BIG DATA AND HADOOP
CREDITS: 3

UNIT-2,3

Dr.M.Amsaprabhaa

Assistant Professor

Department of CSE

Shiv Nadar University Chennai

What is the difference between Hadoop and R?

- But without additional packages, it lacks a bit in terms of memory management and handling large data.
- Hadoop is a powerful tool to process and analyze large amounts of data with its distributed file system HDFS and the map-reduce processing approach.

Is Hadoop a programming language or not?

- The Hadoop framework itself is mostly written in the Java programming language, with some native code in C and command line utilities written as shell scripts.
- Though MapReduce Java code is common, any programming language can be used with Hadoop Streaming to implement the map and reduce parts of the user's program.

RHIPE ("R and Hadoop Integrated Programming Environment") is an R library that allows users to run Hadoop MapReduce jobs within R programming language. R programmers just have to write R map and R reduce functions and the RHIPE library will transfer them and invoke the corresponding Hadoop Map and Hadoop Reduce tasks.

STEPS TO USE HADOOP

Ubuntu:

1. https://www.virtualbox.org/wiki/Downloads
2. https://phoenixnap.com/kb/install-hadoop-ubuntu

Windows:
https://github.com/ruslanmv/How-to-install-Hadoop-on-Windows?tab=readme-ov-file

History of Hadoop

- **Apache Software Foundation** is the developers of Hadoop, and it's co-founders are **Doug Cutting** and **Mike Cafarella**. It's co-founder Doug Cutting named it on his son's toy elephant. In October 2003 the first paper release was Google File System. In January 2006, MapReduce development started on the Apache Nutch which consisted of around 6000 lines coding for it and around 5000 lines coding for HDFS. In April 2006 Hadoop 0.1.0 was released.
- Hadoop is an open-source software framework for storing and processing big data. It was created by Apache Software Foundation in 2006, based on a white paper written by Google in 2003 that described the Google File System (GFS) and the MapReduce programming model. The Hadoop framework allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. It is used by many organizations, including Yahoo, Facebook, and IBM, for a variety of purposes such as data warehousing, log processing, and research. Hadoop has been widely adopted in the industry and has become a key technology for big data processing.

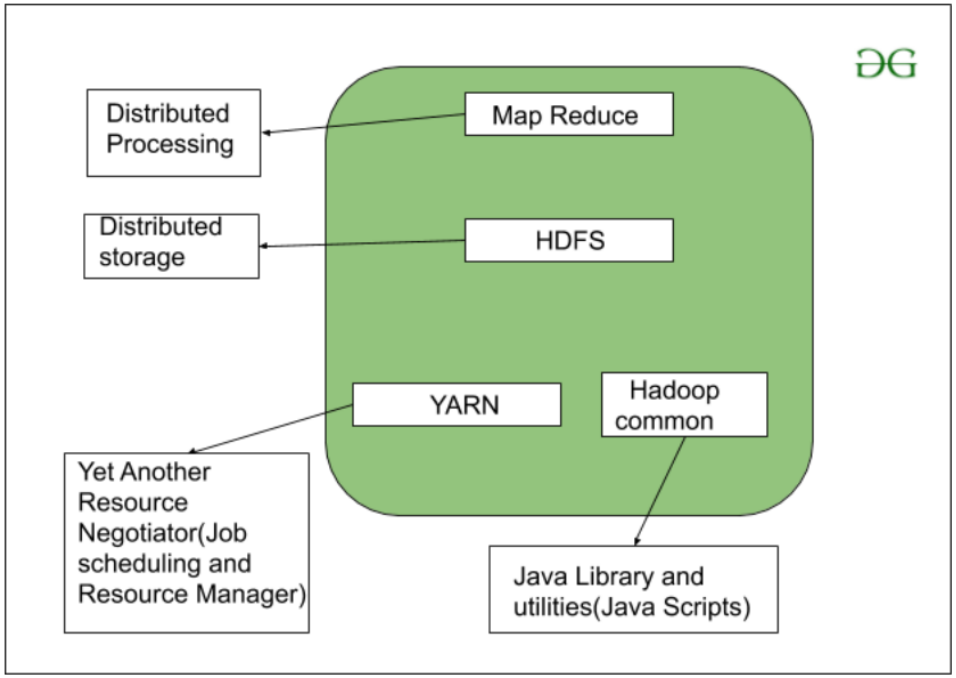
FEATURES OF HADOOP

- 1. It is fault tolerance. 2. It is highly available. 3. It's programming is easy.
- 4. It have huge flexible storage. 5. It is low cost.

