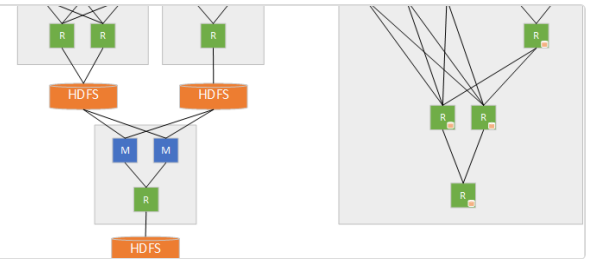


# Week 2 - Introduction to Hadoop

## Introduction to BigData, Hadoop and Spark

Everyone is speaking about Big Data and Data Lakes these days. Many IT professionals see Apache Spark as the solution to every problem. At the same time, Apache Hadoop has been around for more than 10 years and won't go away anytime soon.

<http://chaosmail.github.io/hadoop/2019/01/31/intro-to-bigdata-hadoop-and-spark/>



## Introduction to Hadoop

### What is Hadoop?

Hadoop is an open-source framework used to process enormous datasets.

Hadoop is a set of open-source programs and procedures which can be used as the framework for Big Data operations.

- processing massive data in distributed file systems that are linked together.
- allows for running applications on clusters.
  - A cluster is a collection of computers working together at the same time to perform tasks.

Hadoop is not a database but **an ecosystem that can handle processes and jobs in parallel or concurrently.**

Hadoop is optimized to handle massive quantities of data which could be:

- Structured, tabular data,
- Unstructured data, such as images and videos,
- Semi-structured data, using relatively inexpensive computers.

### How does Hadoop work?

*Data is now in petabytes and exabytes*

*Big Data is the term used to explain the complexity of the data.*

Hadoop has individual components for storing and processing data.

The term Hadoop is often used to refer to both the core components of Hadoop as well as the ecosystem of related projects.

The core components of Hadoop include: **Hadoop Common**, which is an essential part of the Apache Hadoop Framework that refers to the collection of common utilities and libraries that support other Hadoop modules.

### HDFS

There is a storage component called **Hadoop Distributed File System**, or **HDFS**.

It handles large data sets running on commodity hardware. A commodity hardware is low-specifications industry-grade hardware and scales a single Hadoop cluster to hundreds and even thousands.

### Mapreduce

A processing unit of Hadoop and an important core component to the Hadoop framework.

MapReduce processes data by **splitting large amounts of data into smaller units and processes them simultaneously.**

For a while, MapReduce was the only way to access the data stored in the HDFS. There are now other systems like Hive and Pig.

### YARN

And the last component is YARN, which is short for “**Yet Another Resource Negotiator**.”

YARN is a very important component because it prepares the RAM and CPU for Hadoop to run data in batch processing, stream processing, interactive processing, and graph processing, which are stored in HDFS.

## The challenges of Hadoop

Hadoop failed at simple tasks.

- Not good for **processing transactions** due to its lack of random access.
- Not good when the **work cannot be done in parallel** or when there are dependencies within the data. Dependencies arise when record one must be processed before record two.
- Not good for **low latency data access**. “Low latency” allows small delays, unnoticeable to humans, between an input being processed and the corresponding output providing real time characteristics. This can be especially important for Internet connections utilizing services such as trading, online gaming, and Voice Over IP.
- Not good for processing lots of small files, although there is work being done in this area such as IBM's Adaptive MapReduce.
- Hadoop is not good for intensive calculations with little data.

To deal with the shortcomings of Hadoop, new tools like Hive were built on top of Hadoop.

- Hive provided SQL-like query and provided users with strong statistical functions.
- Pig was popular for its multi query approach to cut down the number of times that the data is scanned.

## Conclusion

- Hadoop is an open-source frame framework for Big Data
- The core components of Hadoop are HDFS, MapReduce, YARN, and Hadoop Common
- The drawbacks of Hadoop outweighed the benefits.

## Intro to MapReduce

### What is MapReduce?

MapReduce is a programming pattern that enables massive scalability across hundreds or thousands of servers in a Hadoop cluster.

As the processing component, MapReduce is the heart of Apache Hadoop.

A processing technique and a program model for distributed computing, it is based on Java.



**Distributed computing** is a system or machine with multiple components located on different machines. Each component has its own job, but the components communicate with each other to run as one system to the end user.

The MapReduce algorithm consists of 2 important tasks - **Map and Reduce**.

Many MapReduce programs are written in Java. MapReduce can also be coded in C++, Python, Ruby, R and so on.

### Map and Reduce

MapReduce framework contains two tasks, Map and Reduce.

