

# Big Data Hadoop and Spark Developer



## HDFS: The Storage Layer



# Learning Objectives

By the end of this lesson, you will be able to:

- 👁 Summarize how the Hadoop Distributed File System (HDFS) stores data across a cluster
- 👁 Illustrate the HDFS architecture and its components
- 👁 Demonstrate the use of the HDFS Command Line Interface (CLI)

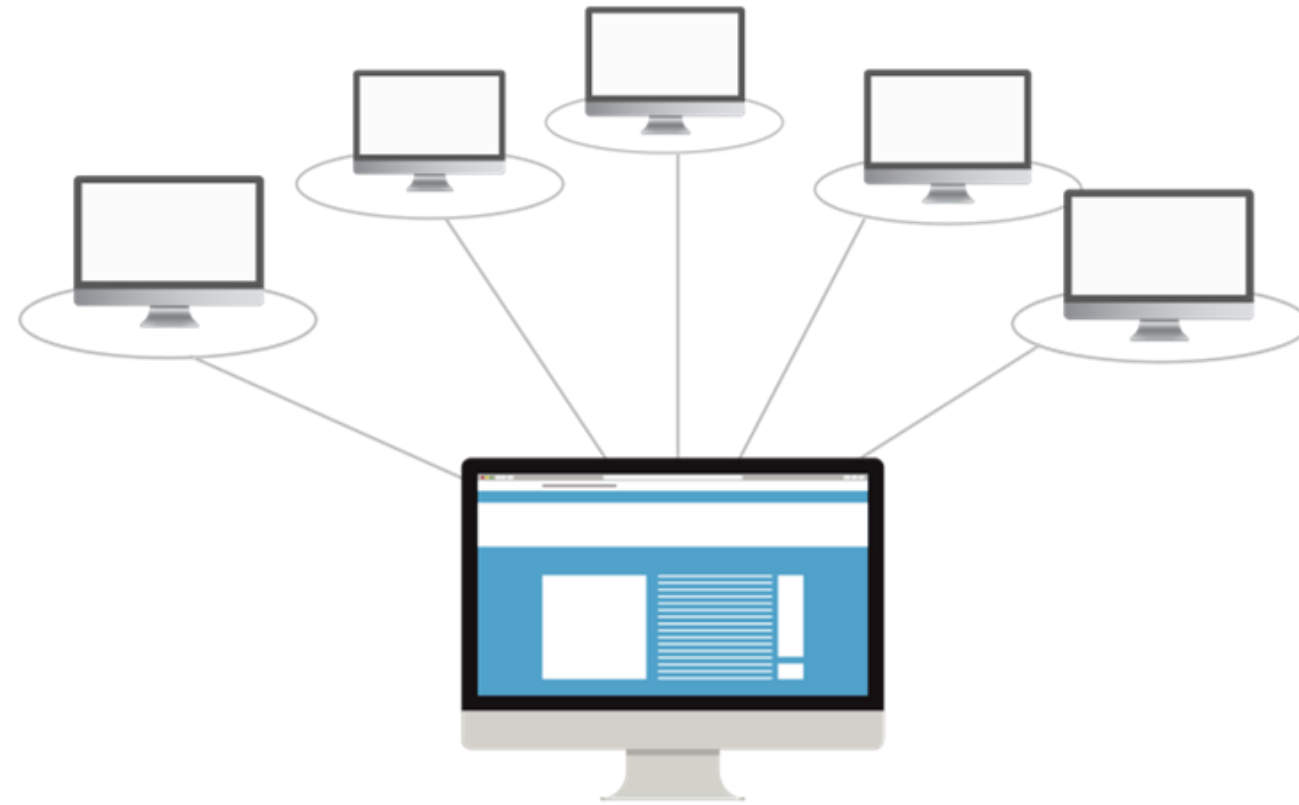




# **Hadoop Distributed File System (HDFS)**

# What Is HDFS?

HDFS is a distributed file system that provides access to data across Hadoop clusters.



It manages and supports the analyses of large volumes of big data.

# Challenges of Traditional Systems

In the traditional system, storing and retrieving volumes of data had three major issues:

## Cost



\$10,000 to \$14,000  
per terabyte

## Speed



Search and analysis  
is time-consuming

## Reliability



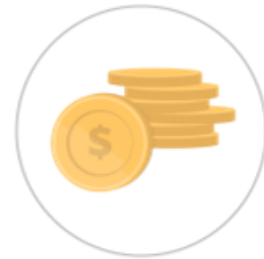
Fetching data is difficult

# Need for HDFS

HDFS resolves all the three major issues of the traditional file system.

