## HADOOP:

1. Hadoop is an open-source software programming framework for storing a large amount of data and performing the computation.
2. Its framework is based on Java programming with some native code in C and shell scripts.
3. **Apache Software Foundation** is the developers of Hadoop, and its co-founders are Doug Cutting and Mike Cafarella.
4. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers.
5. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

## FEATURES:

1. Low Cost
2. High Computing Power
3. Scalability
4. Huge & Flexible Storage
5. Fault Tolerance & Data Protection

## HADOOP ARCHITECTURE:

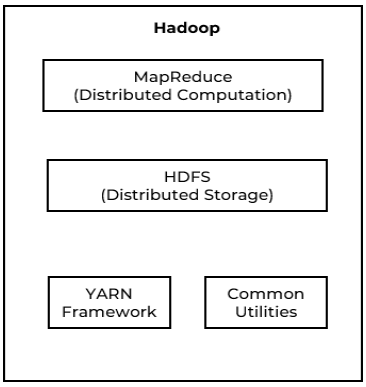



Figure : Hadoop Architecture

At its core, Hadoop has two major layers namely:

a. Processing/Computation layer (MapReduce), and
b. Storage layer (Hadoop Distributed File System).

## MapReduce:

1. MapReduce is a parallel programming model for writing distributed applications.
2. It is used for efficient processing of large amounts of data (multi-terabyte data-sets), on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.
3. The MapReduce program runs on Hadoop which is an Apache open-source framework.

**Hadoop Distributed File System:**

1. The Hadoop Distributed File System (HDFS) is based on the Google File System (GFS)
2. It provides a distributed file system that is designed to run on commodity hardware.
3. It has many similarities with existing distributed file systems.
4. However, the differences from other distributed file systems are significant.
5. It is highly fault-tolerant and is designed to be deployed on low-cost hardware.
6. It provides high throughput access to application data and is suitable for applications having large datasets.

**Hadoop framework also includes the following two modules:**

1. **Hadoop Common:** These are Java libraries and utilities required by other Hadoop modules.
2. **Hadoop YARN:** This is a framework for job scheduling and cluster resource management.

**ADVANTAGES:**

1. Ability to store a large amount of data.
2. High flexibility.
3. Cost effective.
4. High computational power.
5. Tasks are independent.
6. Linear scaling.

**DISADVANTAGES:**

1. Not very effective for small data.
2. Hard cluster management.
3. Has stability issues.
4. Security concerns.

**PHYSICAL ARCHITECTURE OF HADOOP:**

1. Hadoop is an open-source software framework which provides huge data storage.
2. Running Hadoop means running a set of resident programs.
3. These resident programs are also known as **daemons**.
4. These daemons may be running on the same server or on the different servers in the network.
5. Below figure shows Hadoop cluster topology.

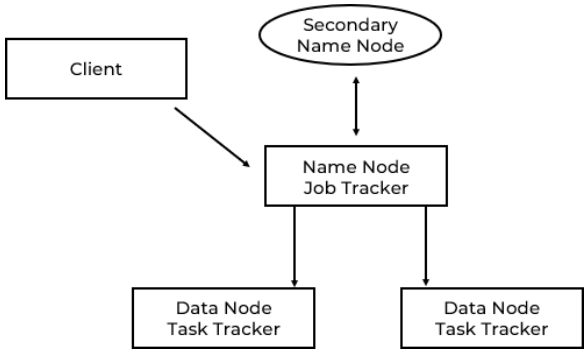



Figure 1.5: Hadoop Cluster Topology

**WORKING:**

1. When the client submits his job, it will go to the name node.
2. Now name node will decide whether to accept the job or not.
3. After accepting the job, the name node will transfer the job to the job tracker.
4. Then the job tracker will divide the job into components and transfer them to data nodes.
5. Now data nodes will further transfer the jobs to the task tracker.
6. Now the actual processing will be done here, means the execution of the job submitted is done here.
7. Now, after completing the part of the jobs assigned to them, the task tracker will submit the completed task to the job tracker via the data node.
8. Now, coming on secondary name node, the task of secondary name node is to just monitor the whole process ongoing.
9. Now, **physical architecture of Hadoop is a Master-slave process**, here name node is a master, job tracker is a part of master and data nodes are the slaves.