#### Map

Map takes in an input file and performs some mapping tasks by processing and extracting important data information **into a key value pairs** and these are the preliminary output list.

Some more reorganization goes on before the preliminary output is sent to the Reducer.

#### Reduce

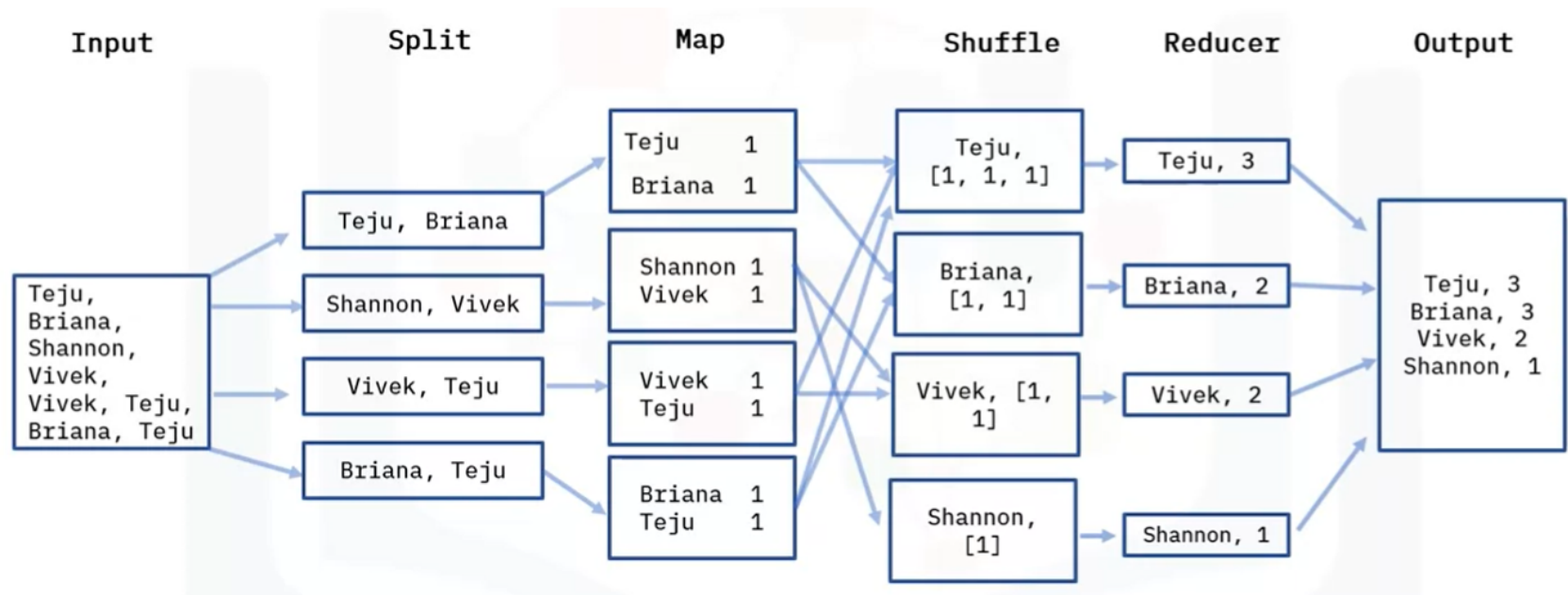
The Reducer works with multiple map functions and **aggregates the pairs using their keys** to produce a final output.

MapReduce **keeps track of its tasks by creating unique keys** to ensure that all the processes are solving the same problem.

### How MapReduce work?

- Map step: takes a set of data and converts it into another set of data, where individual elements are broken down into key/value pairs.

- The key is the name, and the value is the content.
- The input data is a file that is saved in the Hadoop file system called HDFS.



*E.G: Now let's assume we have an input file that contains names of people, and we would like to do a word count on the unique name occurrences.*

First, the data is split into the following four files, each of them in key value pair and are worked on separately

The reducer processes the data that comes from the map. After processing, it produces a new set of outputs, which will be stored in the HDFS.

**The Reducer starts with shuffling.** Shuffling sorts the key and a list of values in a list,

The Reducer layer **then aggregates the values** in the list and saves them, then the final output is saved in an output file.

## Why use MapReduce?

The advantages of MapReduce is its ability to **allow for a high level of parallel jobs across multiple nodes.**



A node is an independent computer used for processing and storing big volumes of data.

Hadoop has two types of nodes, the **name node** and the **data node**.