Hadoop – Architecture

- Hadoop works on MapReduce Programming Algorithm that was introduced by Google.
- Today lots of Big Brand Companies are using Hadoop in their Organization to deal with big data,

  eg. Facebook, Yahoo, Netflix, eBay, etc.
- The Hadoop Architecture Mainly consists of 4 components.
  1. MapReduce
  2. HDFS(Hadoop Distributed File System)
  3. YARN(Yet Another Resource Negotiator)
  4. Common Utilities or Hadoop Common
-



INTRODUCTION

What is Hadoop?

- Hadoop is an open source software programming framework for storing a large amount of data and performing the computation.
- Its framework is based on Java programming with some native code in C and shell scripts.
- Hadoop is an open-source software framework that is used for storing and processing large amounts of data in a distributed computing environment.
- It is designed to handle big data and is based on the MapReduce programming model, which allows for the parallel processing of large datasets.

Hadoop has two main components:

1. ***HDFS (Hadoop Distributed File System):*** This is the storage component of Hadoop, which allows for the storage of large amounts of data across multiple machines. It is designed to work with commodity hardware, which makes it cost-effective.
2. ***YARN (Yet Another Resource Negotiator):*** This is the resource management component of Hadoop, which manages the allocation of resources (such as CPU and memory) for processing the data stored in HDFS.

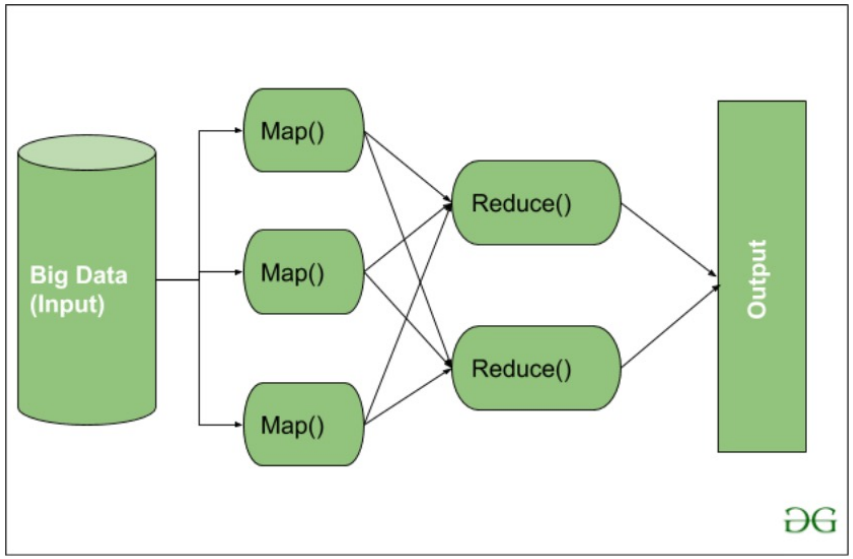
- Hadoop also includes several additional modules that provide additional functionality, such as Hive (a SQL-like query language), Pig (a high-level platform for creating MapReduce programs), and HBase (a non-relational, distributed database).
- Hadoop is commonly used in big data scenarios such as data warehousing, business intelligence, and machine learning.
- It's also used for data processing, data analysis, and data mining.
- It enables the distributed processing of large data sets across clusters of computers using a simple programming model.

Hadoop key features for big data processing

- Distributed Storage: Hadoop stores large data sets across multiple machines, allowing for the storage and processing of extremely large amounts of data.
- Scalability: Hadoop can scale from a single server to thousands of machines, making it easy to add more capacity as needed.
- Fault-Tolerance: Hadoop is designed to be highly fault-tolerant, meaning it can continue to operate even in the presence of hardware failures.
- Data locality: Hadoop provides data locality feature, where the data is stored on the same node where it will be processed, this feature helps to reduce the network traffic and improve the performance
- High Availability: Hadoop provides High Availability feature, which helps to make sure that the data is always available and is not lost.
- Flexible Data Processing: Hadoop's MapReduce programming model allows for the processing of data in a distributed fashion, making it easy to implement a wide variety of data processing tasks.
- Data Integrity: Hadoop provides built-in checksum feature, which helps to ensure that the data stored is consistent and correct.
- Data Replication: Hadoop provides data replication feature, which helps to replicate the data across the cluster for fault tolerance.
- Data Compression: Hadoop provides built-in data compression feature, which helps to reduce the storage space and improve the performance.
- YARN: A resource management platform that allows multiple data processing engines like real-time streaming, batch processing, and interactive SQL, to run and process data stored in HDFS.