## Cost

Has zero licensing and support costs



## Reliability

Copies data multiple times

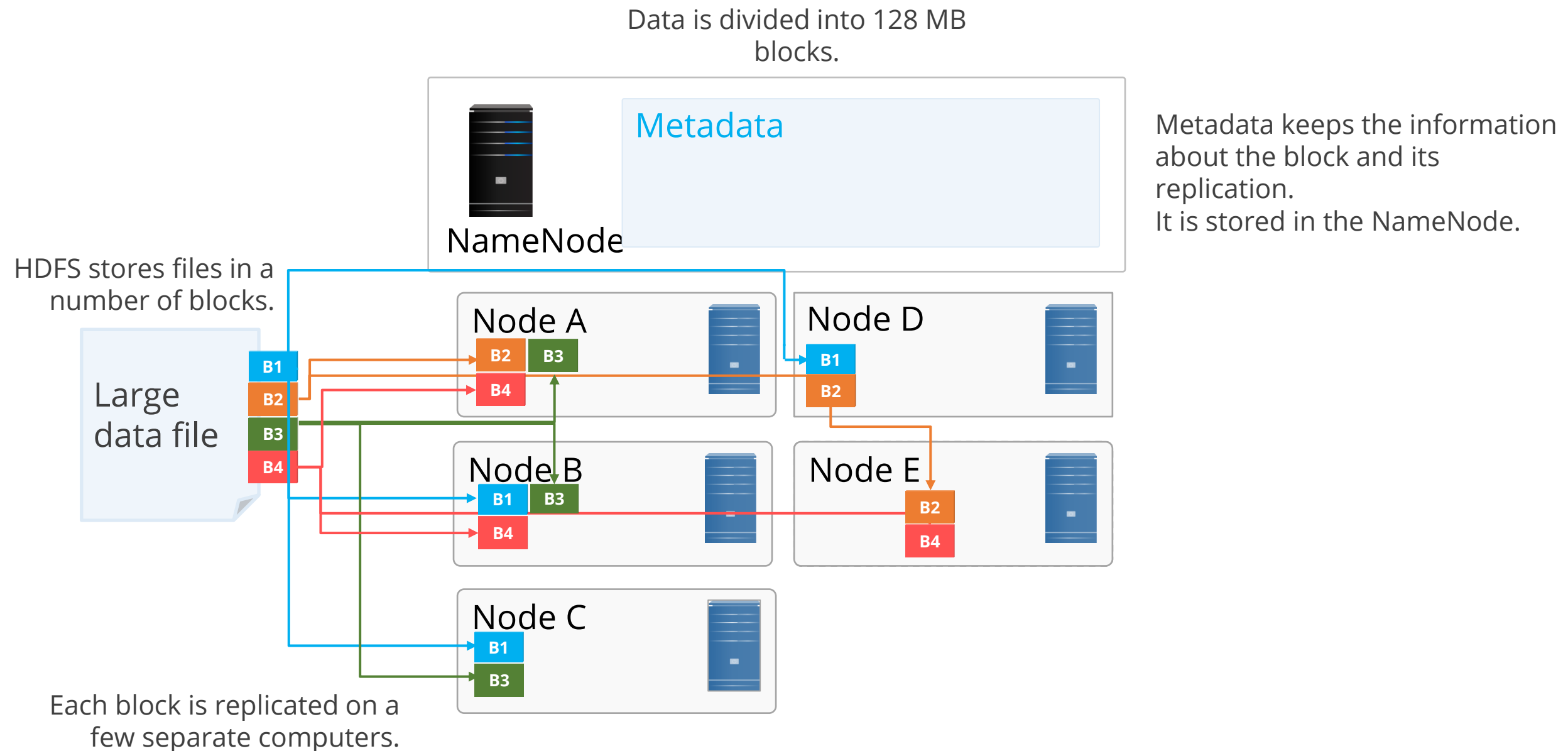


## Speed

Reads or writes more than one terabyte of data per second

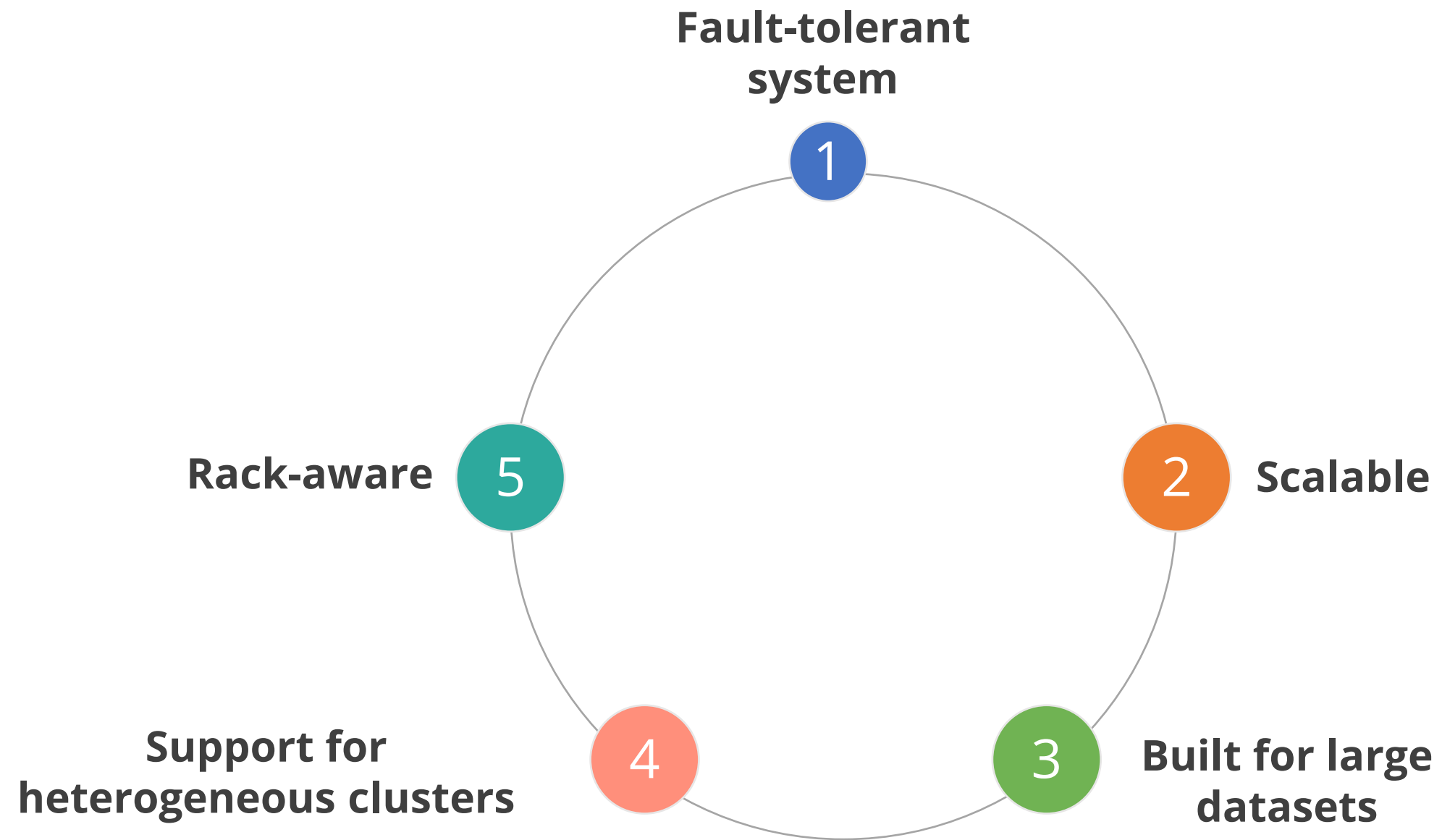
# HDFS Storage

HDFS is a distributed file system that uses NameNode and DataNode architectures to allow high-performance data access across highly scalable Hadoop clusters.





# Characteristics of HDFS

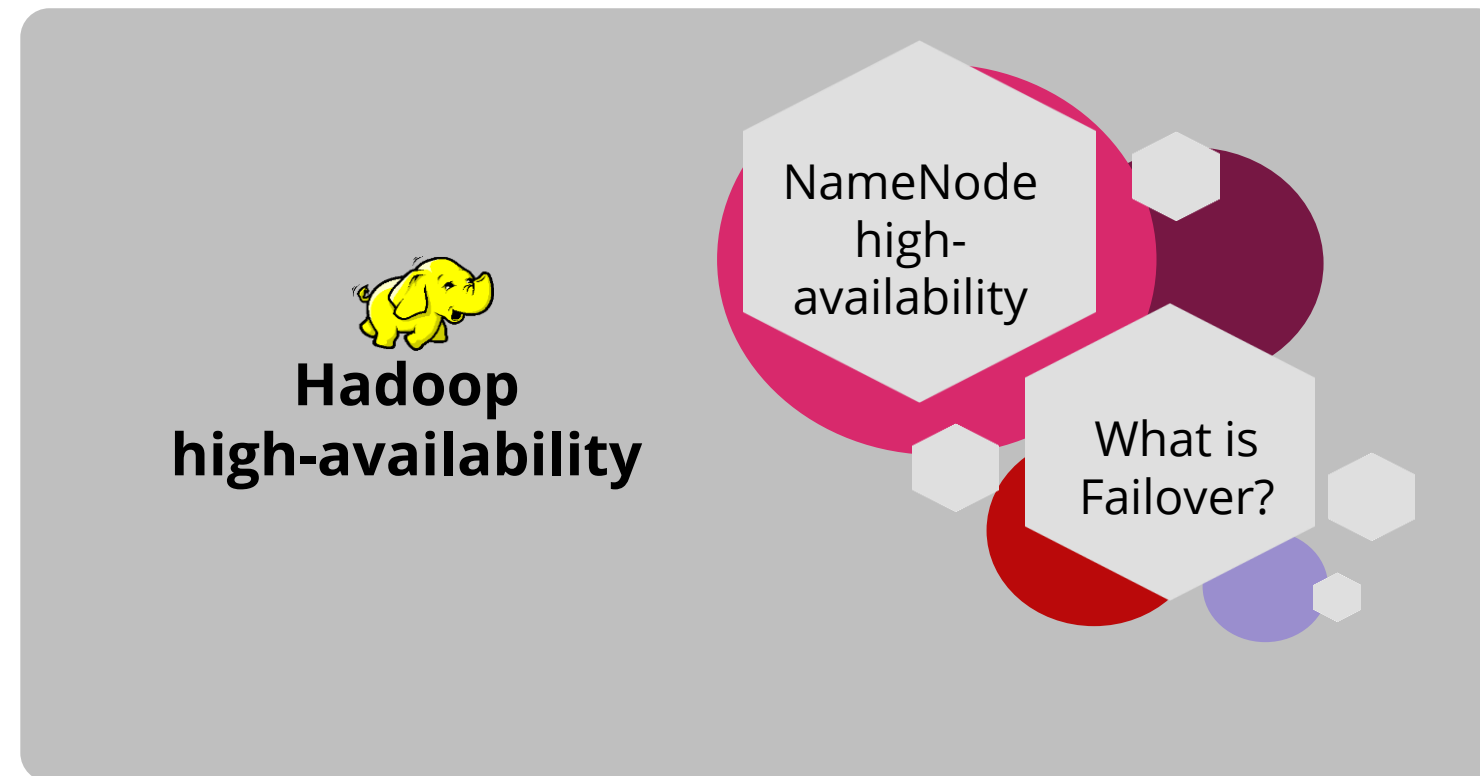




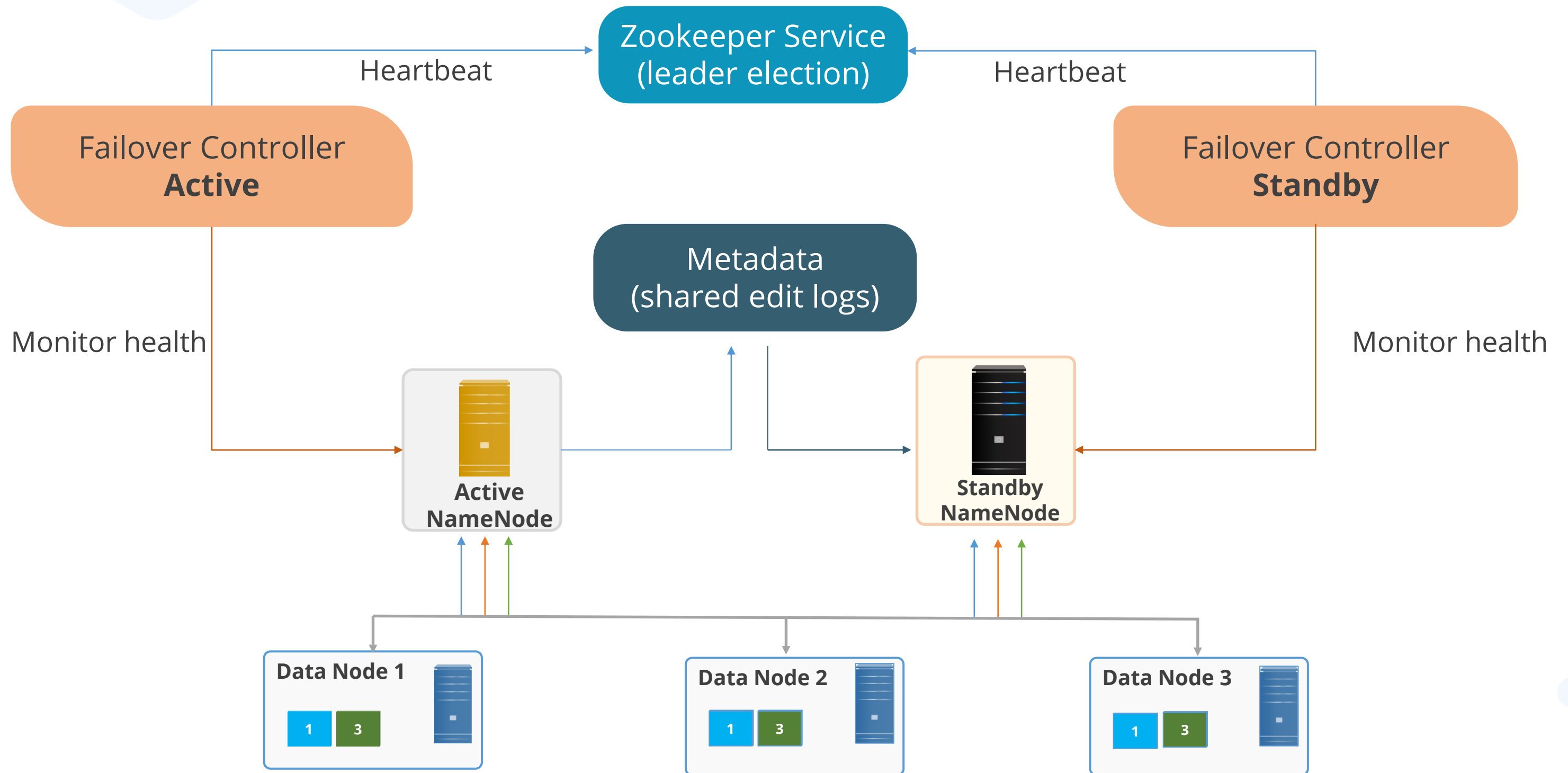
# **HDFS Architecture and Components**

# HDFS High-Availability Architecture

Hadoop's high-availability functionality ensures that the Hadoop cluster remains operational despite unfavorable conditions, such as NameNode failure, DataNode failure, or computer crashes. It indicates that the data will be available through a standby NameNode when the NameNode fails.