- Allows for splitting and running independent tasks in parallel by dividing each task which in turn saves time.
- MapReduce is very flexible and can process data that come in tabular and non-tabular forms.
  - Therefore, MapReduce provides business value to organizations regardless of how their data is structured.
- Support for different languages
- Provides a platform for analysis, data warehousing and more.

## Common use cases

MapReduce can be used for social media platforms like LinkedIn and Instagram to analyze who visited, viewed, and interacted with your profile posts.

Map reduce is used by Netflix to recommend movies based on what you have watched in the past by using the user's interests.

Used in financial institutions like banks and credit card companies to flag and detect anomalies in user transactions.

Used in the advertisement industry to understand a user's behavior by how they engage with ads.

Google ads work by using MapReduce to understand the engagement of users with an ad.

## Conclusion

- MapReduce is a framework used in Parallel Computing, it contains two major tasks: Map and Reduce.
- Map processes data into Key Value pairs, sorts and organizes the data.
- The Reducer aggregates and computes a set of results and produces a final output.

- It is flexible for all data types like structured and unstructured data and can be applied to multiple industries such as social media, entertainment, and many more.

## Hadoop Ecosystem

Apart from the core components of Hadoop, Hadoop Common refers to the common utilities and libraries that support the other Hadoop modules.

Hadoop Distributed File System (HDFS) stores the data collected from the ingestion and distributes the data across multiple nodes.

MapReduce is used for making Big Data manageable by processing them in clusters.

Yet Another Resource Negotiator (YARN) is the resource manager across clusters.

The extended Hadoop Ecosystem consists of libraries or software packages that are commonly used with or installed on top of the Hadoop core.

## Hadoop Ecosystem

The Hadoop ecosystem is made up of components that support one another for Big Data processing.



We can examine the Hadoop ecosystem based on the various stages.

- When data is received from multiple sources, Flume and Sqoop are responsible for ingesting the data and transferring them to the Storage component, HDFS and HBase.
- Then, the data is distributed to a MapReduce framework like Pig and Hive to process and analyze the data, and the processing is done by parallel computing.
- After all that is done, tools like Hue are used to access the refined data.

## Ingest data

Ingesting is the first stage of Big Data processing.

Deal with big data: get data from different sources, then use tools like Flume and Sqoop.

### Flume

- Collects, aggregates, and transfers Big Data to the storage system.
- Flume has a simple and flexible architecture based on streaming data flows
- Uses a simple extensible data model that allows for online analytic application.

### Sqoop

- An open-source product designed to transfer bulk data between relational database systems and Hadoop.
- Sqoop looks in the relational database and summarizes the schema.
- Generates MapReduce code to import and export data as needed.
- Allows to quickly develop any other MapReduce applications that use the records that Sqoop stored into HDFS.

## Store data

### Hbase

- A column-oriented non-relational database system that runs on top of HDFS.
- Provides real time wrangling access to the Hadoop file system.
- Uses hash tables to store data in indexes and allow for random access of data, which makes lookups faster.

## Cassandra

- Cassandra is a scalable, NoSQL database designed to have no single point of failure.

## Analyze data

### Pig

- Used for analyzing large amounts of data.
- A procedural data flow language and a procedural programming language that follows an order and set of commands.

### Hive

- Used mainly for creating reports and operates on the server side of a cluster.
- A declarative programming language, which means it allows users to express which data they wish to receive.

## Access data

The final stage is “Access data,” where users have access to the analyzed and refined data.

### Impala

- A scalable system that allows non-technical users to search and access the data in Hadoop.
- Not need to be skilled in programming to use Impala.

### Hue

- Hue is an acronym for *Hadoop user experience*.
- Allows you to upload, browse, and query data.
- Can run Pig jobs and workflow in Hue.
- Provides a SQL editor for several query languages like Hive, and MySQL.

## Conclusion

- The four main stages of the Hadoop Ecosystem are Ingest, Store, Process and Analyze, and Access,
  - examples of the tools associated with each stage
  - how each stage interacts with the others.
- 

## HDFS

HDFS stands for Hadoop Distributed File System.

A distributed file system is a file system that is distributed on multiple file servers and allows programmers to access or store files from any network or computer.

- HDFS is the storage layer of Hadoop.
- HDFS works by splitting the files into blocks, then creating replicas of the blocks, and storing them on different machines.
- HDFS is built to access streaming data seamlessly.
- HDFS uses a command line interface to interact with Hadoop.



Streaming means that HDFS provides a constant bitrate when transferring data rather than having the data being transferred in waves.

