# SNS COLLEGE OF TECHNOLOGY

## COIMBATORE

### AN AUTONOMOUS INSTITUTION

Accredited by NBA – AICTE and Accredited by NAAC – UGC with ‘A’ Grade

Approved by AICTE New Delhi & affiliated to the Anna University, Chennai

## DEPARTMENT OF MCA

**Course Name : 16CAT702 - BIG DATA ANALYTICS**

**Class : II Year / III Semester**

**Unit II - Introduction**

**Topic I – History of Hadoop - The Hadoop Distributed File System**

# Hadoop - History or Evolution

Hadoop is an open source framework overseen by Apache Software Foundation which is written in Java for storing and processing of huge datasets with the cluster of commodity hardware.

There are mainly two problems with the big data.

- First one is to store such a huge amount of data
- Second one is to process that stored data

The traditional approach like RDBMS is not sufficient due to the heterogeneity of the data.

# Hadoop - History or Evolution

Hadoop comes as the solution to the problem of big data i.e. storing and processing the big data with some extra capabilities.

There are mainly two components of Hadoop which are

- **Hadoop Distributed File System (HDFS)** and
- **Yet Another Resource Negotiator(YARN)**.

# Hadoop History

- Hadoop was started with **Doug Cutting and Mike Cafarella** in the year 2002 when they both started to work on Apache Nutch project.
- Apache Nutch project was the process of building a search engine system that can index 1 billion pages.
- After a lot of research on Nutch, they concluded that such a system will cost around half a million dollars in hardware, and along with a monthly running cost of $30, 000 approximately, which is very expensive.

# Hadoop History

- So, they realized that their project architecture will not be capable enough to the workaround with billions of pages on the web.
- So they were looking for a feasible solution which can reduce the implementation cost as well as the problem of storing and processing of large datasets.
- **In 2003,** they came across a paper that described the architecture of Google's distributed file system, called **GFS (Google File System)** which was published by Google, for storing the large data sets.

# Hadoop History

- Now they realize that this paper can solve their problem of storing very large files which were being generated because of web crawling and indexing processes. But this paper was just the half solution to their problem.
- **In 2004,** Google published one more paper on the technique **MapReduce**, which was the solution of processing those large datasets.
- Now this paper was another half solution for Doug Cutting and Mike Cafarella for their Nutch project.

# Hadoop History

- Now they realize that this paper can solve their problem of storing very large files which were being generated because of web crawling and indexing processes. But this paper was just the half solution to their problem.
- **In 2004,** Google published one more paper on the technique **MapReduce**, which was the solution of processing those large datasets.
- Now this paper was another half solution for Doug Cutting and Mike Cafarella for their Nutch project.

# Hadoop History

- These both techniques (GFS & MapReduce) were just on white paper at Google. Google didn't implement these two techniques.
- Doug Cutting knew from his work on Apache Lucene ( It is a free and open-source information retrieval software library, originally written in Java by Doug Cutting in 1999) that open-source is a great way to spread the technology to more people.
- So, together with Mike Cafarella, he started implementing Google's techniques (GFS & MapReduce) as open-source in the Apache Nutch project.

# Hadoop History

- **In 2005,** Cutting found that Nutch is limited to only 20-to-40 node clusters. He soon realized two problems:
- **(a)** Nutch wouldn't achieve its potential until it ran reliably on the larger clusters

**(b)** And that was looking impossible with just two people (Doug Cutting & Mike Cafarella).

The engineering task in Nutch project was much bigger than he realized. So he started to find a job with a company who is interested in investing in their efforts. And he found Yahoo!.Yahoo had a large team of engineers that was eager to work on this there project.

# Hadoop History

- So **in 2006,** Doug Cutting joined Yahoo along with Nutch project. He wanted to provide the world with an open-source, reliable, scalable computing framework, with the help of Yahoo.

- So at Yahoo first, he separates the distributed computing parts from Nutch and **formed a new project Hadoop (He gave name Hadoop it was the name of a yellow toy elephant which was owned by the Doug Cutting's son. and it was easy to pronounce and was the unique word.**) Now he wanted to make Hadoop in such a way that it can work well on thousands of nodes. So with GFS and MapReduce, he started to work on Hadoop.

# Hadoop History

- So **in 2006,** Doug Cutting joined Yahoo along with Nutch project. He wanted to provide the world with an open-source, reliable, scalable computing framework, with the help of Yahoo.

- So at Yahoo first, he separates the distributed computing parts from Nutch and **formed a new project Hadoop (He gave name Hadoop it was the name of a yellow toy elephant which was owned by the Doug Cutting's son. and it was easy to pronounce and was the unique word.**) Now he wanted to make Hadoop in such a way that it can work well on thousands of nodes. So with GFS and MapReduce, he started to work on Hadoop.

# Hadoop History

**In 2007,** Yahoo successfully tested Hadoop on a 1000 node cluster and start using it.

In January of 2008, Yahoo released Hadoop as an open source project to **ASF(Apache Software Foundation**). And in July of 2008, Apache Software Foundation successfully tested a 4000 node cluster with Hadoop.