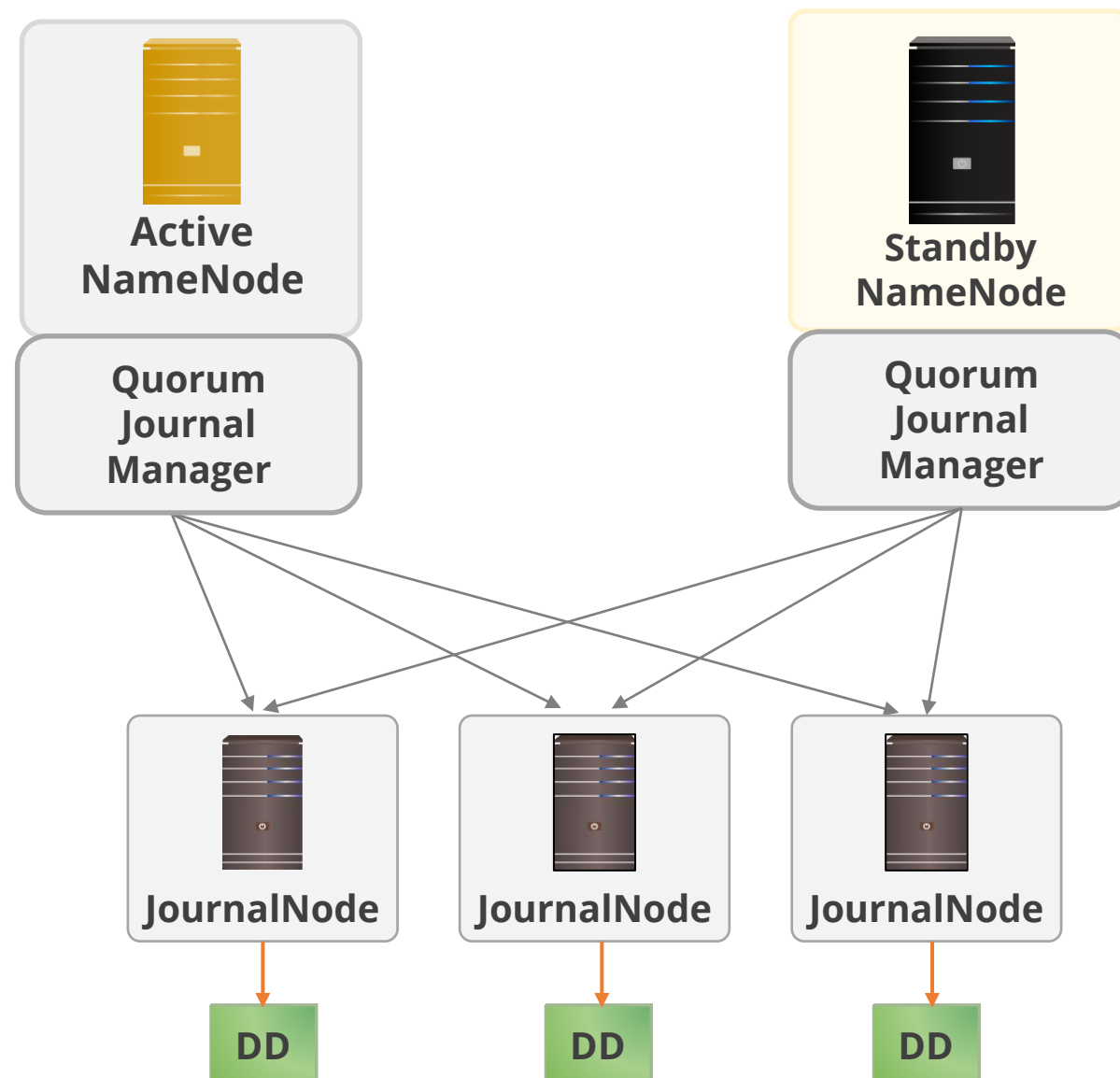
# HDFS High-Availability Architecture



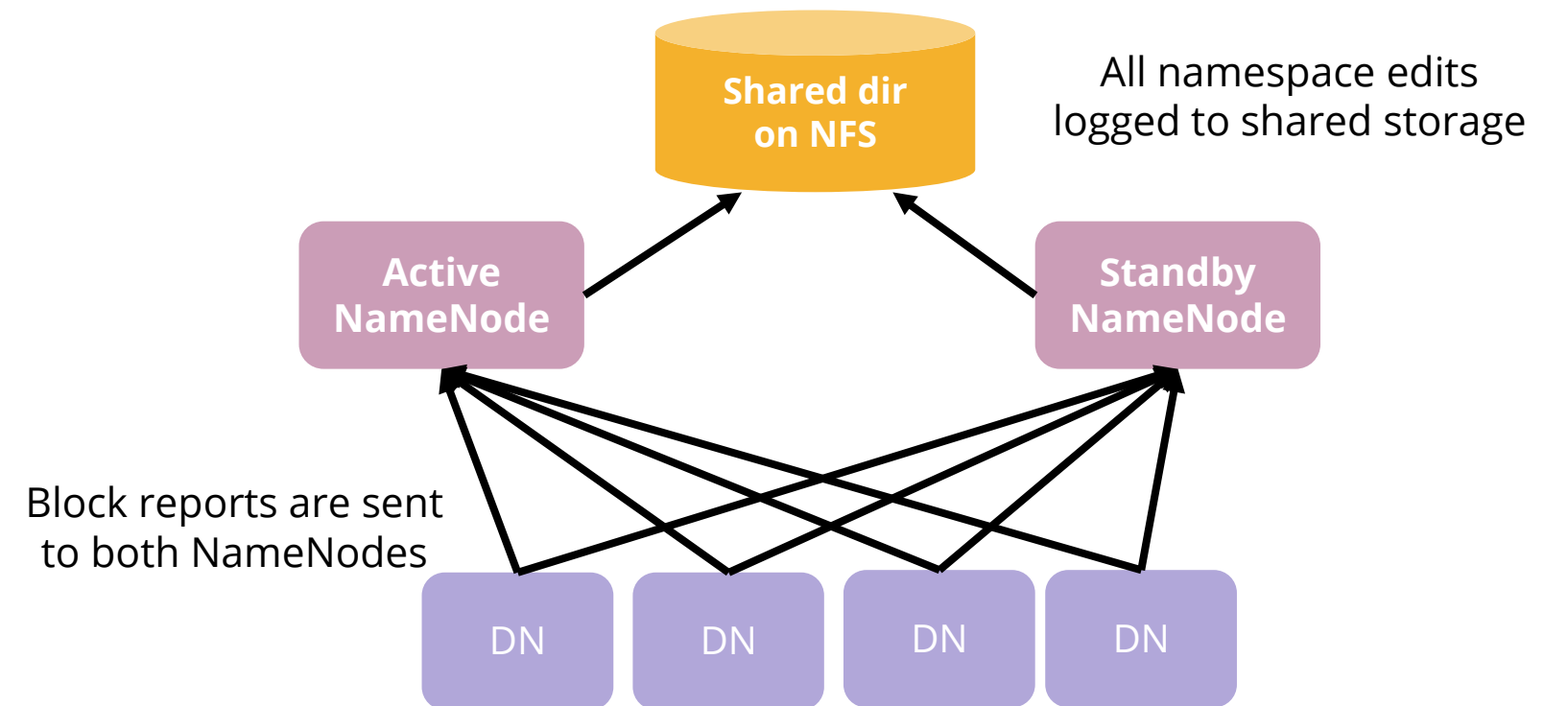
# Types of HDFS Architectures

There are two types of HDFS high-availability architectures:

## Quorum-based storage

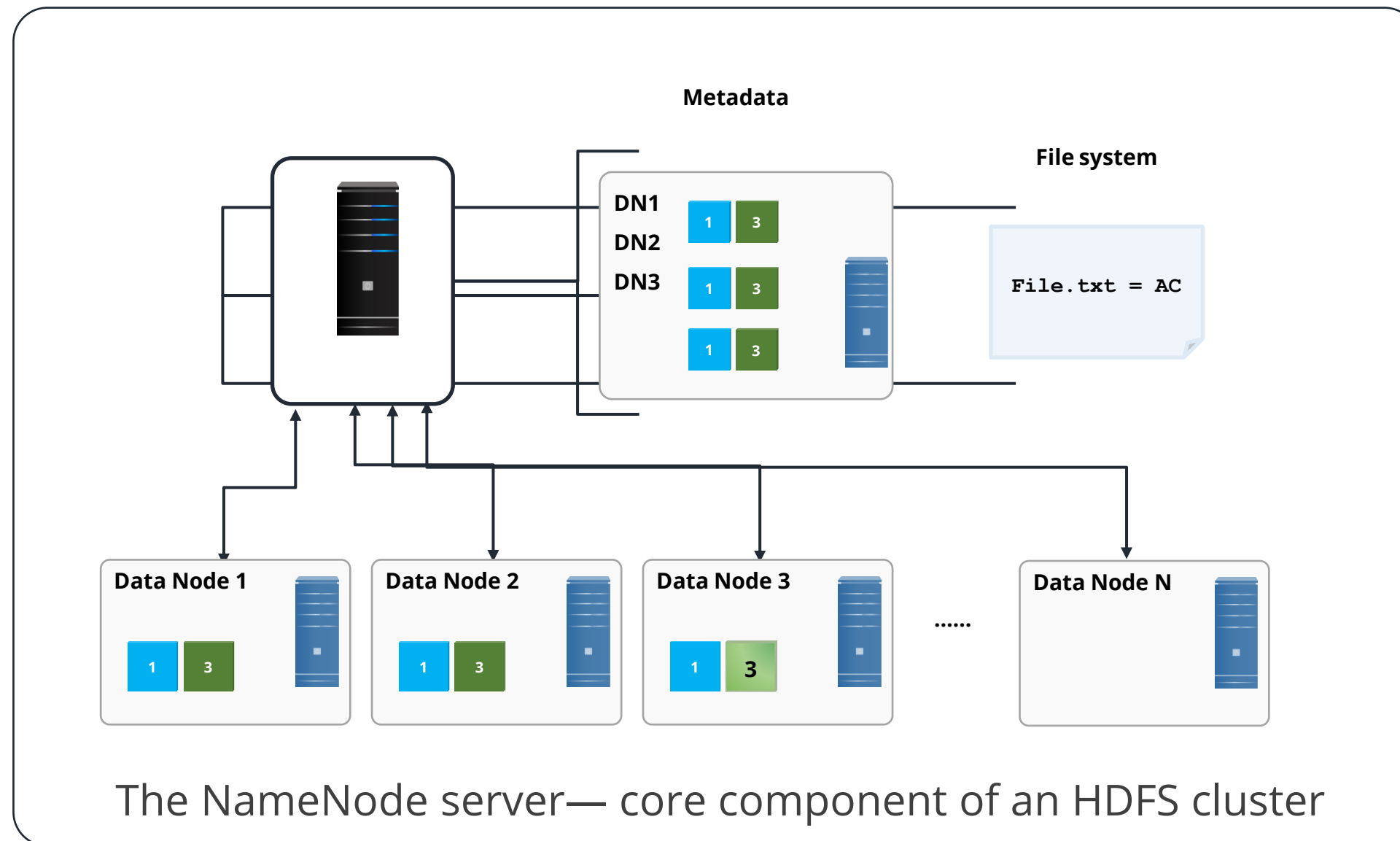


## Shared storage using NFS



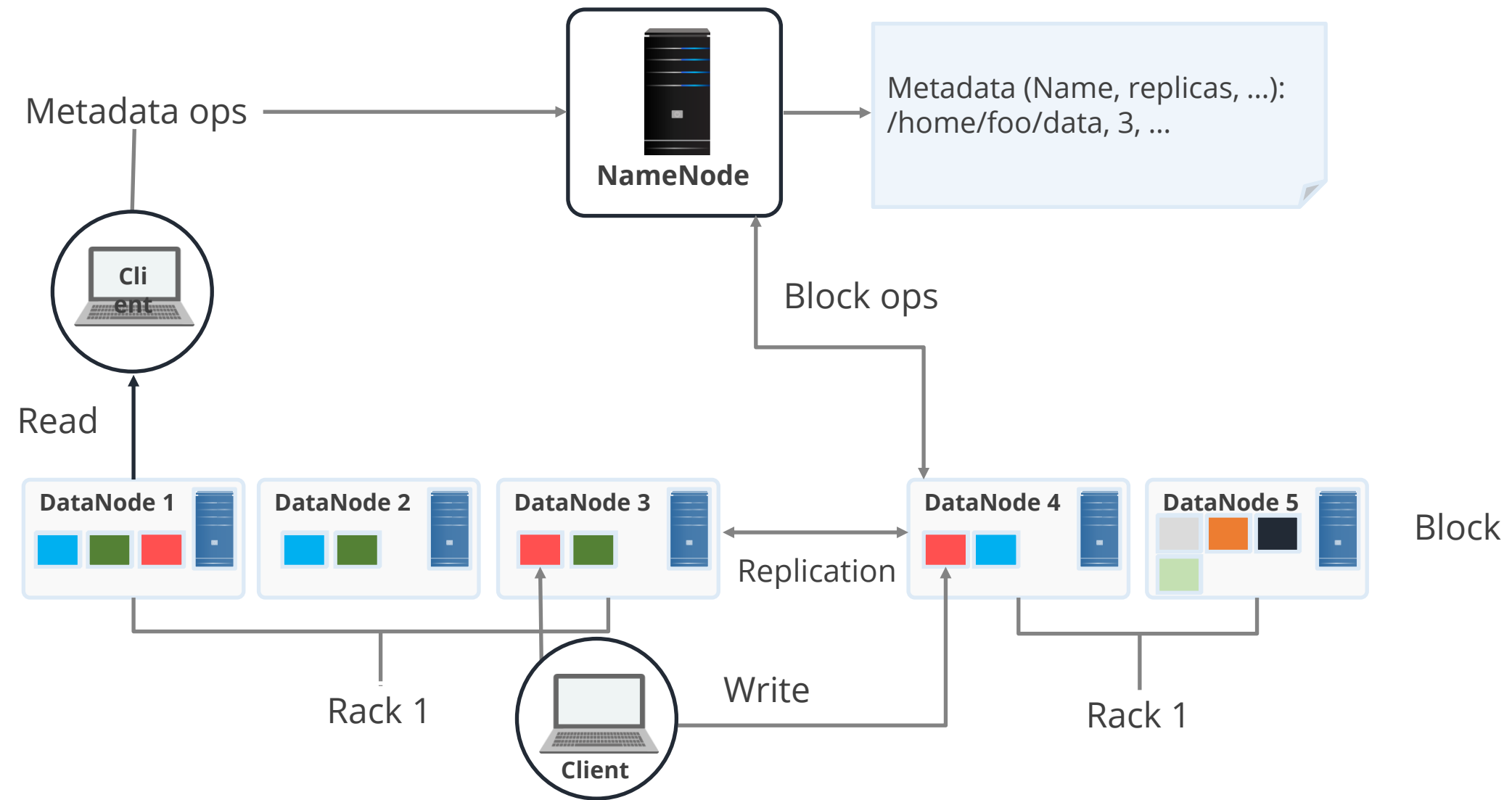
# HDFS Component: NameNode

The HDFS components comprise different servers like NameNode, DataNode, and Secondary NameNode.



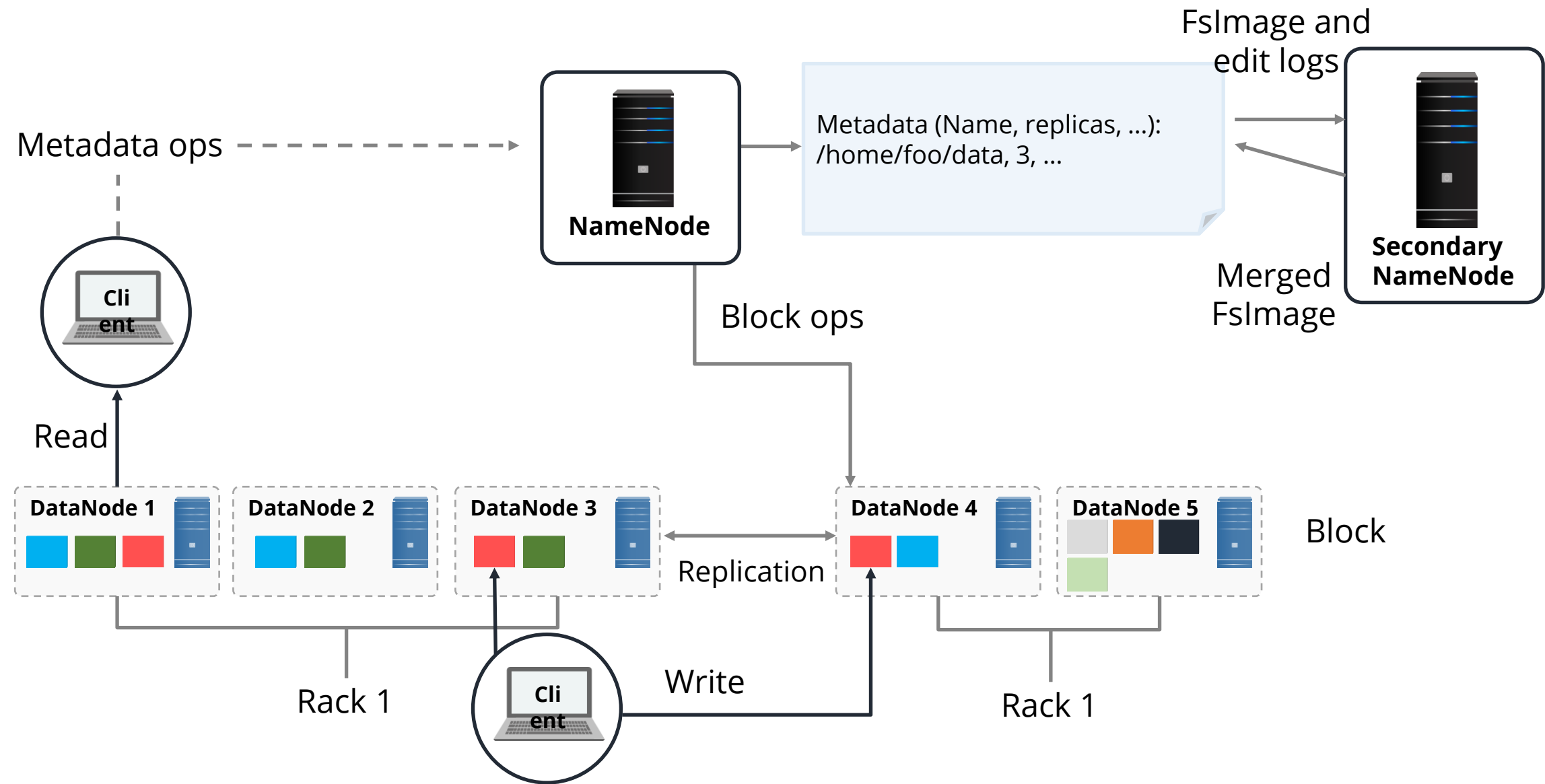
# HDFS Component: DataNode

The DataNode is a multiple instance server. DataNode servers are responsible for storing and maintaining the data blocks.



