique ID, contains the actual content or data, and has metadata which includes information such as who can access the file and what type of data it contains.
HDFS is not persistent, meaning that data is lost when the Hadoop cluster is shut down or terminated.	Amazon S3 and ADLS Gen2 are persistent, meaning that data is not lost when the cluster is terminated.
Data storage is tightly coupled with compute, meaning that you need to pay for compute resources even if you only need storage.	Storage is decoupled from compute, meaning that you need to only pay for the storage resources you use, without needing to pay for compute resources if you don't need them.
When using HDFS, if you have two Hadoop clusters, one with 4 nodes and another with 8 nodes, and you want the 8-node cluster to access data from the 4-node cluster, this is not possible. The data stored in HDFS is tied to the specific cluster it was stored on, so you cannot easily access it from another Hadoop cluster.	In Amazon S3 / ADLS Gen2, any number of clusters can access the same data.

Distributed Processing

MapReduce is a Programming Paradigm

MapReduce has 2 phases :

1. Map
2. Reduce

Principal of data locality

Is the data going to code or code is going to the data?? Code is generally small but data is big. So, the principle of data locality means the data is processed on the same machine where it is kept.

- The output of mapper is not the final output , but it is an intermediate output.
- The output of all the mappers go to one other machine (it can be one of the DN), where the reduce activity takes place.
- Mapper will give you parallelism.