MapReduce

- MapReduce nothing but just like an Algorithm or a data structure that is based on the YARN framework.
- The major feature of MapReduce is to perform the distributed processing in parallel in a Hadoop cluster which Makes Hadoop working so fast.
- When you are dealing with Big Data, serial processing is no more of any use.
- MapReduce has mainly 2 tasks which are divided phase-wise:
- In first phase, **Map** is utilized and in next phase **Reduce** is utilized.

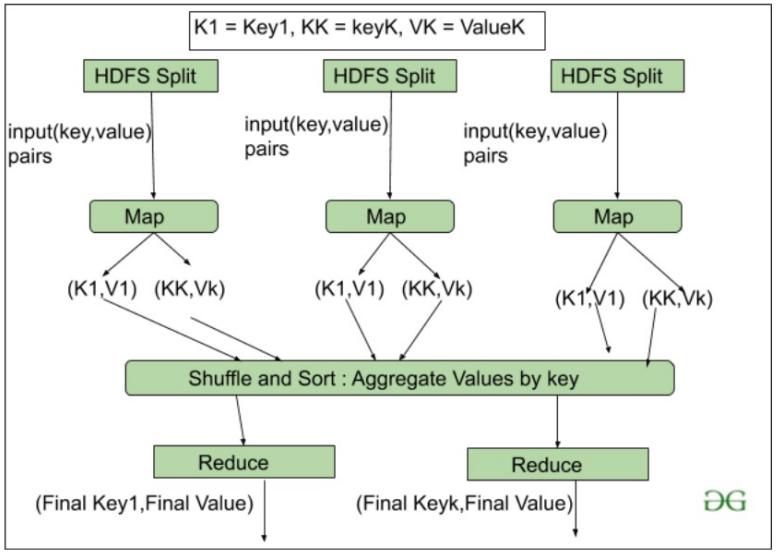


MapReduce

- The Input is a set of Data.
- The Map() function here breaks this DataBlocks into **Tuples** that are nothing but a key-value pair.
- These key-value pairs are now sent as input to the Reduce().
- The Reduce() function then combines this broken Tuples or key-value pair based on its Key value and form set of Tuples, and perform some operation like sorting, summation type job, etc. which is then sent to the final Output Node.
- Finally, the Output is Obtained.
- The data processing is always done in Reducer depending upon the business requirement of that industry.
- This is How First Map() and then Reduce is utilized one by one.

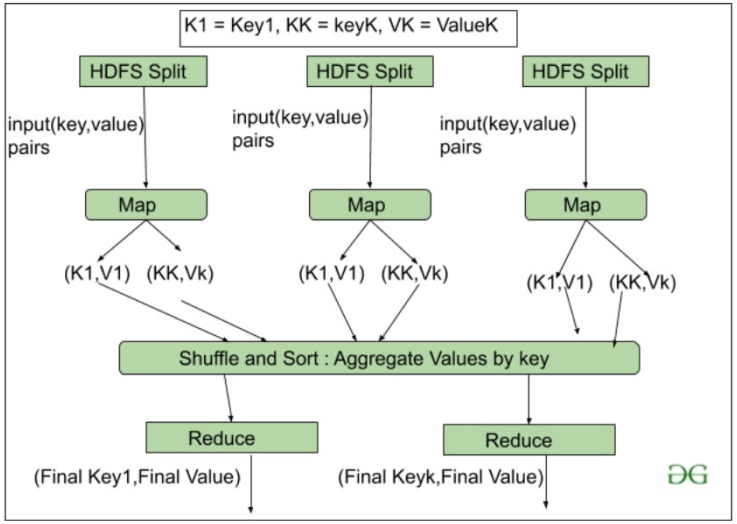
Map Task

- **RecordReader** The purpose of *recordreader* is to break the records. It is responsible for providing key-value pairs in a Map() function. The key is actually is its locational information and value is the data associated with it.
- **Map:** A map is nothing but a user-defined function whose work is to process the Tuples obtained from record reader. The Map() function either does not generate any key-value pair or generate multiple pairs of these tuples.
- **Combiner:** Combiner is used for grouping the data in the Map workflow. It is similar to a Local reducer. The intermediate key-value that are generated in the Map is combined with the help of this combiner. Using a combiner is not necessary as it is optional.
- **Partitionar:** Partitional is responsible for fetching key-value pairs generated in the Mapper Phases. The partitioner generates the shards corresponding to each reducer. Hashcode of each key is also fetched by this partition. Then partitioner performs it's(Hashcode) modulus with the number of reducers(*key.hashcode()%(number of reducers)*).