# HDFS Component: SNN

In the event of data loss or any disaster, NameNode acts as a backup node.

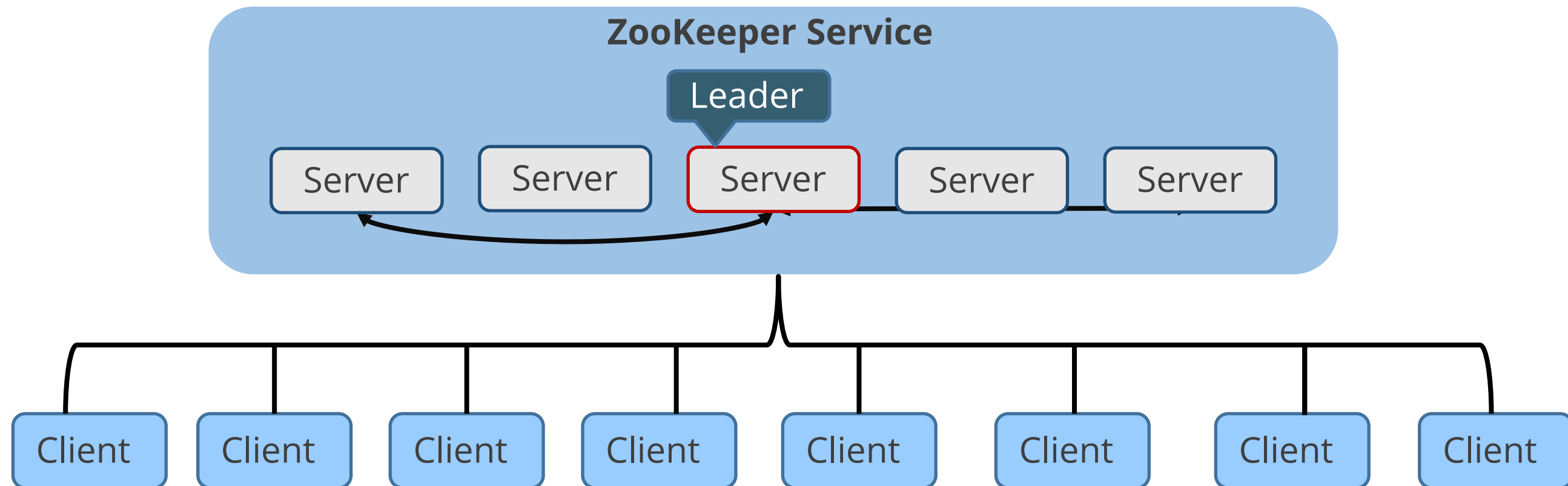


It takes hourly snapshots of data stored by an active NameNode and stores it in FsImage.



# HDFS Component: ZooKeeper

ZooKeeper allows distributed processes to coordinate with each other through a shared hierarchical namespace.



The ZooKeeper implementation is simple and replicated, which puts a premium on high performance, high availability, and a strictly ordered access.

# HDFS Component: ZooKeeper

Automatic HDFS Failover relies on ZooKeeper for the following:

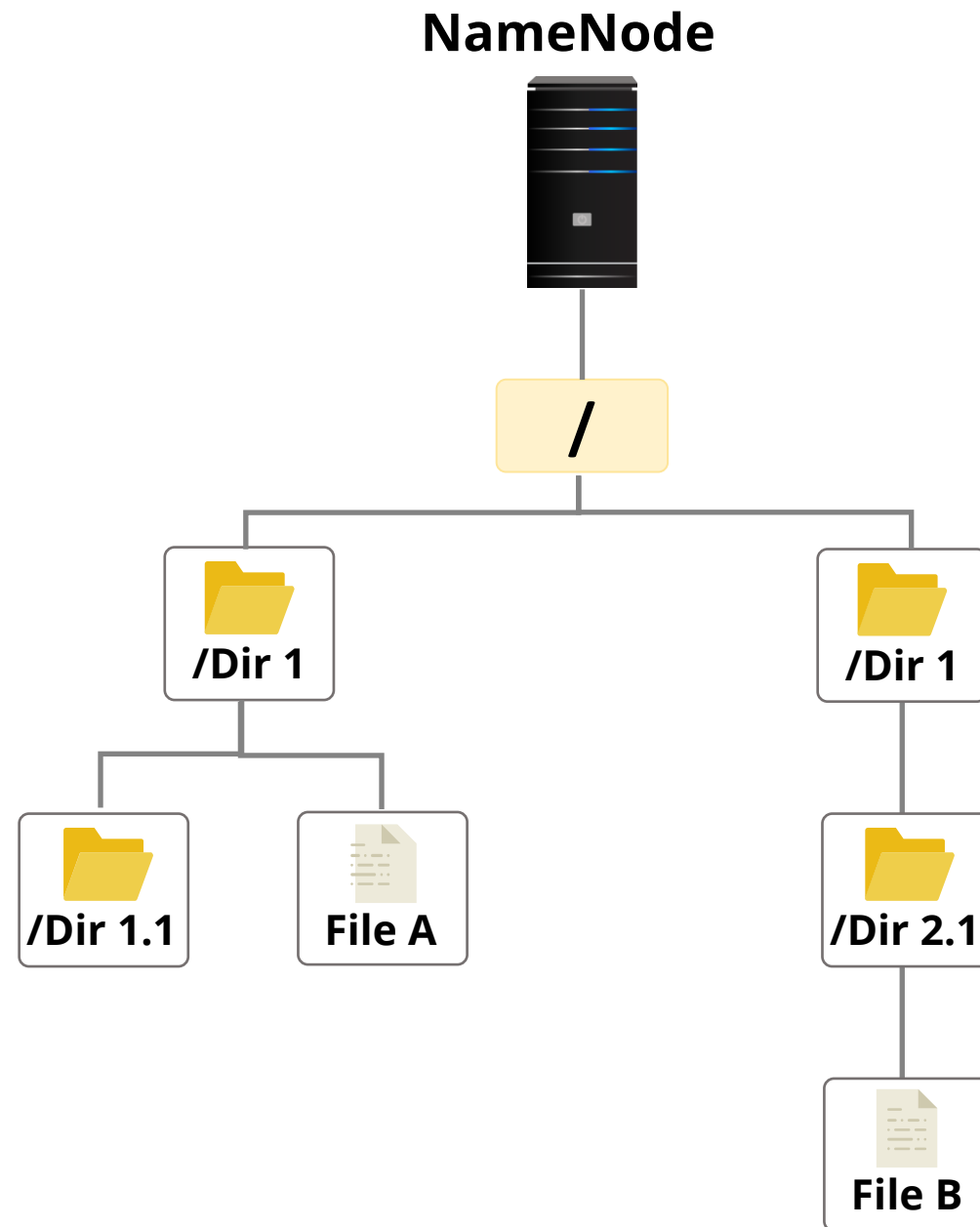


Failure detection



Active NameNode election

# HDFS Component: File System Namespace

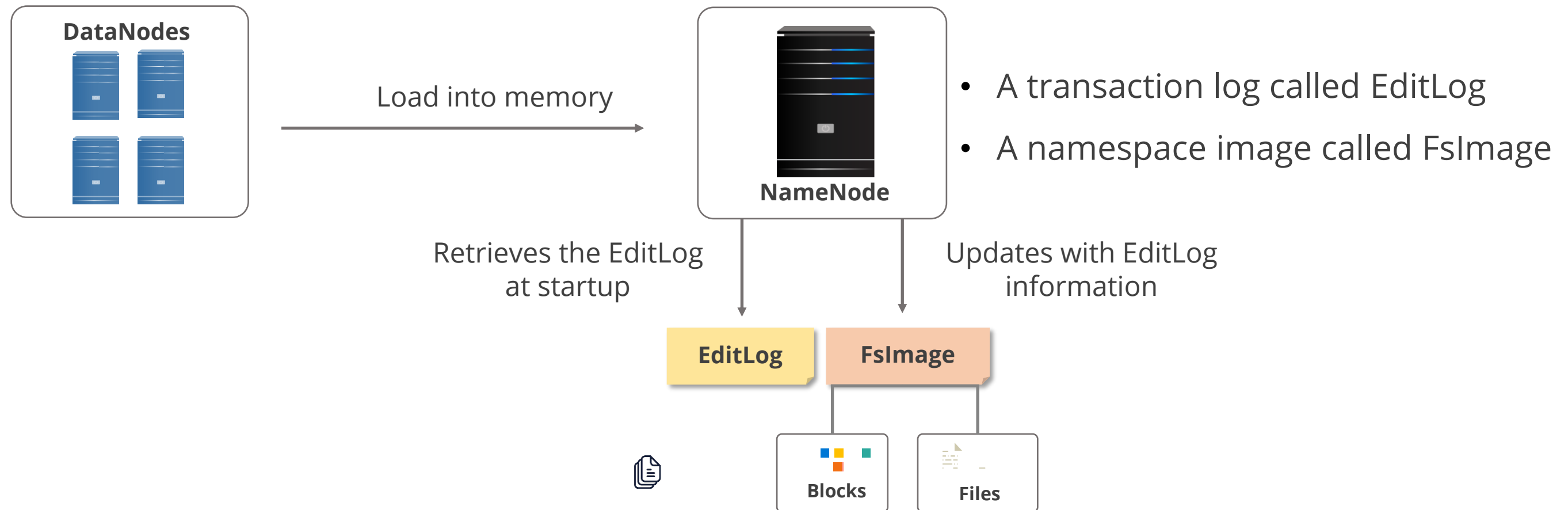


The HDFS file system:

- Allows user data to be stored in files
- Follows a hierarchical file system with directories and files
- Supports operations like create, remove, move, and rename

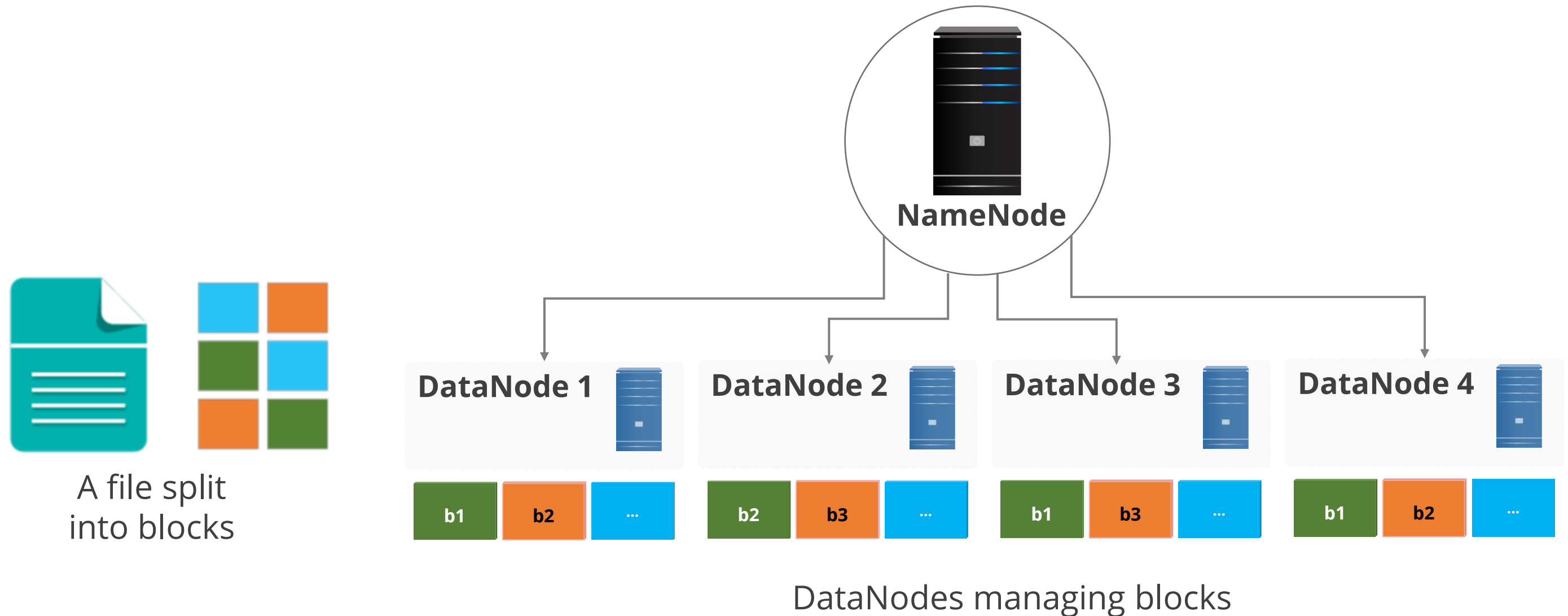
# NameNode Operation

The NameNode maintains two persistent files:



# Data Block Split

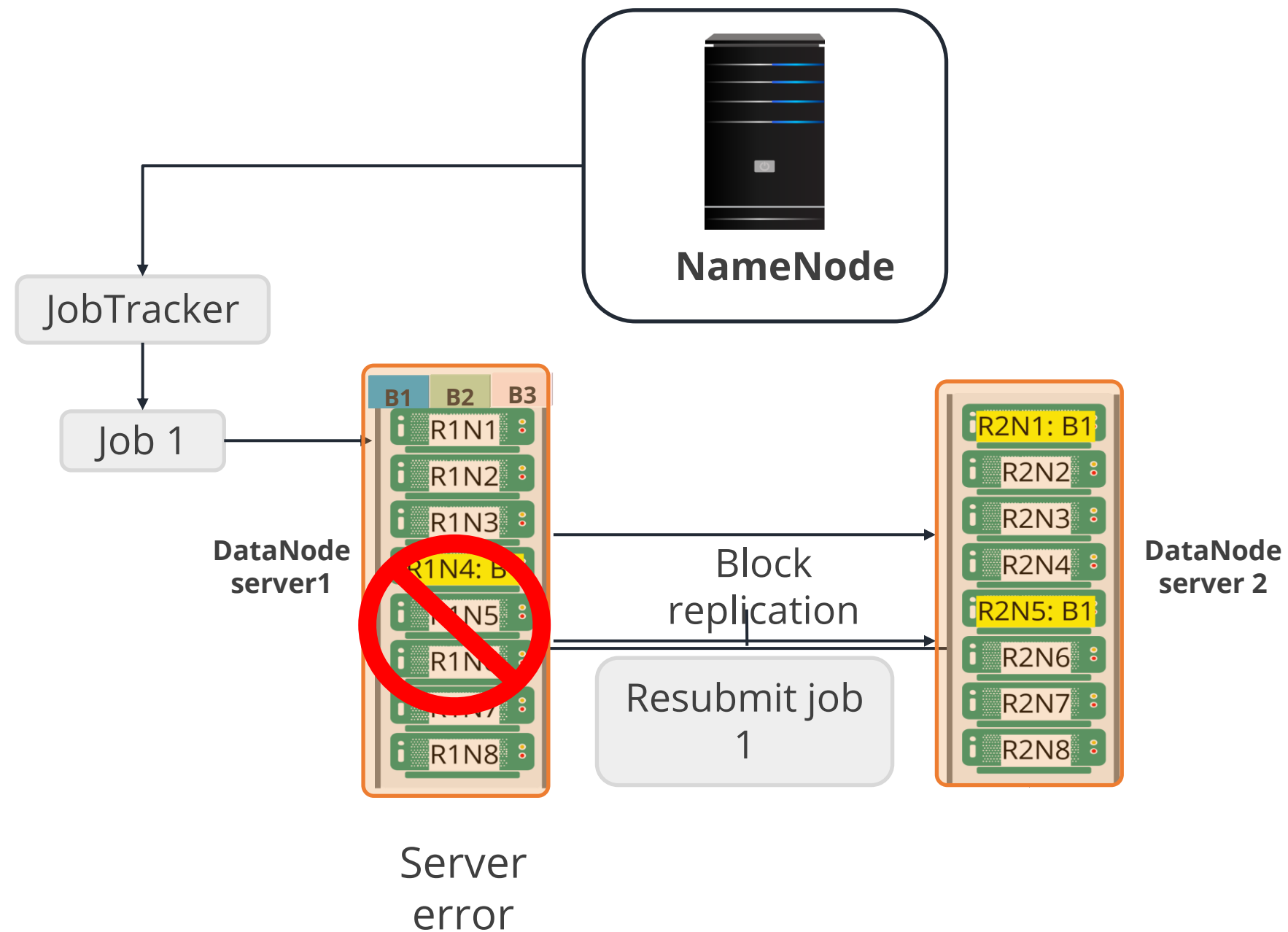
Each file is split into one or more blocks that are stored and replicated in DataNodes.



The data block approach provides simplified replication, fault tolerance, and reliability.

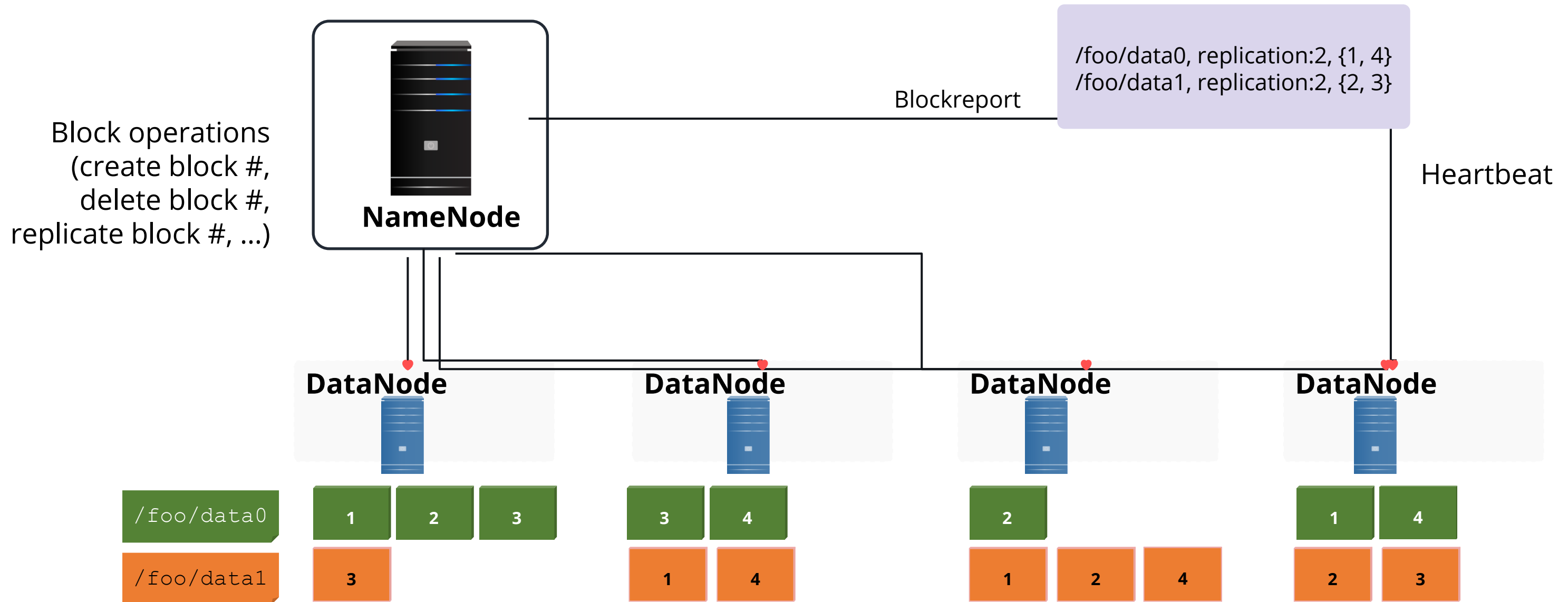
# Block Replication Architecture

HDFS represents the unstructured data in the form of data blocks. It performs block replication on multiple DataNodes.