## Key features



Cost efficient	The storage hardware is not expensive
Large amounts of data	HDFS can store up to petabytes of data
Replication	Makes copies of the data on multiple machines
Fault tolerant	If one machine crashes, a copy of the data can be found somewhere else and work continues
Scalable	One cluster can be scaled into hundreds of nodes
Portable	Can easily move across multiple platforms

- The commodity hardware that stores the data is **not expensive**, and therefore reduces storage costs.
- HDFS can store very **large amounts of data**, as large as petabytes - in any format, tabular and non-tabular. It splits these large amounts of data into small chunks called blocks.
- Ability to **replicate** and minimize the costs associated with data losses when there is failure with one of the hardware units.
- That capability makes HDFS **fault tolerant**. In the event of a data loss of one of the computers, the data can be found on another computer and work continues.
- HDFS is also highly **scalable**. A single cluster can scale into hundreds of nodes.
- **Portability** is also one of the key features, as HDFS is designed to easily move from one platform to another.

## HDFS concepts

### Blocks

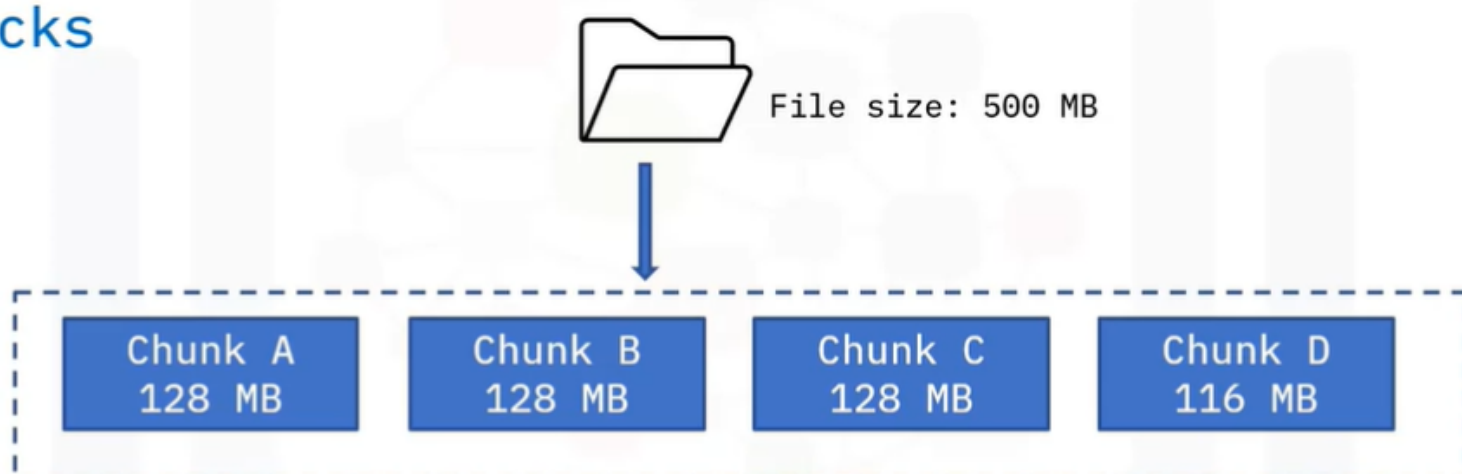
When HDFS receives files, files are broken into smaller chunks called blocks.

- A block is the minimum amount of data that can be read or written and provides fault tolerance.
- Depending on your system configuration, the default block size could be 64 or 128 megabytes



For example, if we had a 500MB file, with a default block chunk size of 128MB, the file will be divided into 3 blocks of 128MB and one block of 116MB.