**COMPONENTS:**

**I) Name Node:**

1. It is the master of HDFS (Hadoop file system).
2. It contains Job Tracker, which keeps tracks of a file distributed to different data nodes.
3. Name Node directs Data Node regarding the low level I/O tasks.
4. Failure of Name Node will lead to the failure of the full Hadoop system.

**II) Data Node:**

1. Data node is the slave of HDFS.
2. A data node can communicate with each other through the name node to avoid replication in the provided task.
3. For replication of data a data node may communicates with other data nodes.
4. Data node continually informs local change updates to name nodes.
5. To create, move or delete blocks, data node receives instructions from the local disk.

**III) Job Tracker:**

1. Job Tracker determines which file to process.
2. There can be only one job tracker for per Hadoop cluster.
3. Job Tracker runs on a server as a master node of the cluster.

**IV) Task Tracker:**

1. Only single task tracker is present per slave node.
2. Task Tracker performs tasks given by job tracker and continuously communicates with the job tracker.

**V) SSN (Secondary Name Node):**

1. Its main purpose is to monitor.
2. State Monitoring of cluster HDFS is done by SNN.
3. SNN resides on its own machine also.
4. One SSN is present per cluster.

**CORE HADOOP COMPONENTS:**

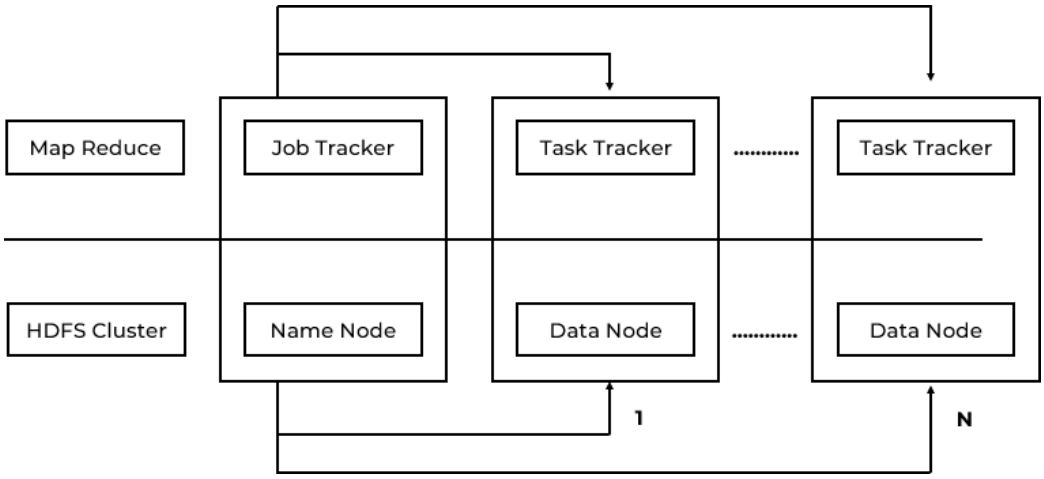



Figure 1.6: Hadoop Core Components

1. Hadoop has a master-slave topology.
2. In this topology, we have one master node and multiple slave nodes.
3. Master node's function is to assign a task to various slave nodes and manage resources. The slave nodes do the actual computing.
4. Slave nodes store the real data whereas on master we have metadata.
5. Figure 1.6 shows Hadoop core components.

**HADOOP DISTRIBUTED FILE SYSTEM (HDFS):**

1. HDFS is a file system for Hadoop.
2. HDFS is based on Google File System (GFS).

3. It runs on clusters on commodity hardware.
4. The file system has several similarities with the existing distributed file systems.

**Characteristics:**

1. High Fault Tolerant.
2. High throughput.
3. Supports application with massive datasets.
4. Streaming access to file system data.
5. Can be built out of commodity hardware.