# Replication Method

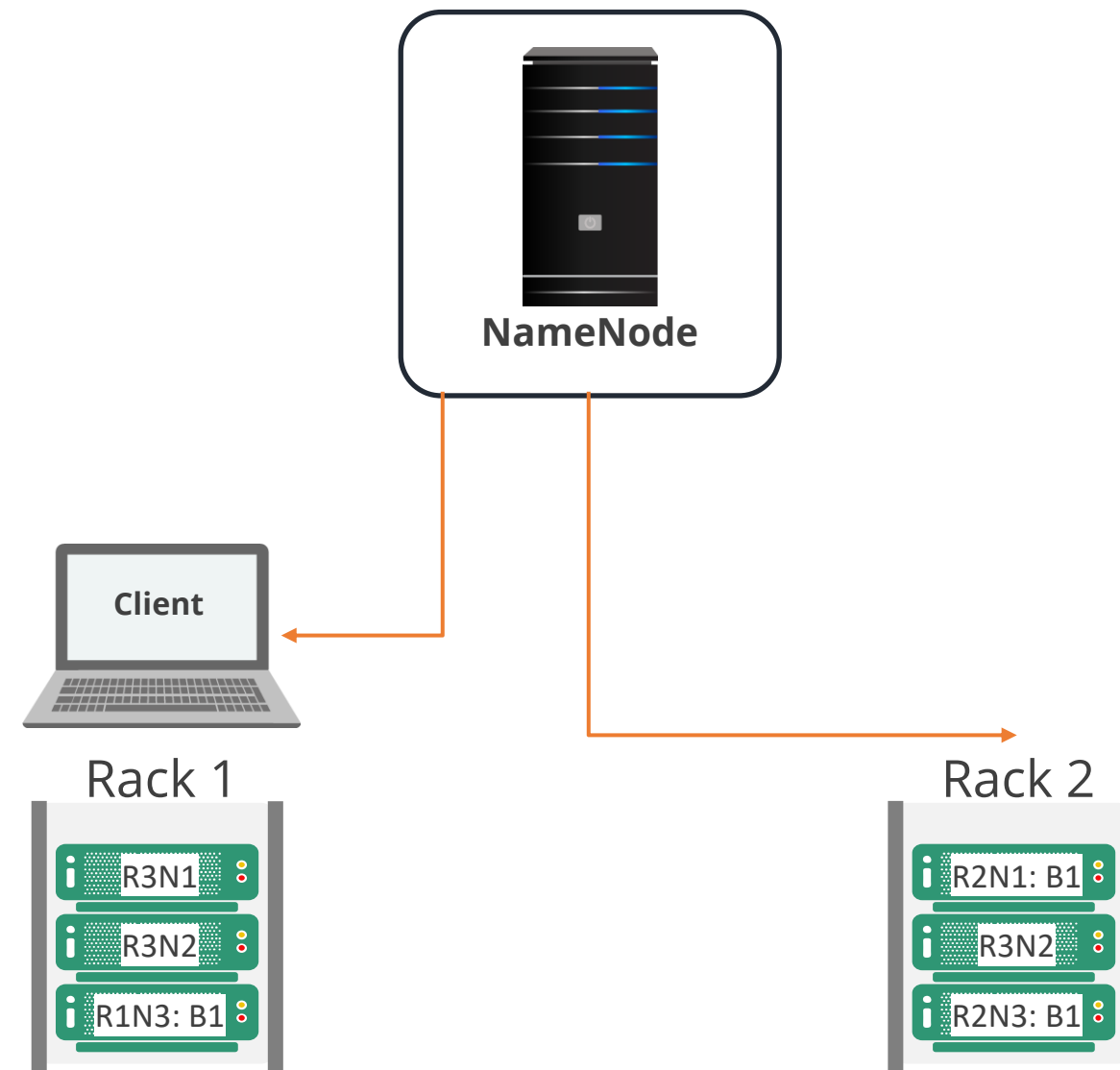
Each file is split into a sequence of blocks. Except for the last one, all blocks in the file are of the same size.



# Data Replication Topologies

One of the suggested replication topologies is as follows:

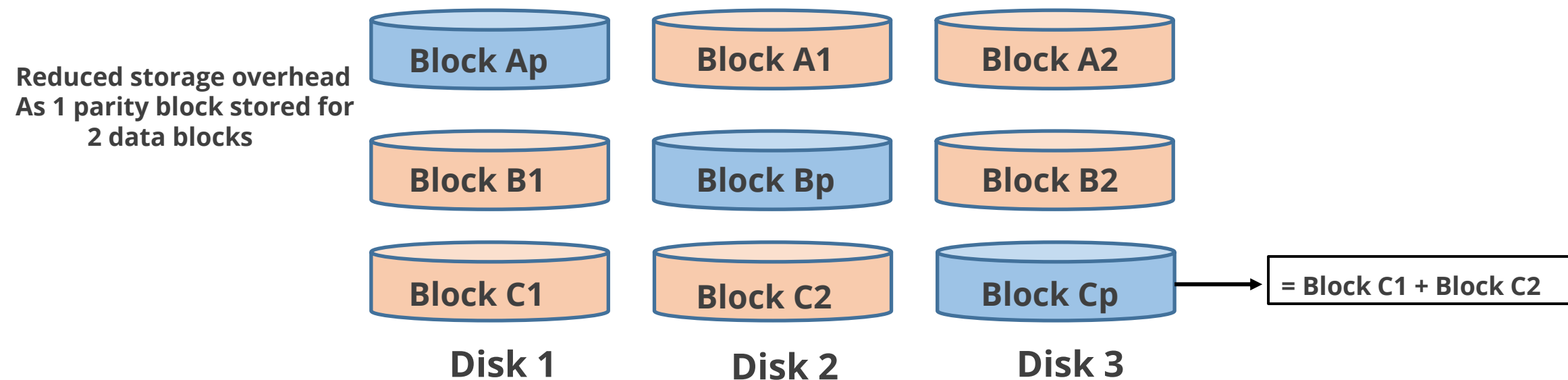
Click to add text





# HDFS: Erasure Coding

It is a data protection technique where data is broken into fragments, expanded, encoded with redundant information, and stored in different locations or storage media.

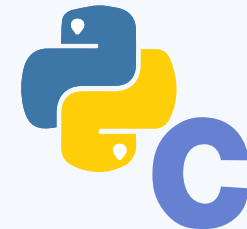


If a storage medium fails or data is corrupted, the data can be reconstructed from parts stored on other storage media.

# HDFS Access

HDFS provides the following access mechanisms:

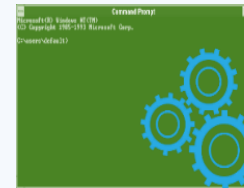
Java API for applications



Python access and C wrapper for non-Java applications



Web GUI utilized through an HTTP browser



FS shell for executing commands on HDFS

# HDFS Command Line

Copies *simplilearn.txt* from the local disk to the user's directory in HDFS

`$hdfs dfs -put simplilearn.txt  
simplilearn.txt`

Displays a list of the contents of the directory path provided by the user, and shows the names, permissions, owner, size, and modification date for each entry