Reduce Task

- **Shuffle and Sort:** The Task of Reducer starts with this step, the process in which the Mapper generates the intermediate key-value and transfers them to the Reducer task is known as *Shuffling*. Using the Shuffling process the system can sort the data using its key value.
  Once some of the Mapping tasks are done Shuffling begins that is why it is a faster process and does not wait for the completion of the task performed by Mapper.
- **Reduce:** The main function or task of the Reduce is to gather the Tuple generated from Map and then perform some sorting and aggregation sort of process on those key-value depending on its key element.
- **OutputFormat:** Once all the operations are performed, the key-value pairs are written into the file with the help of record writer, each record in a new line, and the key and value in a space-separated manner.

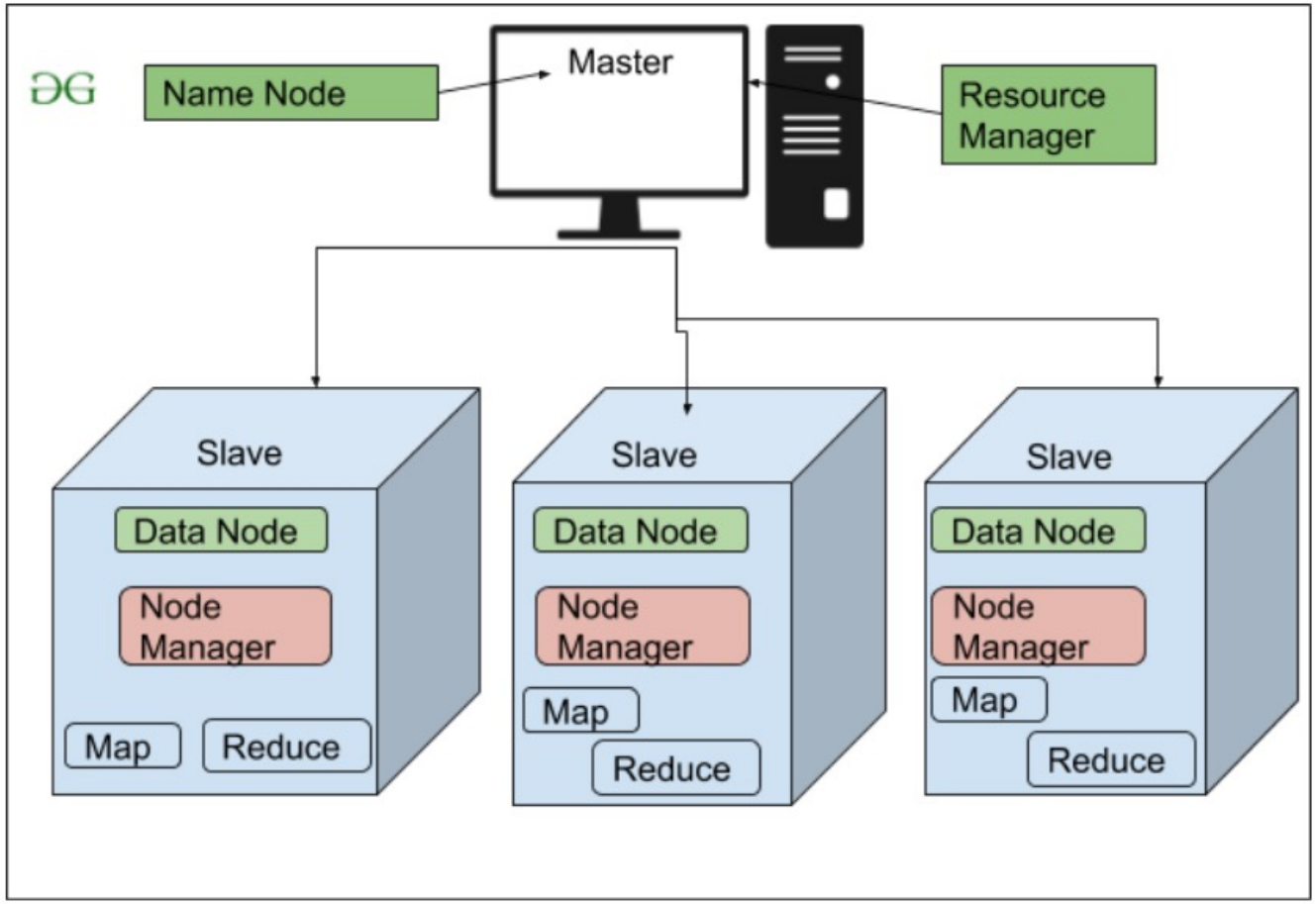


Hadoop Distributed File System

- HDFS(Hadoop Distributed File System) is utilized for storage permission.
- It is mainly designed for working on commodity Hardware devices(inexpensive devices), working on a distributed file system design.
- HDFS is designed in such a way that it believes more in storing the data in a large chunk of blocks rather than storing small data blocks.