**HDFS Architecture.**

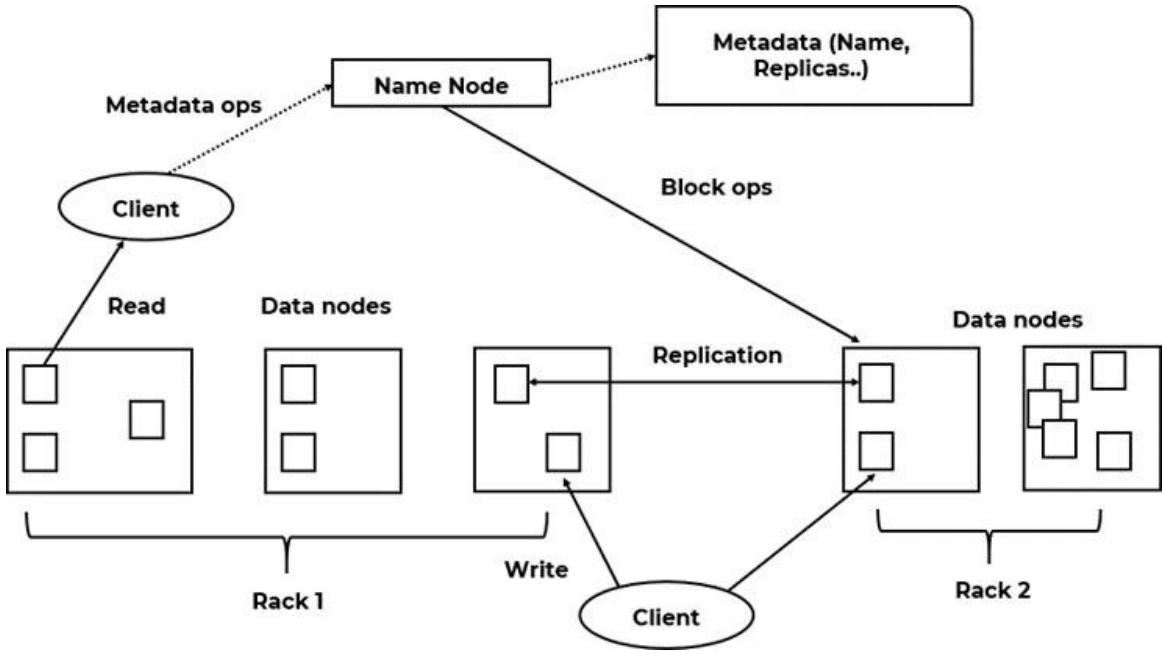



Figure 1.7: HDFS Architecture

HDFS follows the master-slave architecture, and it has the following elements.

**I) Namenode:**

1. It is a deamon which runs on master node of hadoop cluster.
2. There is only one namenode in a cluster.
3. It contains metadata of all the files stored on HDFS which is known as namespace of HDFS.
4. It maintains two files i.e., Edit Log & FsImage.
5. EditLog is used to record every change that occurs to file system metadata (transaction history)
6. FsImage stores entire namespace, mapping of blocks to files and file system properties.
7. The FsImage and the EditLog are central data structures of HDFS.
8. The system having the namenode acts as the master server and it does the following tasks:
   a. Manages the file system namespace.
   b. Regulates client's access to files.

c. It also executes file system operations such as renaming, closing, and opening files and directories.

**II) Datanode:**

1. It is a deamon which runs on slave machines of Hadoop cluster.
2. There are number of datanodes in a cluster.
3. It is responsible for serving read/write request from the clients. It also performs block creation, deletion, and replication upon instruction from the Namenode.
4. It also sends a Heartbeat message to the namenode periodically about the blocks it hold.
5. Namenode and Datanode machines typically run a GNU/Linux operating system (OS).

**III) Block:**

1. Generally, the user data is stored in the files of HDFS.
2. The file in a file system will be divided into one or more segments and/or stored in individual data nodes.
3. These file segments are called as blocks.
4. In other words, the minimum amount of data that HDFS can read or write is called a Block.
5. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

**MAPREDUCE:**

1. MapReduce is a **software framework**.
2. MapReduce is the **data processing layer of Hadoop**.
3. It is a software framework that allows you to write applications for processing a large amount of data.
4. MapReduce runs these applications in parallel on a cluster of low-end machines.
5. It does so in a reliable and fault-tolerant manner.
6. In MapReduce an application is broken down into number of small parts.
7. These small parts are also called as fragments or blocks.
8. These blocks then can be run on any node in the cluster.
9. Data Processing is done by MapReduce.
10. MapReduce scales and runs an application to different clutter machines.
11. There are two primitives used for data processing by MapReduce known as Mappers & Reducers.
12. MapReduce use lists and key/value pairs for processing of data.