`$hdfs dfs -ls /user/simpl/test`

Creates a directory called *test* under the user's home directory

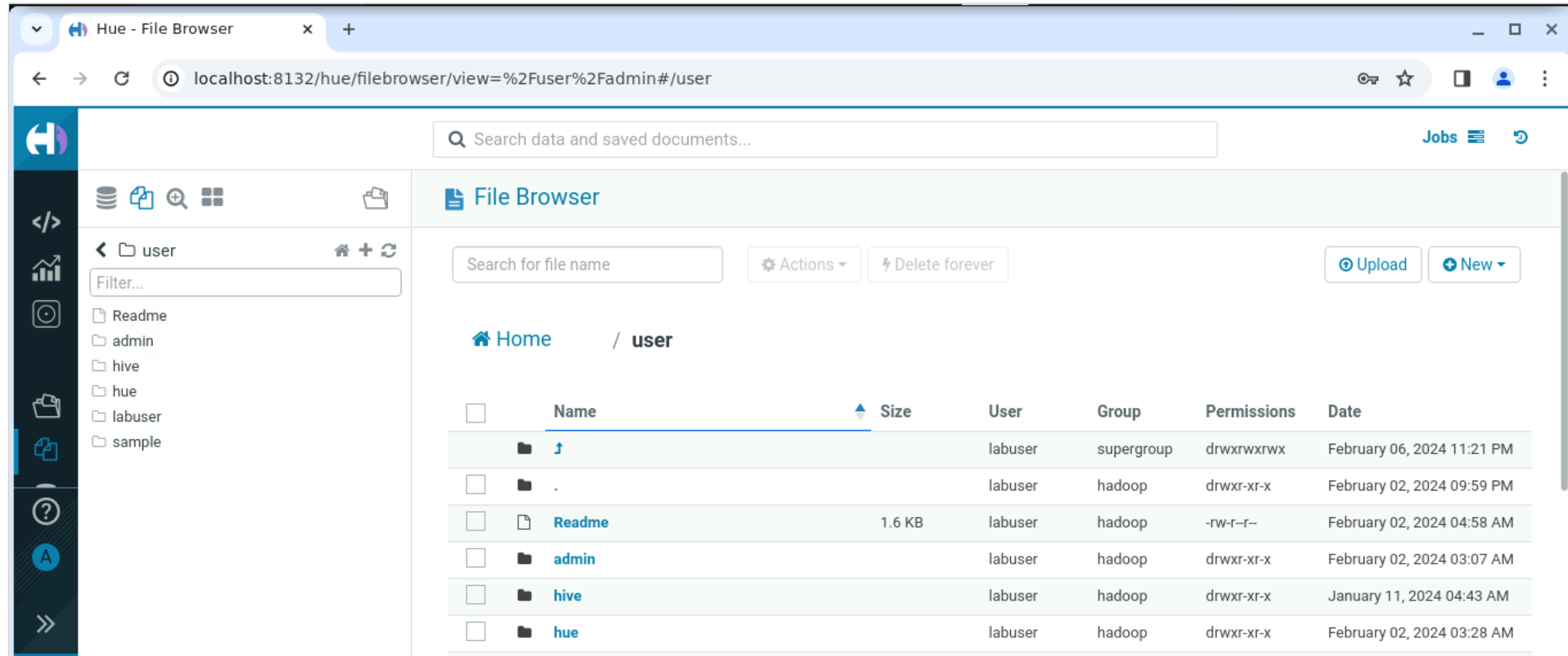
`$hdfs dfs -mkdir /user/simpl/test`

Delete the directory *test* and all content within

`$hdfs dfs -rm -f test`

# Hue File Browser

The file browser in Hue lets you view and manage your HDFS directories and files.



Additionally, you can create, move, rename, modify, upload, download, and delete directories and files.

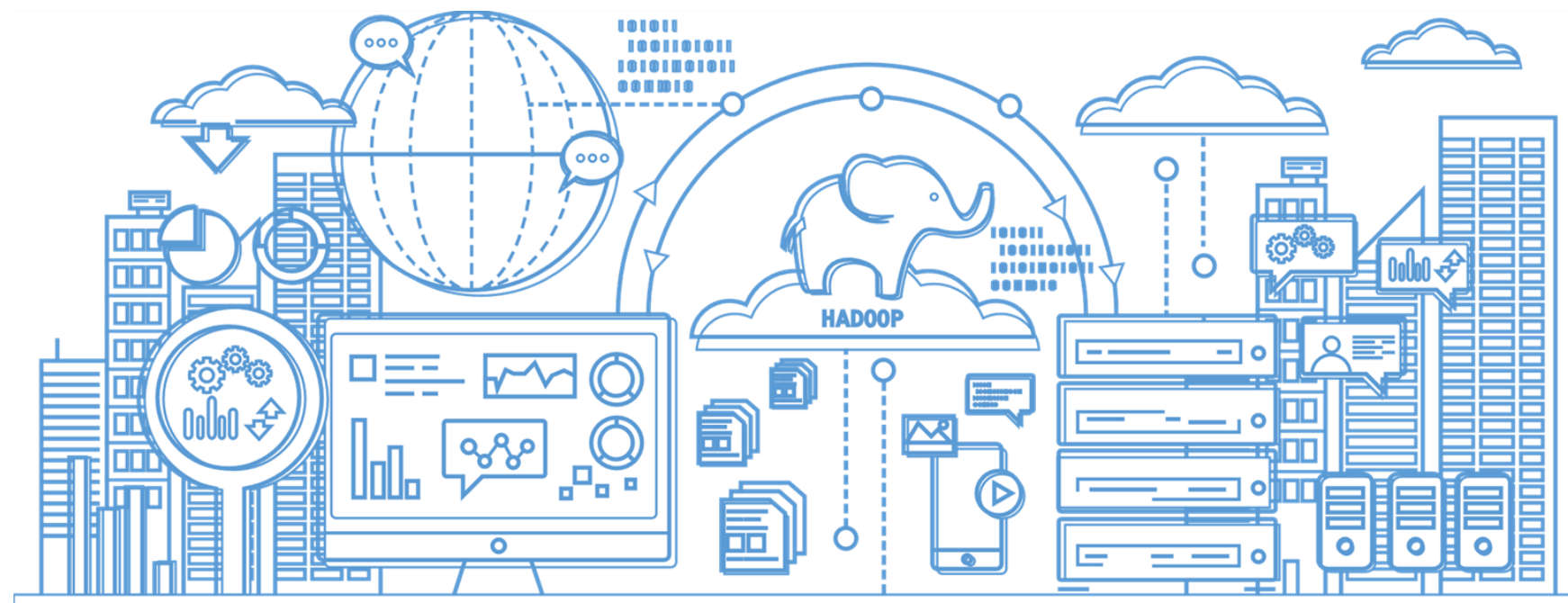


## **Case Study: Analyzing Uber Datasets Using Hadoop Framework**

# Analyzing Uber Datasets Using Hadoop Framework

Uber started by using RDBMS for storage, but there was a need for scalability and reliability after a few years, which increased by 400% every year. In 2016, there were about 100 petabytes of data, which was mixed, i.e., batch and real-time. After the previous solution failed, they began utilizing HDFS and then PRESTO.

They began using HDFS for batch and streaming analytics in a variety of applications, including fraud detection, machine learning, and ETA calculation.



## Assisted Practice: HDFS commands



**Duration: 10 Minutes**

**Problem Scenario:** Write commands to manage HDFS by creating Sample.txt, establishing the Simplilearn directory, transferring the file, displaying content, retrieving block details, and removing the file.

**Objective:** To master HDFS file management, create a text file, establish a Simplilearn directory, transfer a sample file, display its content, retrieve block details, and remove the file from HDFS.

### **Steps Overview:**

Step 1: Create a text file named **Sample.txt** in the vi editor

Step 2: Create a directory named **Simplilearn** in HDFS

Step 3: Transfer a sample text file from your local filesystem to the HDFS director

## Assisted Practice: HDFS commands



**Duration: 10 Minutes**

Step 4: Show the content of the HDFS file uploaded on the command prompt

Step 5: Get details of all replicated blocks of the sample file uploaded

Step 6: Remove the text file from the HDFS directory

**Note: The solution to this assisted practice is provided under the Reference Materials section.**



# Key Takeaways

- 👁 HDFS is a distributed file system that provides access to data across Hadoop clusters.
- 👁 The HDFS components comprise different servers, like NameNode, DataNode, and Secondary NameNode.
- 👁 HADOOP CLI is an interactive command-line shell that makes working with Hadoop's Distributed File System (HDFS) easier and more intuitive than the traditional Hadoop command-line tools.





## Knowledge Check

## Knowledge Check

1

**Which of the following statements best describes how a large (100 GB) file is stored in HDFS?**

- A. The file is replicated three times by default. Each copy of the file is stored on a separate DataNode.
- B. The master copy of the file is stored on a single DataNode. The replica copies are divided into fixed-size blocks which are stored on multiple DataNodes.
- C. The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. Multiple blocks from the same file might reside on the same DataNode.
- D. The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. HDFS guarantees that different blocks from the same file are never on the same DataNode.



## Knowledge Check

1

Which of the following statements best describes how a large (100 GB) file is stored in HDFS?

- A. The file is replicated three times by default. Each copy of the file is stored on a separate DataNode.
- B. The master copy of the file is stored on a single DataNode. The replica copies are divided into fixed-size blocks which are stored on multiple DataNodes.
- C. The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. Multiple blocks from the same file might reside on the same DataNode.
- D. The file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. HDFS guarantees that different blocks from the same file are never on the same DataNode.



---

The correct answer is **D**

---

**The large file is divided into fixed-size blocks which are stored on multiple DataNodes. Each block is replicated three times by default. HDFS guarantees that different blocks from the same file are never on the same DataNode.**

**Knowledge  
Check**  
**2**

**How many blocks are required to store a file that is 514 MB in size in HDFS using default block size configuration?**

- A. 4
- B. 5
- C. 6
- D. 7



**Knowledge  
Check**  
**2**

**How many blocks are required to store a file that is 514 MB in size in HDFS using default block size configuration?**

- A. 4
- B. 5
- C. 6
- D. 7

---

The correct answer is **B**

---

**The default block size in Hadoop 3.x is 128 MB. So, a file of size 514 MB will be divided into 5 blocks, where the first four blocks will be 128 MB and the last block will be 2 MB.**



## Knowledge Check

3

**Which of the following statements is NOT true about HDFS?**

- A. We cannot modify files already present in HDFS
- B. Multiple clients can write into an HDFS file concurrently
- C. Both A and B
- D. None of the above



## Knowledge Check

3

Which of the following statements is NOT true about HDFS?

- A. We cannot modify files already present in HDFS
- B. Multiple clients can write into an HDFS file concurrently
- C. Both A and B
- D. None of the above



---

The correct answer is **B**

---

**HDFS follows the write once, read many model. So, multiple clients cannot write into an HDFS file concurrently.**





**Thank You**