### Blocks



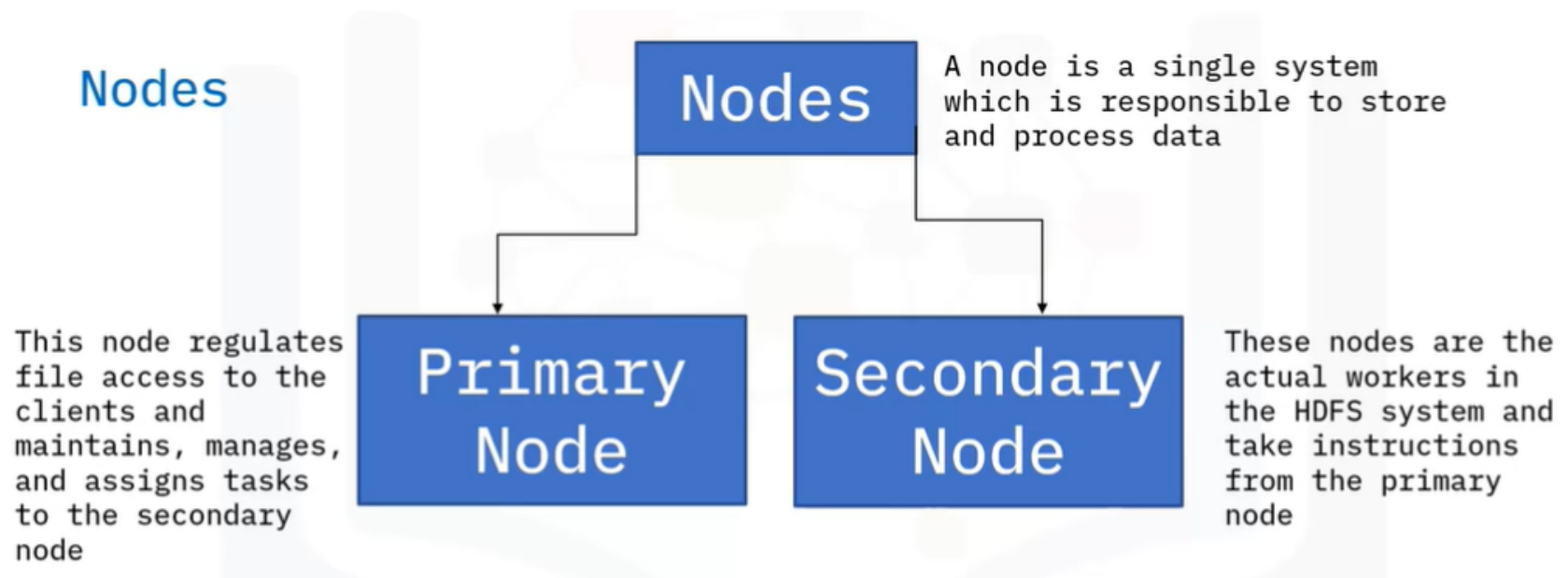
The only time you will have equal splits is if the file size is a multiple of the default block size.

→ Each file stored doesn't have to take up the storage of the pre-configured block size.

### Nodes

A node is a single system which is responsible for storing and processing data.

Think about it as one machine or computer in which data is stored. Remember that HDFS follows the primary/secondary concept.



HDFS has two types of nodes:

- The **Primary node**, known as the **name node**, regulates file access to the clients and maintains, manages, and assigns tasks to the secondary node, also known as a **data node**.
- The **Secondary node** perform read and write requests at the instruction of the name node There can be hundreds of data nodes in the HDFS that manage the storage system.

When performing operations like read and write, it is important that the name node maximizes performance by choosing the data nodes closest to themselves.

This could be by choosing data nodes on the same rack or in nearby racks. This is called rack awareness.

## Rack awareness in HDFS

*A rack is the collection of about 40-50 data nodes using the same network switch.*

- Reduce the network traffic and improve cluster performance.
- The name node keeps the rack ID information.
- Replication is done by rack awareness as well.

It is done by making sure replicas of a data node are in different racks. So, if a rack is down, you can still obtain the data from another rack.

## Replication

HDFS is known for optimizing replication. HDFS uses the rack awareness concept to create replicas to make sure that the data is reliable and available, and that the network bandwidth is properly utilized.

Replication is:

