**INTRODUCTION:**

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.

**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:**

* 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.
* 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.

**1. Key Components of Hadoop:**

Hadoop consists of four main components:

**a. Hadoop Distributed File System (HDFS)**:

* A distributed file system that stores large datasets by splitting them into blocks and distributing them across multiple nodes in a cluster.
* **Key features**:
  + **Fault-tolerant**: Data is replicated across multiple nodes to ensure reliability.
  + **Scalable**: Can handle massive amounts of data by adding more nodes to the cluster.
  + **High throughput**: Optimized for batch processing of large datasets.

**b. MapReduce**:

* A programming model for processing large datasets in parallel across a Hadoop cluster.
* **Map**: Splits the input data into key-value pairs, processes them independently, and generates intermediate key-value pairs.
* **Reduce**: Aggregates the intermediate key-value pairs to produce the final result.
* **Example**: Word count in a text document, where "Map" counts words in chunks, and "Reduce" aggregates the counts.

**c. YARN (Yet Another Resource Negotiator)**:

* YARN is Hadoop’s resource management layer that handles job scheduling and resource allocation across the cluster.
* **Key functions**:
  + Manages computing resources in clusters.
  + Schedules and monitors jobs (tasks) for execution.

**d. Hadoop Common**:

* A set of shared libraries and utilities that support the other Hadoop modules. These include essential tools like configuration files, Java libraries, and file management utilities.

**Hadoop Workflow:**

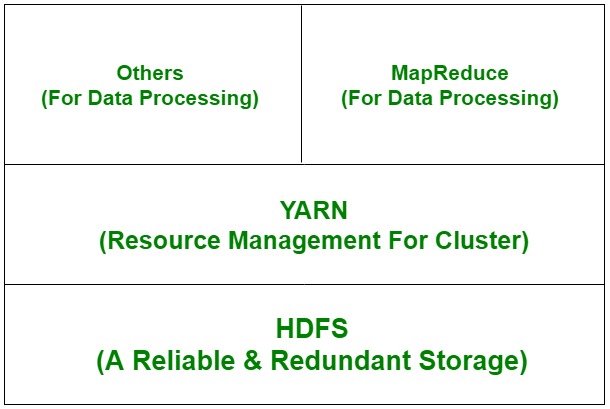
* Data is stored in **HDFS**.
* **MapReduce** jobs are used to process data in parallel across the cluster.
* **YARN** manages resources and scheduling tasks across the nodes.
* Data analysis or transformations are performed using the Hadoop ecosystem tools.

**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 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.

[](https://media.geeksforgeeks.org/wp-content/uploads/had.jpg)

*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.

**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.

# HDFS (Hadoop Distributed File System)

**HDFS** is the primary storage system of the Hadoop framework. It is designed to store very large datasets (terabytes or petabytes of data) across many machines in a large cluster, providing fault tolerance and high throughput access to data.

**1. Key Features of HDFS:**

* **Scalability**: Can scale to handle thousands of nodes and petabytes of data.
* **Fault Tolerance**: Data is replicated across multiple nodes, ensuring availability even if some nodes fail.
* **High Throughput**: Optimized for batch processing of large datasets rather than real-time access to small files.
* **Streaming Data Access**: HDFS is designed to support streaming data access, meaning it is optimized for high throughput rather than low-latency data access.

**2. HDFS Architecture:**

HDFS follows a **master-slave architecture** where the system is divided into:

**a. NameNode (Master Node)**:

* **Role**: Manages the metadata of the filesystem (i.e., the directory structure, file locations, blocks, replication factors).
* **Responsibilities**:
  + Keeps track of which blocks are stored on which DataNodes.
  + Coordinates file system operations like opening, closing, and renaming files or directories.
* **Single Point of Failure (SPoF)**: If the NameNode fails, the entire HDFS cluster becomes inaccessible, though high availability (HA) configurations with standby NameNodes can mitigate this.

**b. DataNodes (Slave Nodes)**:

* **Role**: Store the actual data in HDFS, broken down into blocks.
* **Responsibilities**:
  + Serve read/write requests from clients.
  + Periodically report back to the NameNode with information about stored blocks (block reports).
  + Perform block creation, deletion, and replication as instructed by the NameNode.

**3. Blocks in HDFS:**

* **Block Size**: HDFS splits large files into blocks, typically 128 MB or 256 MB in size.
* **Why Blocks?**:
  + Files are broken into blocks to distribute storage across multiple DataNodes.
  + Blocks are stored independently across the cluster, enabling parallel processing.
* **Block Replication**:
  + Each block is replicated across multiple DataNodes (default replication factor is 3).
  + Replication ensures fault tolerance—if one DataNode fails, the data is still available on other nodes.
  + The NameNode keeps track of which blocks reside on which DataNodes.

**4. HDFS Operations:**

**a. File Write**:

* A client requests the NameNode to write a file.
* The NameNode splits the file into blocks and assigns DataNodes to store the replicas.
* The client then writes each block to the assigned DataNodes in a pipeline fashion, ensuring that each block is replicated across the nodes.

**b. File Read**:

* A client requests the NameNode to read a file.
* The NameNode provides the client with the locations of the blocks.
* The client reads the blocks directly from the DataNodes.