# Hadoop History

**In 2009,** Hadoop was successfully tested to sort a PB (PetaByte) of data in less than 17 hours for handling billions of searches and indexing millions of web pages.

And **Doug Cutting left the Yahoo and joined Cloudera to fulfill the challenge of spreading Hadoop to other industries.**

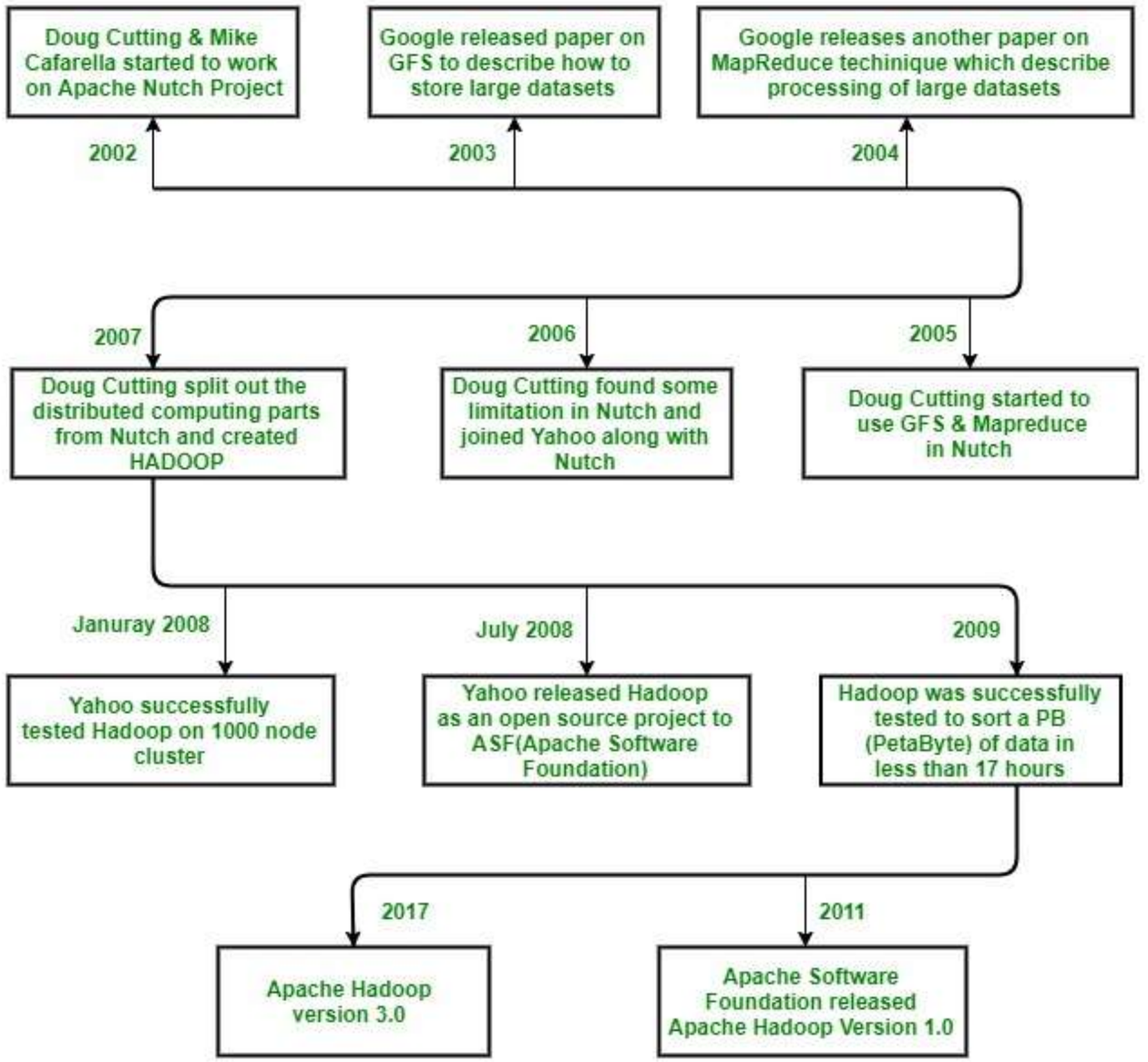
In **December of 2011,** Apache Software Foundation released **Apache Hadoop version 1.0.**

And later in Aug 2013, **Version 2.0.6 was available**.

And currently, we have **Apache Hadoop version 3.0** which released in **December 2017**.