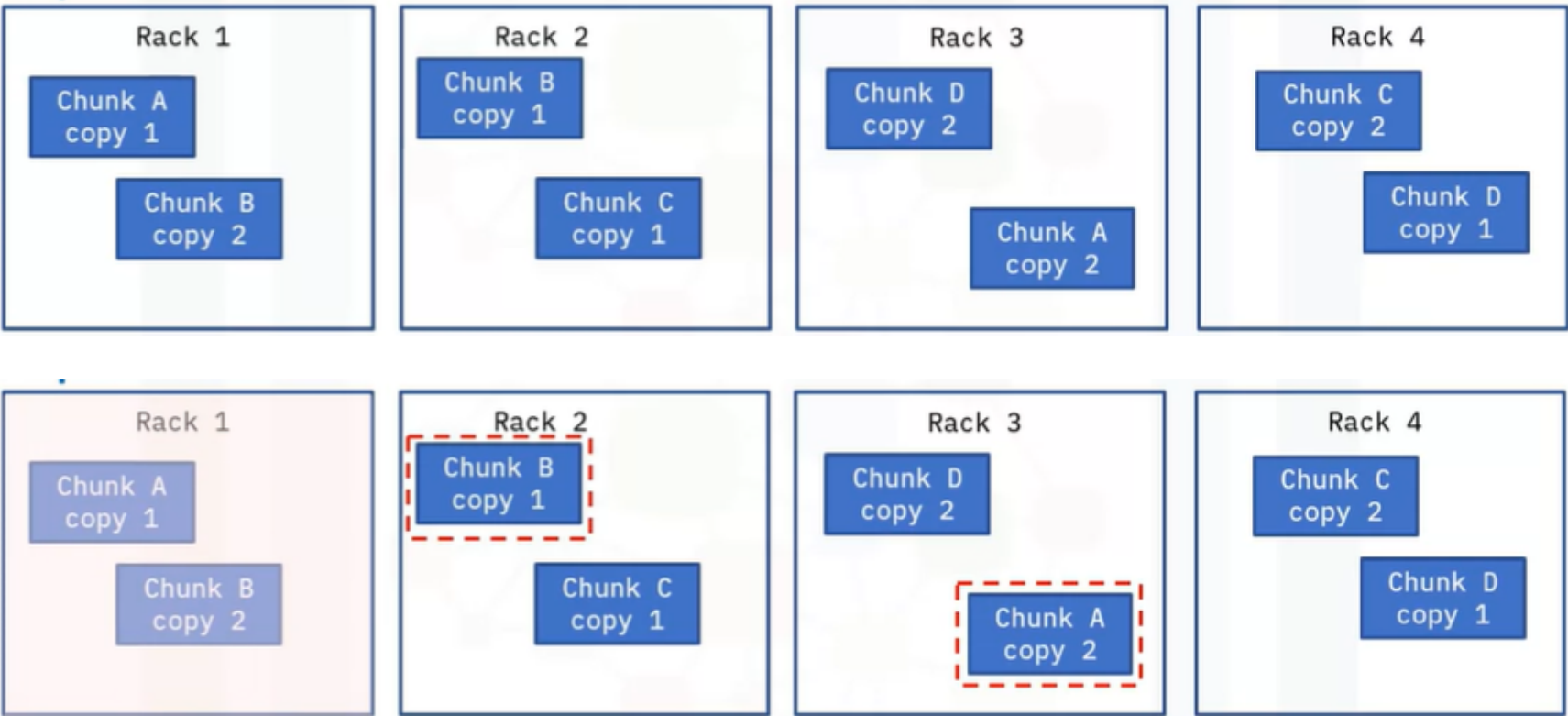
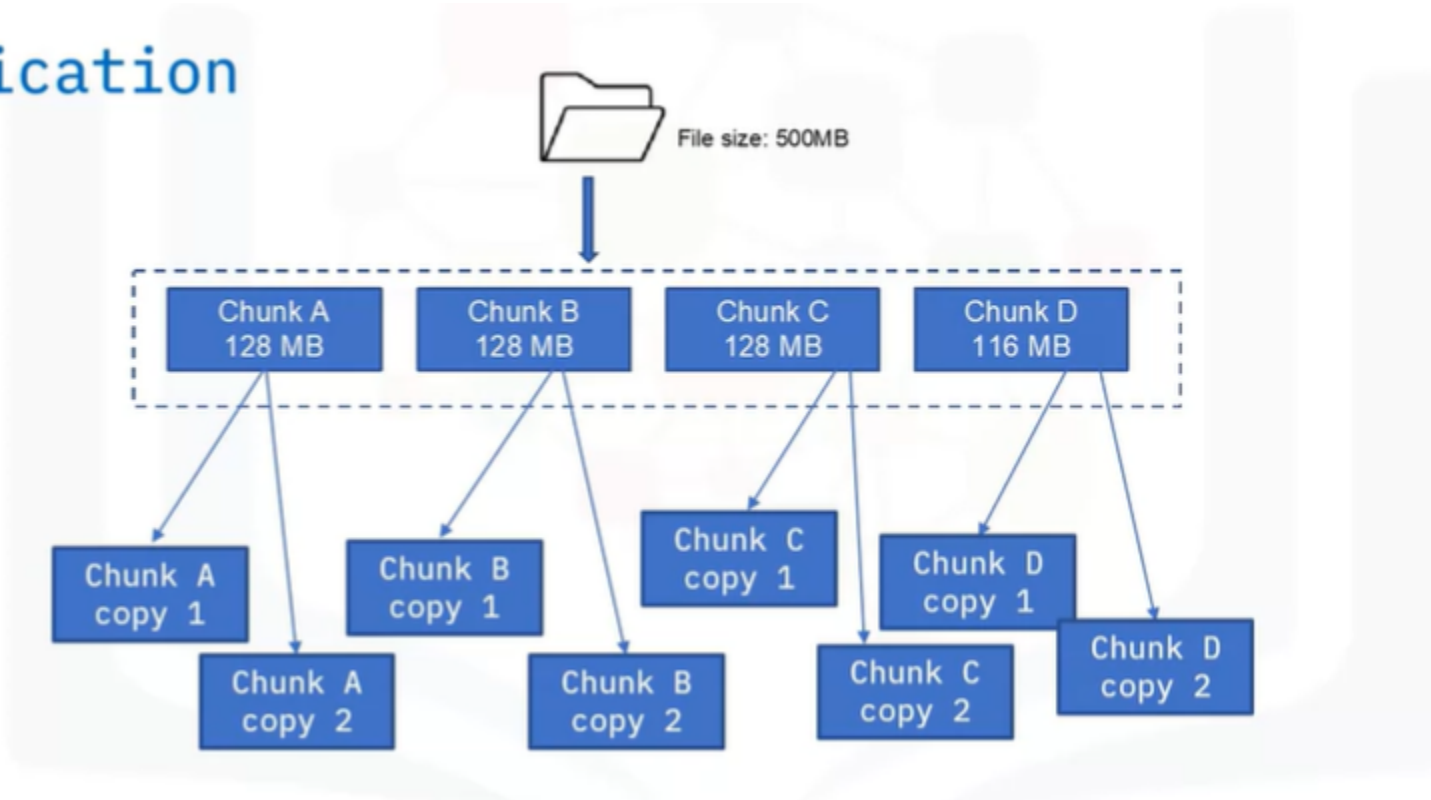
- Creating a copy of the data block.
- When crashes happen, replication provides backup of the data blocks.
- **Replication factor** - number of times you make a copy of the data block.