- HDFS in Hadoop provides Fault-tolerance and High availability to the storage layer and the other devices present in that Hadoop cluster.
- Data storage Nodes in HDFS.
    1. *NameNode(Master)*
    2. *DataNode(Slave)*

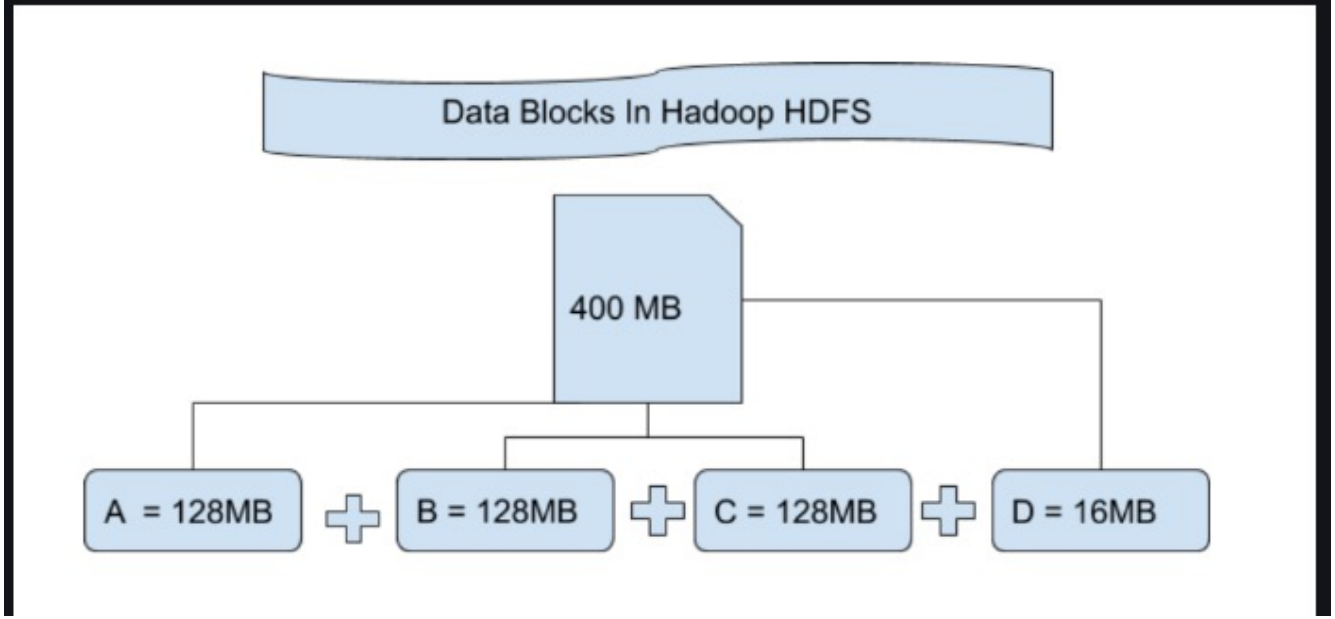
1. **NameNode:**NameNode works as a Master in a Hadoop cluster that guides the Datanode(Slaves). Namenode is mainly used for storing the Metadata i.e. the data about the data. Meta Data can be the transaction logs that keep track of the user's activity in a Hadoop cluster. Meta Data can also be the name of the file, size, and the information about the location(Block number, Block ids) of Datanode that Namenode stores to find the closest DataNode for Faster Communication. Namenode instructs the DataNodes with the operation like delete, create, Replicate, etc.
2. **DataNode:** DataNodes works as a Slave DataNodes are mainly utilized for storing the data in a Hadoop cluster, the number of DataNodes can be from 1 to 500 or even more than that. The more number of DataNode, the Hadoop cluster will be able to store more data. So it is advised that the DataNode should have High storing capacity to store a large number of file blocks.