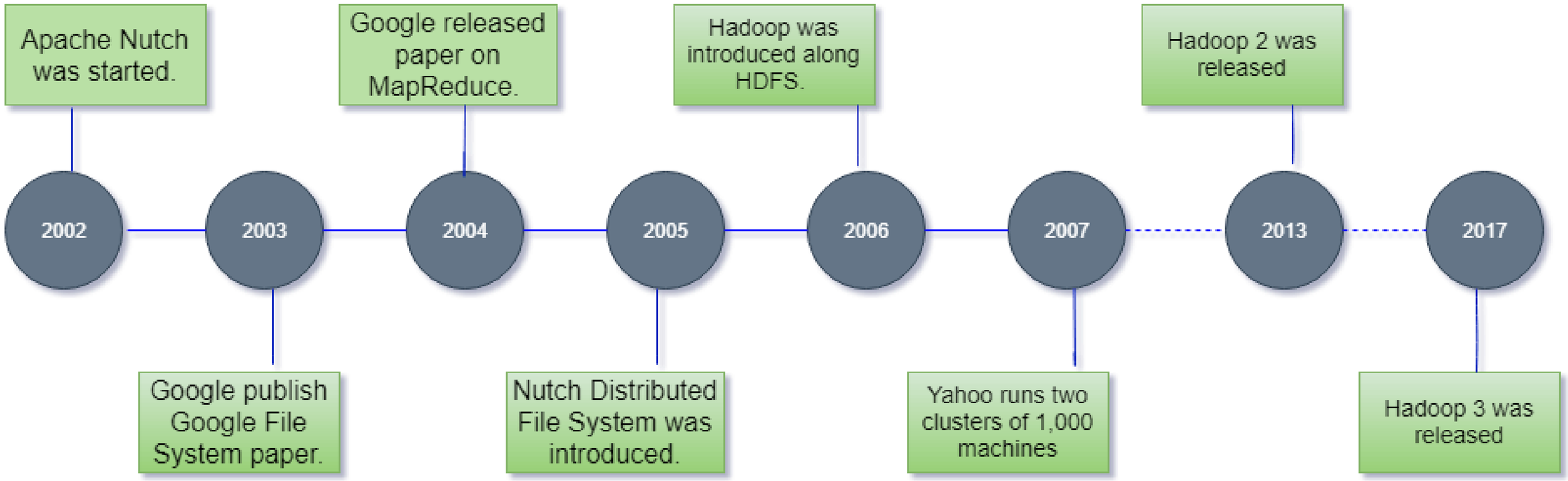
# Hadoop History

## Hadoop



# History of Hadoop

The Hadoop was started by Doug Cutting and Mike Cafarella in 2002. Its origin was the Google File System paper, published by Google.



# What is Hadoop

What is the need of Hadoop?

It emerged as a solution to the “**Big Data**” problems

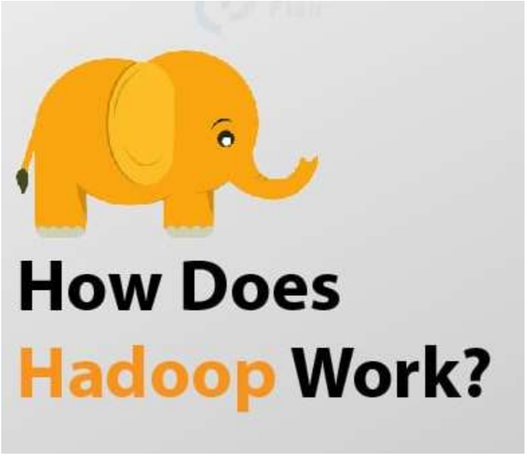
**a. Storage for Big Data –** HDFS Solved this problem. It stores Big Data in Distributed Manner. HDFS also stores each file as blocks. Block is the smallest unit of data in a filesystem.

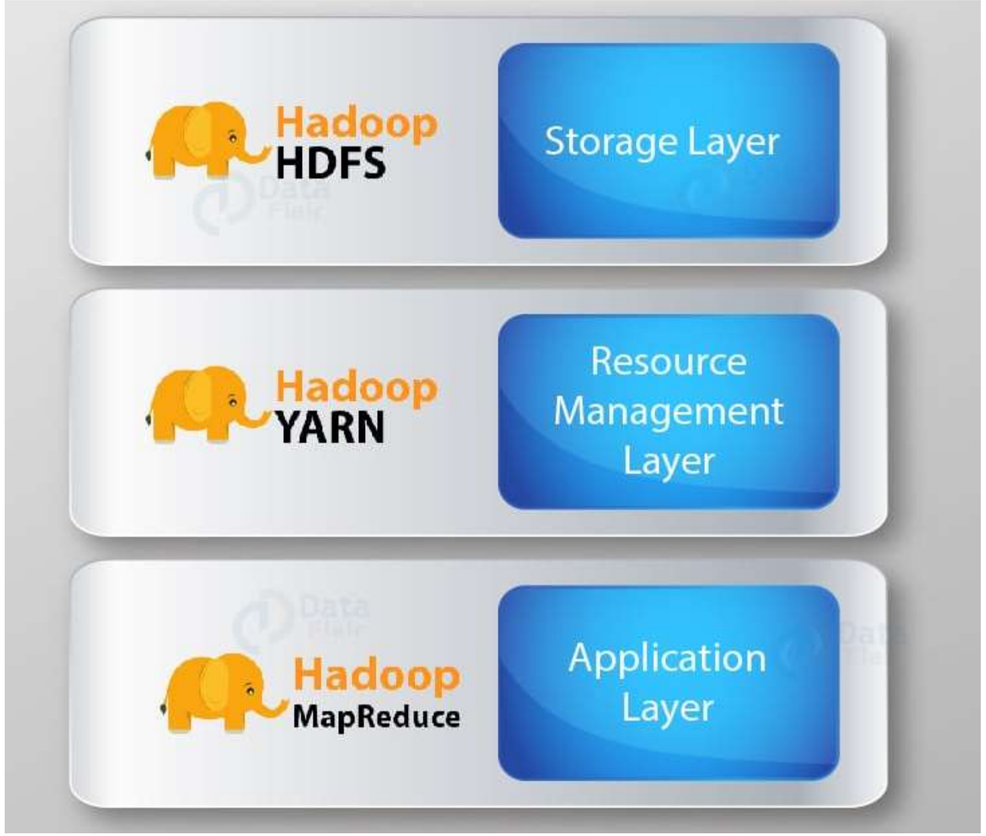
Suppose you have 512MB of data. And you have configured HDFS such that it will create 128Mb of data blocks. So HDFS divide data into 4 **blocks** (512/128=4) and stores it across different DataNodes. It also replicates the data blocks on different datanodes.