**MapReduce Core Functions:**

**I) Read Input:**

1. It divides input into small blocks.
2. These blocks then get assigned to a Map function.

**II) Function Mapping:**

1. It converts file data to smaller, intermediate <key, value> pairs.

**III) Partition, Compare & Sort:**

a. Partition Function: With the given key and number of reducers it finds the correct reducer.

b. Compare Function: Map intermediate outputs are sorted according to this compare function.

**IV) Function Reducing:**

1. Intermediate values are reduced to smaller solutions and given to output.

**V) Write Output:**

1. Gives file output

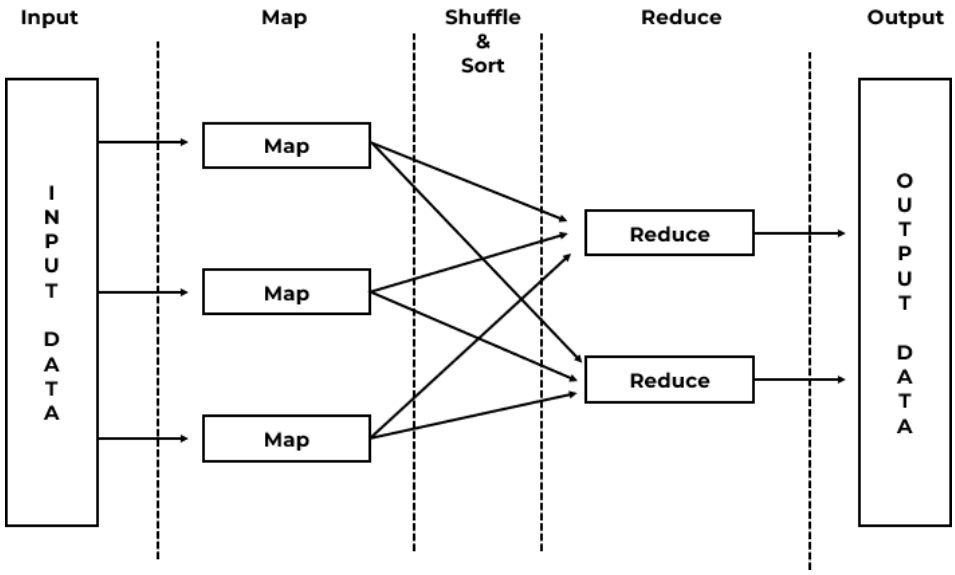



Figure 1.8: General MapReduce DataFlow

**Example:**

File 1: "Hello Babita Hello Jethalal"

File 2: "Goodnight Babita Goodnight Jethalal"

**Operations:**

(1) Map:

| Map 1 | Map 2 |
|---|---|
| <Hello, 1> | <Goodnight, 1> |
| <Babita, 1> | <Babita, 1> |
| <Hello, 1> | <Goodnight, 1> |
| <Jethalal, 1> | <Jethalal, 1> |

(2) Combine:

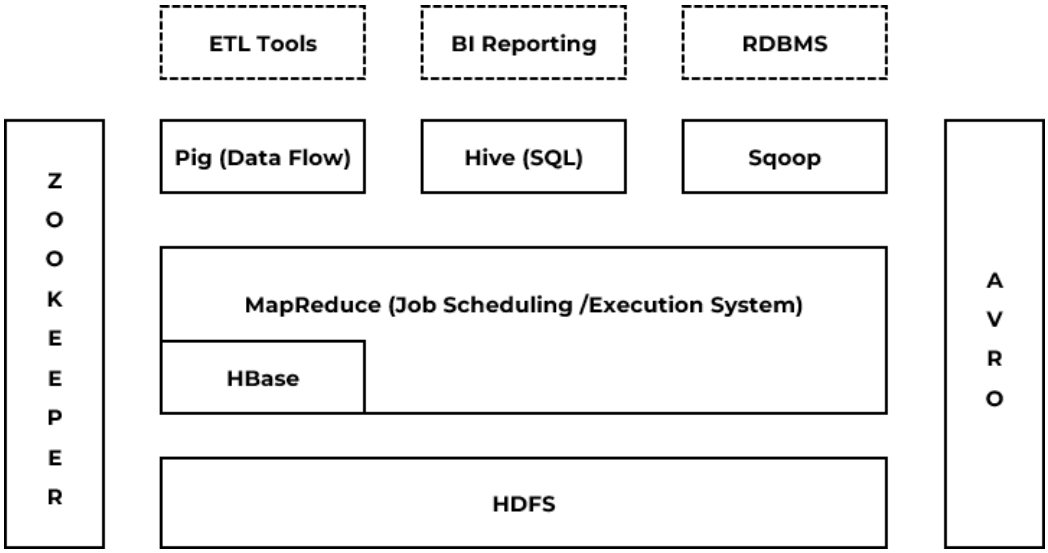
| Map 1 | Map 2 |
|---|---|
| <Babita, 1> | <Babita, 1> |
| <Jethalal, 1> | <Jethalal, 1> |
| <Hello, 2> | <Goodnight, 2> |