High Level Architecture Of Hadoop



File Block In HDFS

File Block In HDFS: Data in HDFS is always stored in terms of blocks. So the single block of data is divided into multiple blocks of size 128MB which is default and you can also change it manually.



File Block In HDFS

Let's understand this concept of breaking down of file in blocks with an example. Suppose you have uploaded a file of 400MB to your HDFS then what happens is this file got divided into blocks of 128MB+128MB+128MB+16MB = 400MB size. Means 4 blocks are created each of 128MB except the last one. Hadoop doesn't know or it doesn't care about what data is stored in these blocks so it considers the final file blocks as a partial record as it does not have any idea regarding it. In the Linux file system, the size of a file block is about 4KB which is very much less than the default size of file blocks in the Hadoop file system. As we all know Hadoop is mainly configured for storing the large size data which is in petabyte, this is what makes Hadoop file system different from other file systems as it can be scaled, nowadays file blocks of 128MB to 256MB are considered in Hadoop.

File Block In HDFS

Replication In HDFS Replication ensures the availability of the data. Replication is making a copy of something and the number of times you make a copy of that particular thing can be expressed as it's Replication Factor. As we have seen in File blocks that the HDFS stores the data in the form of various blocks at the same time Hadoop is also configured to make a copy of those file blocks.

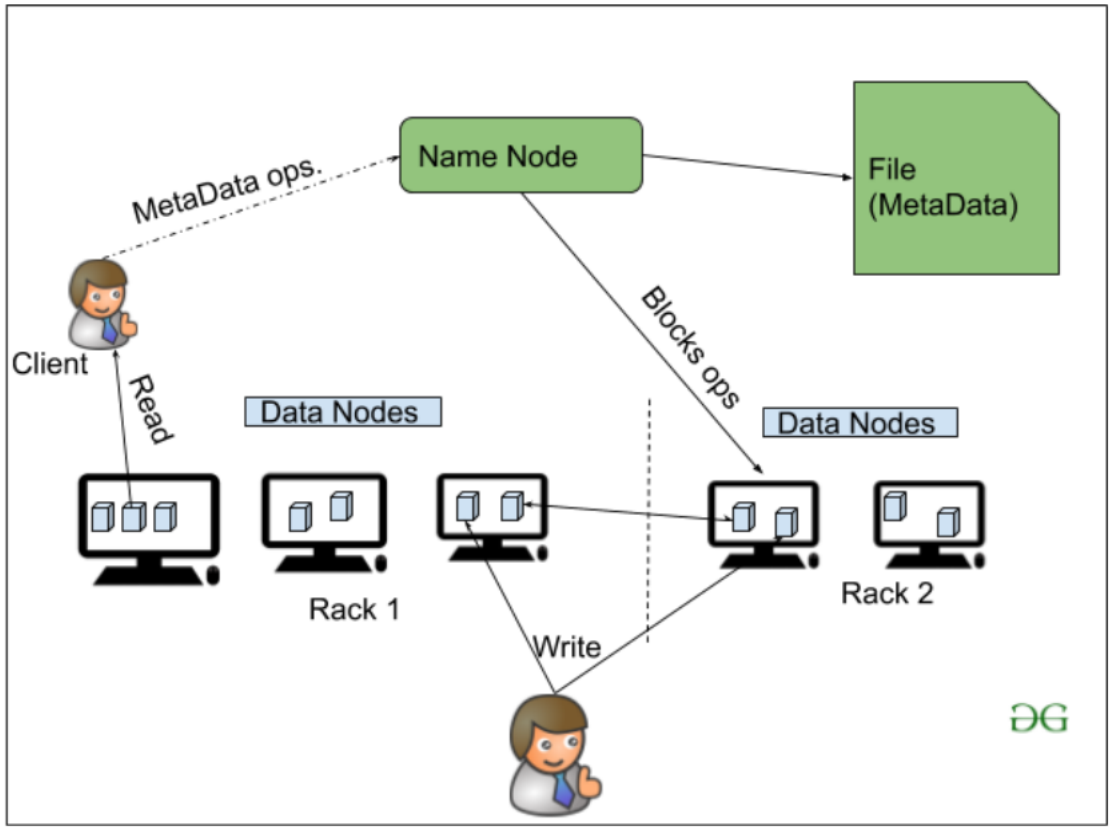
By default, the Replication Factor for Hadoop is set to 3 which can be configured means you can change it manually as per your requirement like in above example we have made 4 file blocks which means that 3 Replica or copy of each file block is made means total of 4×3 = 12 blocks are made for the backup purpose.