Hence, storing big data is not a challenge.

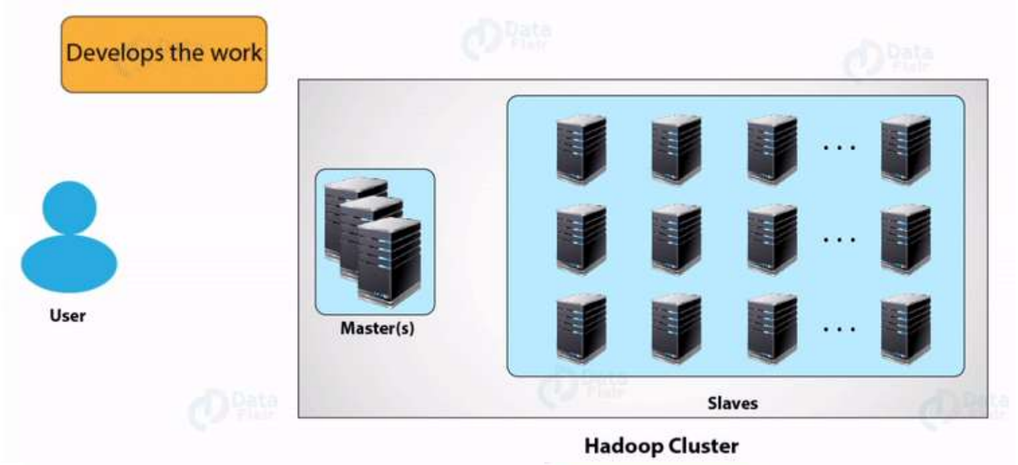
**b. Scalability –** It also solves the Scaling problem. It mainly focuses on horizontal scaling rather than vertical scaling. You can add extra datanodes to HDFS cluster as and when required. Instead of scaling up the resources of your datanodes.