HDFS uses the rack awareness concept and saves the blocks in different racks to make sure that a copy is available in another rack.



Let us look at our 500MB file example. If our replication factor is 2, it will create two copies of each block. That means we will have 4 times 2, which is eight copies of the file blocks for backup.

# Replication

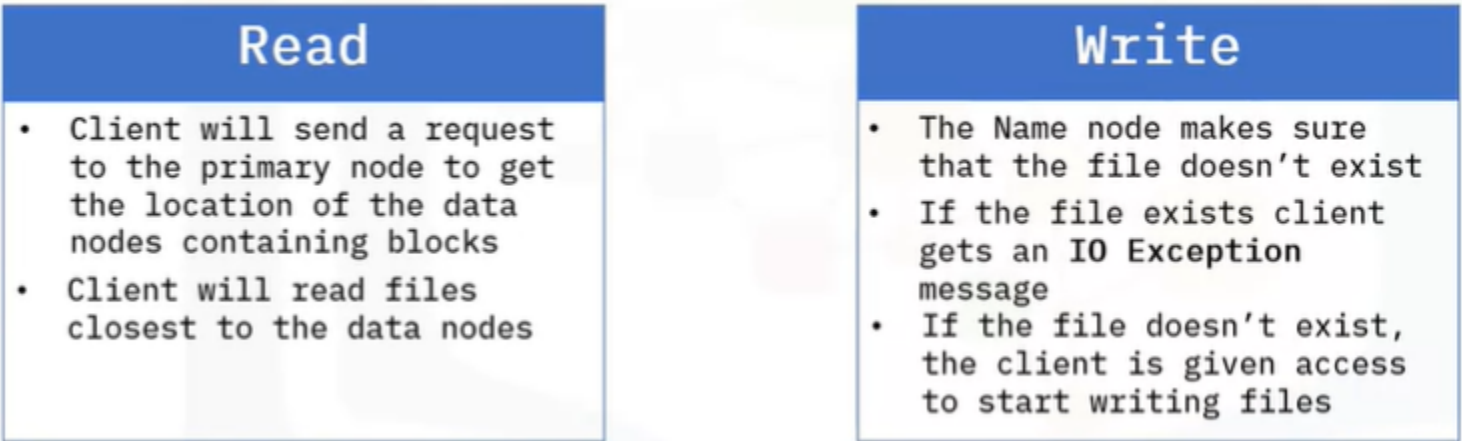


For example, you could have four racks of two copies each. If one of the racks is down or crashes, you will still have a copy of the data blocks in a different rack, and you can still work with the data.

## Read and Write operations

HDFS allows “write once, read many” operations.

→ cannot edit files that are already stored in HDFS, but you can append new data to them.



### Read

Assuming we have a text file.

Client will send a request to the primary node to get the location of the data nodes containing blocks.