This is because for running Hadoop we are using commodity hardware (inexpensive system hardware) which can be crashed at any time. We are not using the supercomputer for our Hadoop setup. That is why we need such a feature in HDFS which can make copies of that file blocks for backup purposes, this is known as fault tolerance.

Now one thing we also need to notice that after making so many replica's of our file blocks we are wasting so much of our storage but for the big brand organization the data is very much important than the storage so nobody cares for this extra storage. You can configure the Replication factor in your *hdfs-site.xml* file.

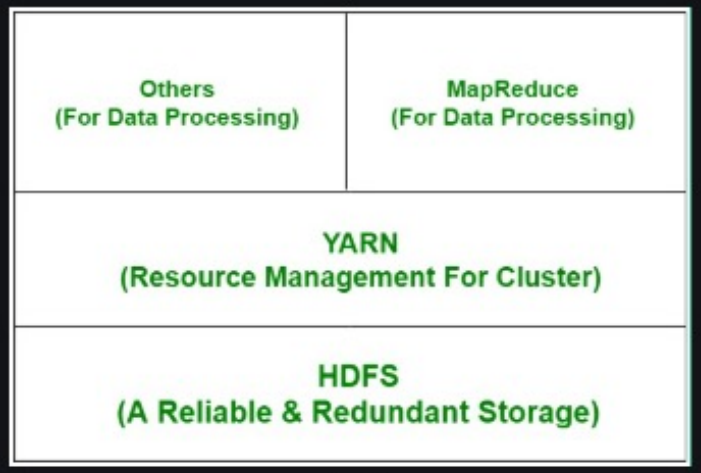
Rack Awareness The rack is nothing but just the physical collection of nodes in our Hadoop cluster (maybe 30 to 40). A large Hadoop cluster is consists of so many Racks . with the help of this Racks information Namenode chooses the closest Datanode to achieve the maximum performance while performing the read/write information which reduces the Network Traffic.

HDFS Architecture



Hadoop Distributed File System

It has distributed file system known as HDFS and this HDFS splits files into blocks and sends them across various nodes in form of large clusters. Also in case of a node failure, the system operates and data transfer takes place between the nodes which are facilitated by HDFS.



HDFS

Advantages of HDFS: It is inexpensive, immutable in nature, stores data reliably, ability to tolerate faults, scalable, block structured, can process a large amount of data simultaneously and many more. **Disadvantages of HDFS:** It's the biggest disadvantage is that it is not fit for small quantities of data. Also, it has issues related to potential stability, restrictive and rough in nature. Hadoop also supports a wide range of software packages such as Apache Flumes, Apache Oozie, Apache HBase, Apache Sqoop, Apache Spark, Apache Storm, Apache Pig, Apache Hive, Apache Phoenix, Cloudera Impala.

Hadoop Distributed File System

Some common frameworks of Hadoop

1. Hive- It uses HiveQl for data structuring and for writing complicated MapReduce in HDFS.
2. Drill- It consists of user-defined functions and is used for data exploration.
3. Storm- It allows real-time processing and streaming of data.
4. Spark- It contains a Machine Learning Library(MLlib) for providing enhanced machine learning and is widely used for data processing. It also supports Java, Python, and Scala.
5. Pig- It has Pig Latin, a SQL-Like language and performs data transformation of unstructured data.
6. Tez- It reduces the complexities of Hive and Pig and helps in the running of their codes faster.

Hadoop framework is made up of the following modules:

1. Hadoop MapReduce- a MapReduce programming model for handling and processing large data.
2. Hadoop Distributed File System- distributed files in clusters among nodes.
3. Hadoop YARN- a platform which manages computing resources.
4. Hadoop Common- it contains packages and libraries which are used for other modules.

Advantages of Hadoop

Advantages:

- Ability to store a large amount of data.
- High flexibility.
- Cost effective.
- High computational power.
- Tasks are independent.
- Linear scaling.

Hadoop has several advantages that make it a popular choice for big data processing:

- Scalability: Hadoop can easily scale to handle large amounts of data by adding more nodes to the cluster.
- Cost-effective: Hadoop is designed to work with commodity hardware, which makes it a cost-effective option for storing and processing large amounts of data.
- Fault-tolerance: Hadoop's distributed architecture provides built-in fault-tolerance, which means that if one node in the cluster goes down, the data can still be processed by the other nodes.
- Flexibility: Hadoop can process structured, semi-structured, and unstructured data, which makes it a versatile option for a wide range of big data scenarios.
- Open-source: Hadoop is open-source software, which means that it is free to use and modify. This also allows developers to access the source code and make improvements or add new features.
- Large community: Hadoop has a large and active community of developers and users who contribute to the development of the software, provide support, and share best practices.
- Integration: Hadoop is designed to work with other big data technologies such as Spark, Storm, and Flink, which allows for integration with a wide range of data processing and analysis tools.