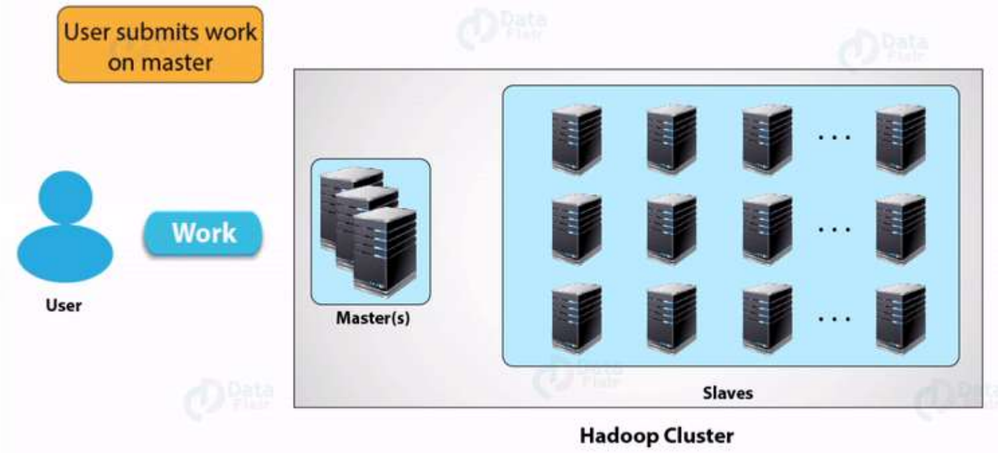
# What is Hadoop

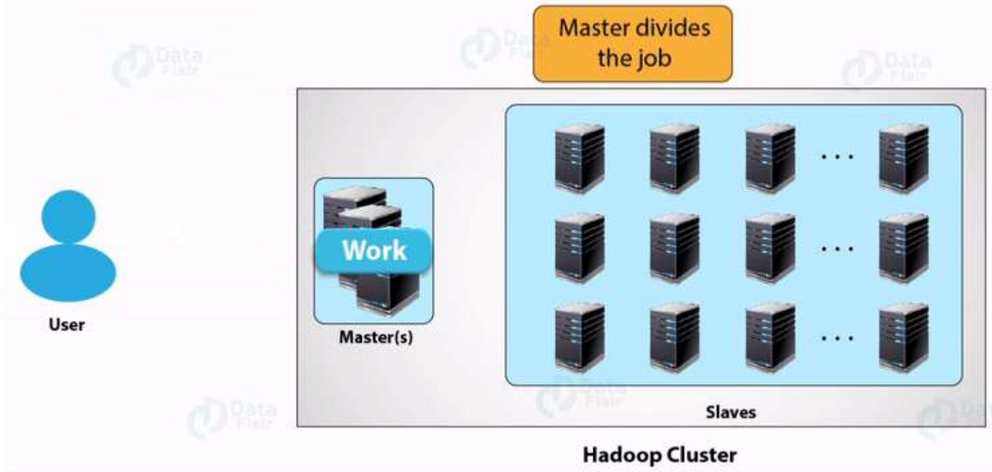


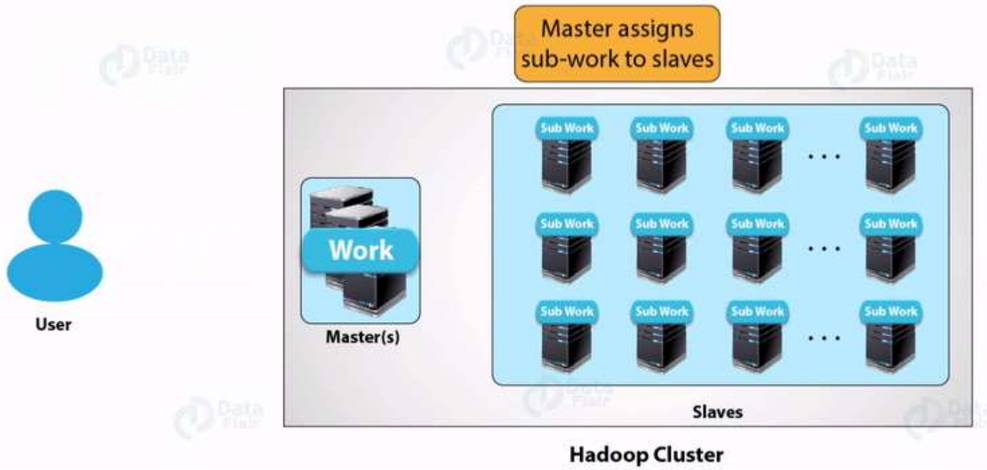



# Hadoop Architecture

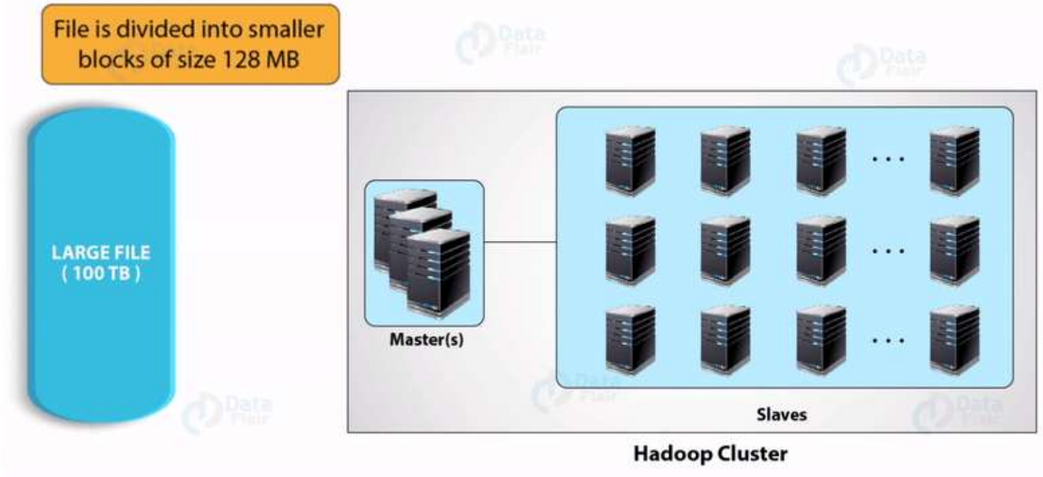


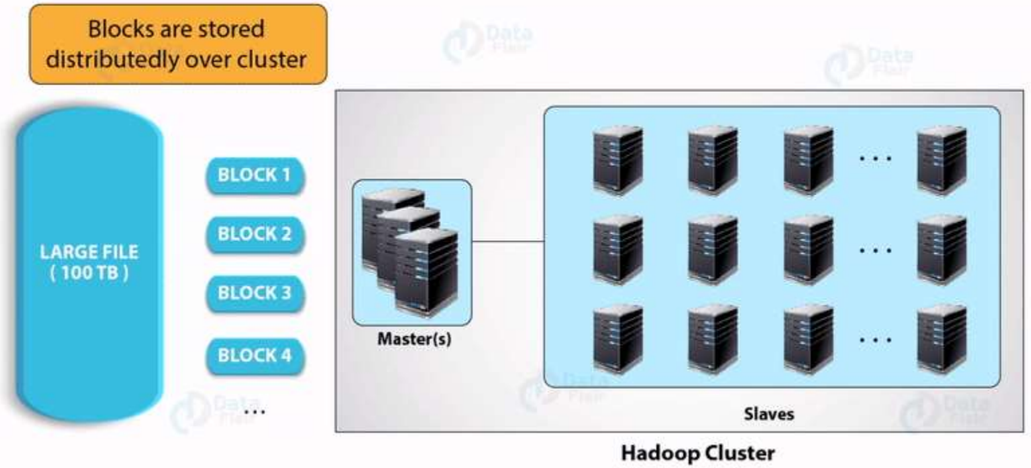



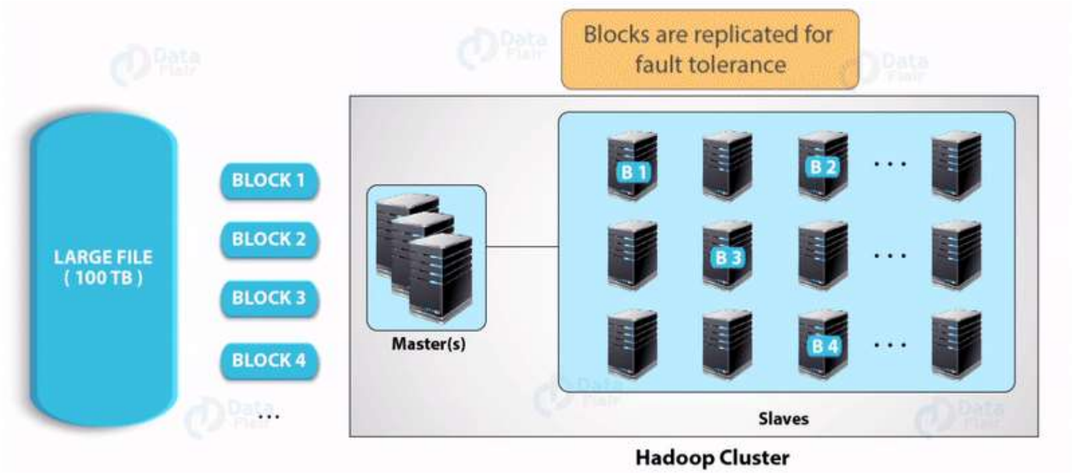



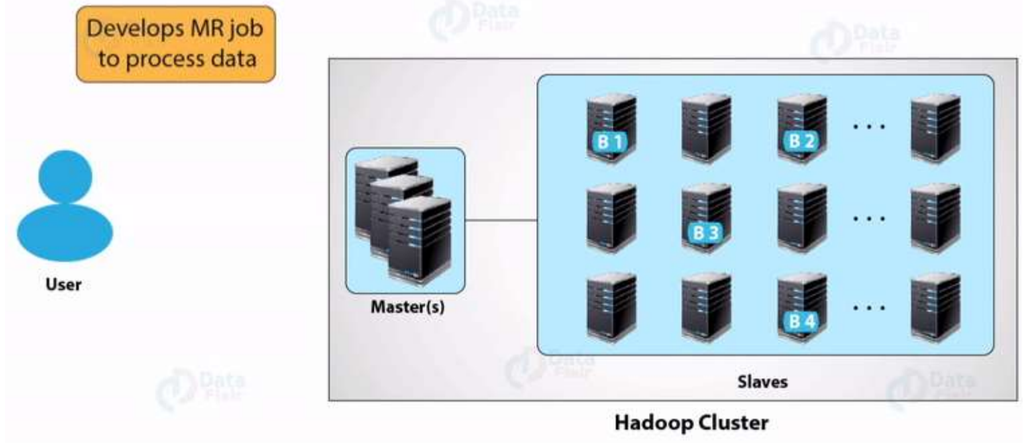
# Data Storage in HDFS

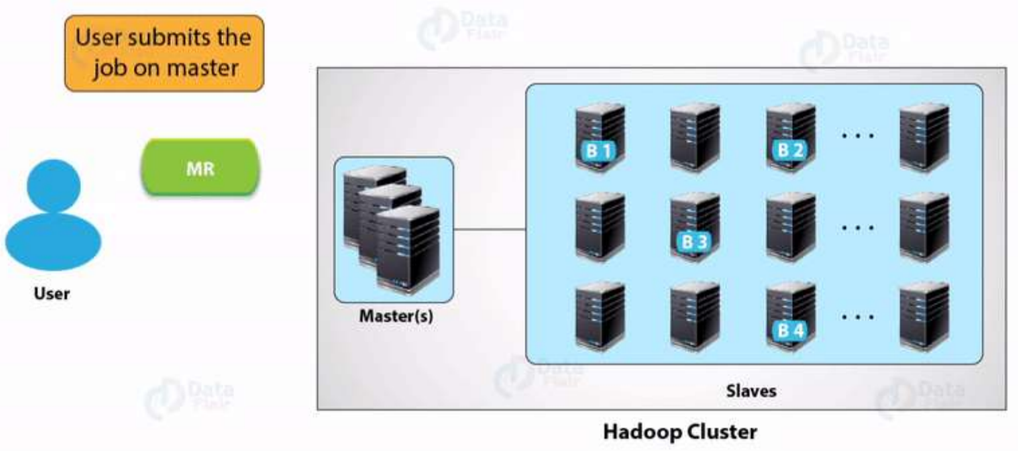


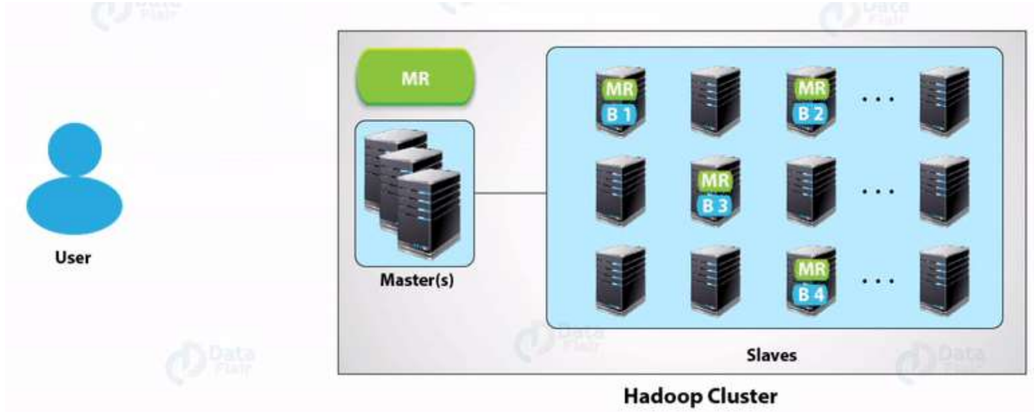



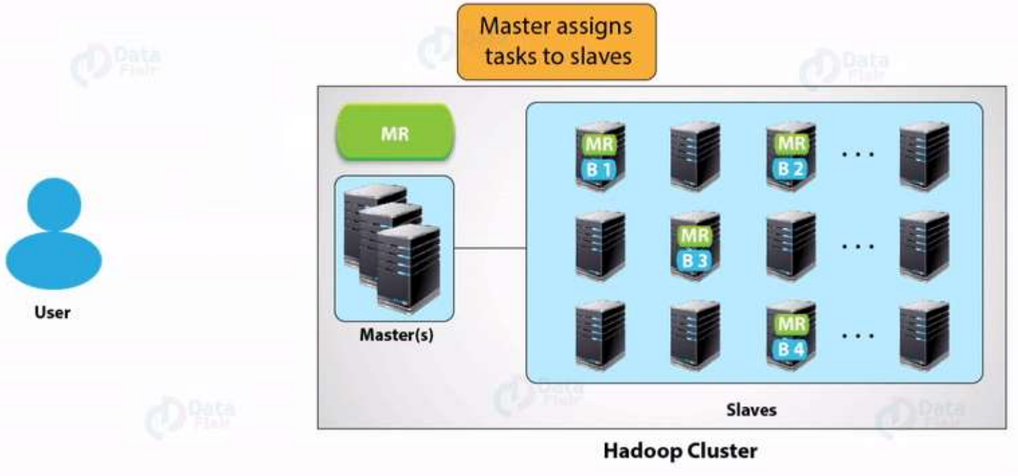



# How Map Reduce Works



# Why Hadoop?

Hence enhancing performance dramatically.

**c. Storing the variety of data –** HDFS solved this problem. HDFS can store all kind of data (structured, semi-structured or unstructured). It also follows *write once and read many models*.

Due to this, you can write any kind of data once and you can read it multiple times for finding insights.

**d. Data Processing Speed –** This is the major problem of big data. In order to solve this problem, move computation to data instead of data to computation. This principle is **Data locality**.

# Hadoop Core Components

Now we will learn the Apache Hadoop core component in detail. It has 3 core components-

HDFS

MapReduce

YARN(Yet Another Resource Negotiator)

Let’s discuss these core components one by one.

# Hadoop Core Components

a. HDFS

**Hadoop distributed file system (HDFS)** is the primary storage system of Hadoop. HDFS store very large files running on a cluster of commodity hardware. It follows the principle of storing less number of large files rather than the huge number of small files.
Stores data reliably even in the case of hardware failure. It provides high-throughput access to the application by accessing in parallel.