(3) Reduce:

<Babita, 2>
<Jethalal, 2>
<Hello, 2>
<Goodnight, 2>

**HADOOP ECOSYSTEM:**

1. Core Hadoop ecosystem is nothing but the different components that are built on the Hadoop platform directly.




Hadoop Ecosystem.

**I) Hadoop Distributed File System (HDFS):**

1. HDFS is the foundation of Hadoop and hence is a very important component of the Hadoop ecosystem.
2. It is Java software that provides many features like scalability, high availability, fault tolerance, cost effectiveness etc.
3. It also provides robust distributed data storage for Hadoop.
4. We can deploy many other software frameworks over HDFS.

**II) MapReduce:**

1. MapReduce is the data processing component of Hadoop.
2. It applies the computation on sets of data in parallel thereby improving the performance.
3. MapReduce works in two phases:
   a. **Map Phase:** This phase takes input as key-value pairs and produces output as key-value pairs. It can write custom business logic in this phase. Map phase processes the data and gives it to the next phase.
   b. **Reduce Phase:** The MapReduce framework sorts the key-value pair before giving the data to this phase. This phase applies the summary type of calculations to the key-value pairs.

**III) Hive:**

1. Hive is a data warehouse project built on the top of Apache Hadoop which provides data query and analysis.
2. It has got the language of its own call **HQL or Hive Query Language**.
3. HQL automatically translates the queries into the corresponding map-reduce job.
4. Main parts of the Hive are –
   a. **MetaStore:** It stores metadata
   b. **Driver:** Manages the lifecycle of HQL statement
   c. **Query Compiler:** Compiles HQL into DAG i.e. Directed Acyclic Graph
   d. **Hive Server:** Provides interface for JDBC/ODBC server.

**IV) Pig:**

1. Pig is a SQL like language used for querying and analyzing data stored in HDFS.
2. Yahoo was the original creator of the Pig.
3. It uses pig latin language.
4. It loads the data, applies a filter to it and dumps the data in the required format.
5. Pig also consists of JVM called Pig Runtime. Various features of Pig are as follows:-
   a. **Extensibility:** For carrying out special purpose processing, users can create their own custom function.
   b. **Optimization opportunities:** Pig automatically optimizes the query allowing users to focus on semantics rather than efficiency.
   c. **Handles all kinds of data:** Pig analyzes both structured as well as unstructured.

**V) HBase:**

1. HBase is a NoSQL database built on the top of HDFS.
2. The various features of HBase are that it is open-source, non-relational, distributed database.
3. It imitates **Google's Bigtable** and written in Java.
4. It provides real-time read/write access to large datasets.

**VI) Zookeeper:**

1. Zookeeper coordinates between various services in the Hadoop ecosystem.
2. It saves the time required for synchronization, configuration maintenance, grouping, and naming.
3. Following are the features of Zookeeper:
   a. **Speed:** Zookeeper is fast in workloads where reads to data are more than write. A typical read: write ratio is 10:1.
   b. **Organized:** Zookeeper maintains a record of all transactions.
   c. **Simple:** It maintains a single hierarchical namespace, similar to directories and files.
   d. **Reliable:** We can replicate Zookeeper over a set of hosts, and they are aware of each other. There is no single point of failure. If major servers are available zookeeper is available.

**VII) Sqoop:**

1. Sqoop imports data from external sources into compatible Hadoop Ecosystem components like HDFS, Hive, HBase etc.
2. It also transfers data from Hadoop to other external sources.
3. It works with RDBMS like TeraData, Oracle, MySQL and so on.
4. The major difference between Sqoop and Flume is that Flume does not work with structured data.
5. But Sqoop can deal with structured as well as unstructured data.