Disadvantages of Hadoop

- Not very effective for small data.
- Hard cluster management.
- Has stability issues.
- Security concerns.
- Complexity: Hadoop can be complex to set up and maintain, especially for organizations without a dedicated team of experts.
- Latency: Hadoop is not well-suited for low-latency workloads and may not be the best choice for real-time data processing.
- Limited Support for Real-time Processing: Hadoop's batch-oriented nature makes it less suited for real-time streaming or interactive data processing use cases.
- Limited Support for Structured Data: Hadoop is designed to work with unstructured and semi-structured data, it is not well-suited for structured data processing
- Data Security: Hadoop does not provide built-in security features such as data encryption or user authentication, which can make it difficult to secure sensitive data.
- Limited Support for Ad-hoc Queries: Hadoop's MapReduce programming model is not well-suited for ad-hoc queries, making it difficult to perform exploratory data analysis.
- Limited Support for Graph and Machine Learning: Hadoop's core component HDFS and MapReduce are not well-suited for graph and machine learning workloads, specialized components like Apache Graph and Mahout are available but have some limitations.
- Cost: Hadoop can be expensive to set up and maintain, especially for organizations with large amounts of data.
- Data Loss: In the event of a hardware failure, the data stored in a single node may be lost permanently.
- Data Governance: Data Governance is a critical aspect of data management, Hadoop does not provide a built-in feature to manage data lineage, data quality, data cataloging, data lineage, and data audit.

YARN(Yet Another Resource Negotiator)

- YARN is a Framework on which MapReduce works. YARN performs 2 operations that are Job scheduling and Resource Management.
- The Purpose of Job schedular is to divide a big task into small jobs so that each job can be assigned to various slaves in a Hadoop cluster and Processing can be Maximized.
- Job Scheduler also keeps track of which job is important, which job has more priority, dependencies between the jobs and all the other information like job timing, etc.
- And the use of Resource Manager is to manage all the resources that are made available for running a Hadoop cluster.

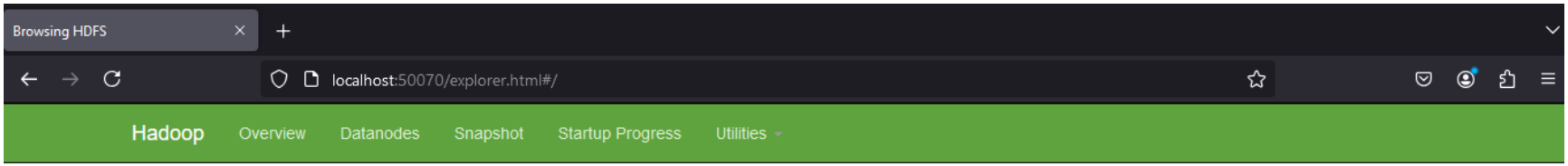
Features of YARN

- Multi-Tenancy
- Scalability
- Cluster-Utilization
- Compatibility

Hadoop common or Common Utilities

- Hadoop common or Common utilities are nothing but our java library and java files or we can say the java scripts that we need for all the other components present in a Hadoop cluster.
- These utilities are used by HDFS, YARN, and MapReduce for running the cluster.
- Hadoop Common verify that Hardware failure in a Hadoop cluster is common so it needs to be solved automatically in software by Hadoop Framework.

```
FAQ
1.What is a file system ?
Used to read and write from and to HD Ex NTFS , EXT , HDFS , S3 ..

2.What is block ? Big file in to small chunk Ex NTFS 16 K , EXT 512 K ..

3.Client and server ?
- Client send Request
- Server Response

4.Types of file system ?
- Standalone - NTFS , EXT
- Distributed - HDFS , S3 , CFS

5.Types of Distributed system ?
- Master and slave - Hadoop
- Peer to peer - Cassandra

6.What is process ? Program in execution

7.What is daemon process ? Background process

8.Cluster and node?
Node - Individual physical or virtual machine
Cluster - Group of node

9.What is client API ?
Application program interface
```



Browse Directory

/ Go!

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	Admin	supergroup	0 B	22/1/2025, 10:14:07 am	0	0 B	sample