**Components of HDFS:**

**NameNode –**It works as Master in the cluster. Namenode stores **meta-data**. A number of blocks, replicas and other details. Meta-data is present in memory in the master. NameNode maintains and also manages the slave nodes, and assigns tasks to them. It should deploy on reliable hardware as it is the centerpiece of HDFS.

**DataNode –** It works as Slave in the cluster. In HDFS, DataNode is responsible for storing actual data in HDFS. DataNode performs read and write operation as per request for the clients. DataNodes can also deploy on commodity hardware.

# Hadoop Core Components

b. MapReduce

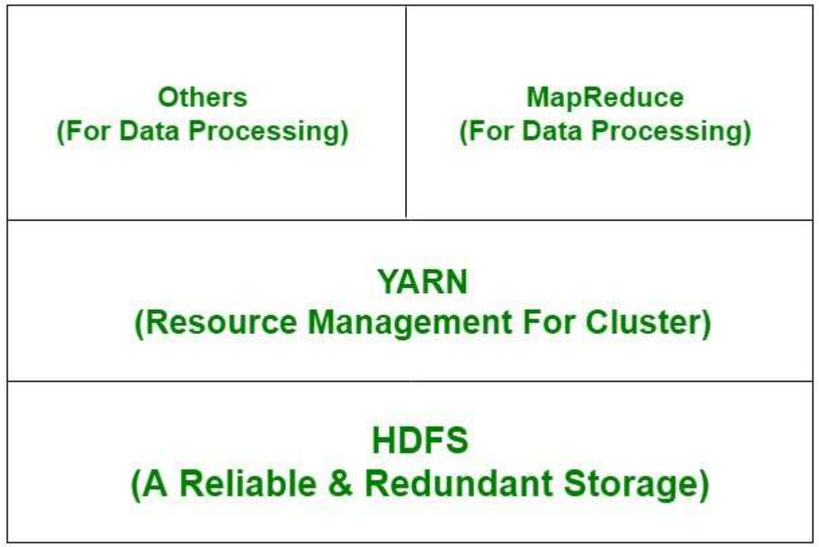
MapReduce is the data processing layer of Hadoop. It processes large structured and unstructured data stored in HDFS. MapReduce also processes a huge amount of data in parallel.

It does this by dividing the job (submitted job) into a set of independent tasks (sub-job).

MapReduce works by breaking the processing into phases: Map and Reduce.

**Map –** It is the first phase of processing, where we specify all the complex logic code.

**Reduce –**It is the second phase of processing. Here we specify light-weight processing like aggregation/summation.



# Hadoop Core Components

c. YARN

YARN provides the resource management. It is the operating system of Hadoop. It responsible for managing and monitoring workloads, also implementing security controls. Apache YARN is also a central platform to deliver data governance tools across the clusters.

YARN allows multiple data processing engines such as real-time streaming, batch processing etc.

**Components of YARN:**

**Resource Manager –** It is a cluster level component and runs on the Master machine. It manages resources and schedule applications running on the top of YARN. It has two components: Scheduler & Application Manager.

**Node Manager –**It is a node level component. It runs on each slave machine. It continuously communicate with Resource Manager to remain up-to-date

# Advantages of Hadoop

Let’s now discuss various Hadoop advantages to solve the big data problems.

**Scalability –**By adding nodes we can easily grow our system to handle more data.

**Flexibility –** In this framework, you don’t have to preprocess data before storing it. You can store as much data as you want and decide how to use later.

**Low-cost –** Open source framework is free and runs on low-cost commodity hardware.

**Fault tolerance –** If nodes go down, then jobs are automatically redirected to other nodes.

**Computing power –** It’s distributed computing model processes big data fast. The more computing nodes you use more processing power you have.

**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.

# Disadvantages of Hadoop

Some Disadvantage of Apache Hadoop Framework is given below-

**Security concerns –** It can be challenging in managing the complex application. If the user doesn't know how to enable platform who is managing the platform, then your data could be a huge risk. Since, storage and network levels Hadoop are missing encryption, which is a major point of concern.

**Vulnerable by nature –** The framework is written almost in java, most widely used language. Java is heavily exploited by cybercriminals. As a result, implicated in numerous security breaches.

**Not fit for small data –**Since, it is not suited for small data. Hence, it lacks the ability to efficiently support the random reading of small files.

**Potential stability issues –** As it is an open source framework. This means that it is created by many developers who continue to work on the project. While constantly improvements are made, It has stability issues. To avoid these issues organizations should run on the latest stable version.

# 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.

# Disadvantages of Hadoop

- 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.

# Reference


1. https://www.javatpoint.com/what-is-hadoop
2. https://techvidvan.com/tutorials/apache-hadoop-tutorials/
3. https://www.geeksforgeeks.org/hadoop-an-introduction/

# THANK YOU