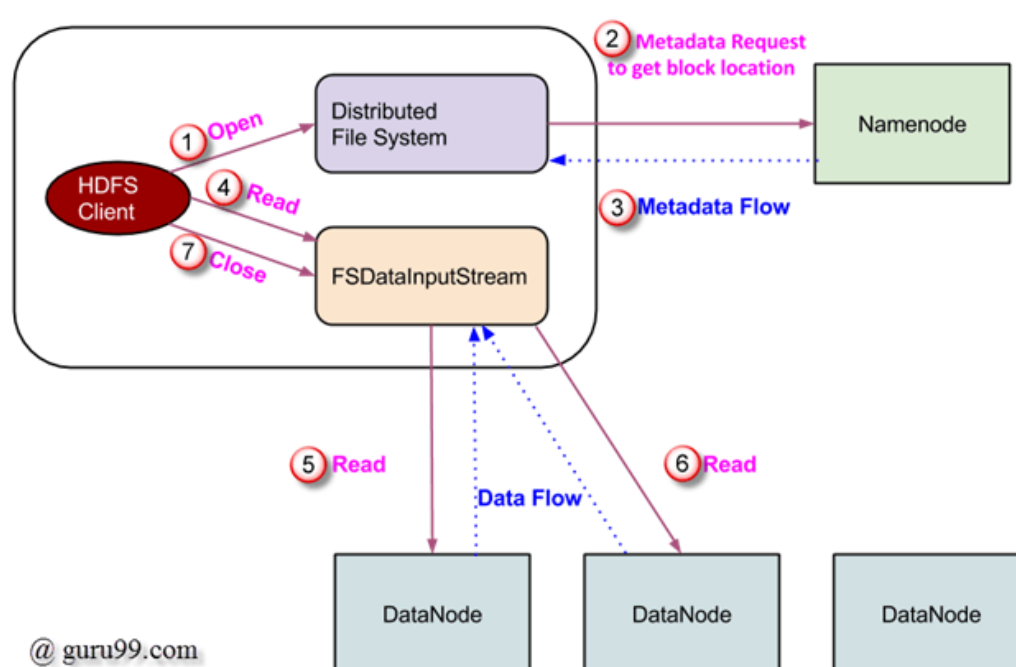
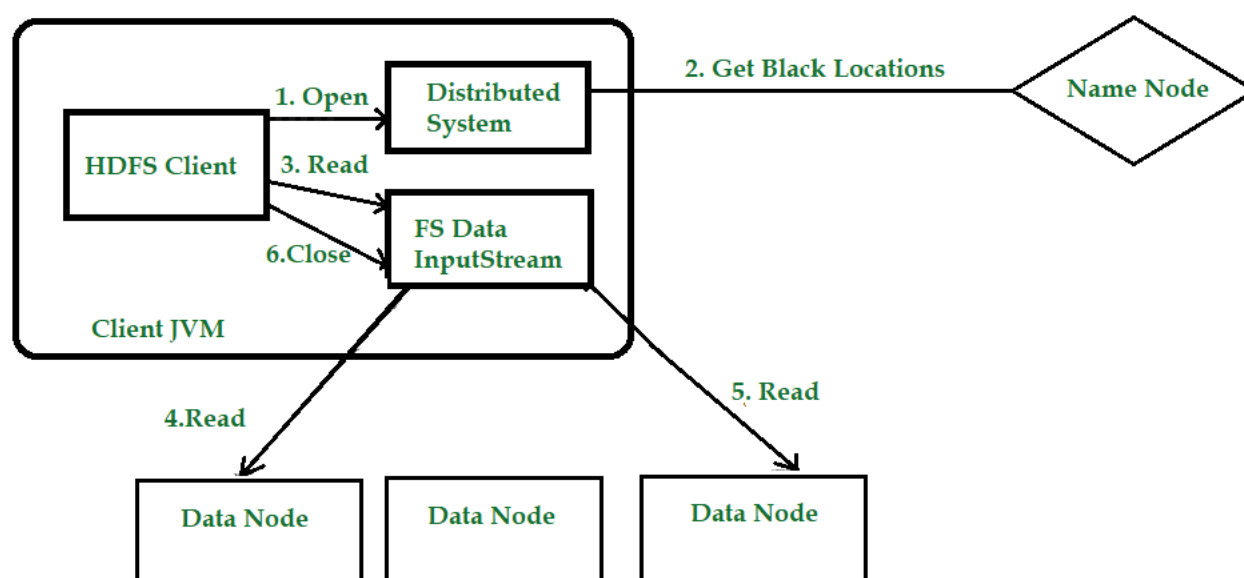
The name node will verify that the client has the correct privileges and provide the client with the locations.



A client in HDFS interacts with the primary and secondary nodes to fulfill a user's request.

The client will then send a request to the closest data nodes through an FS Data Input stream object by calling the read method to read all the files.

And when the client is done, the client will use the close method to end the session.



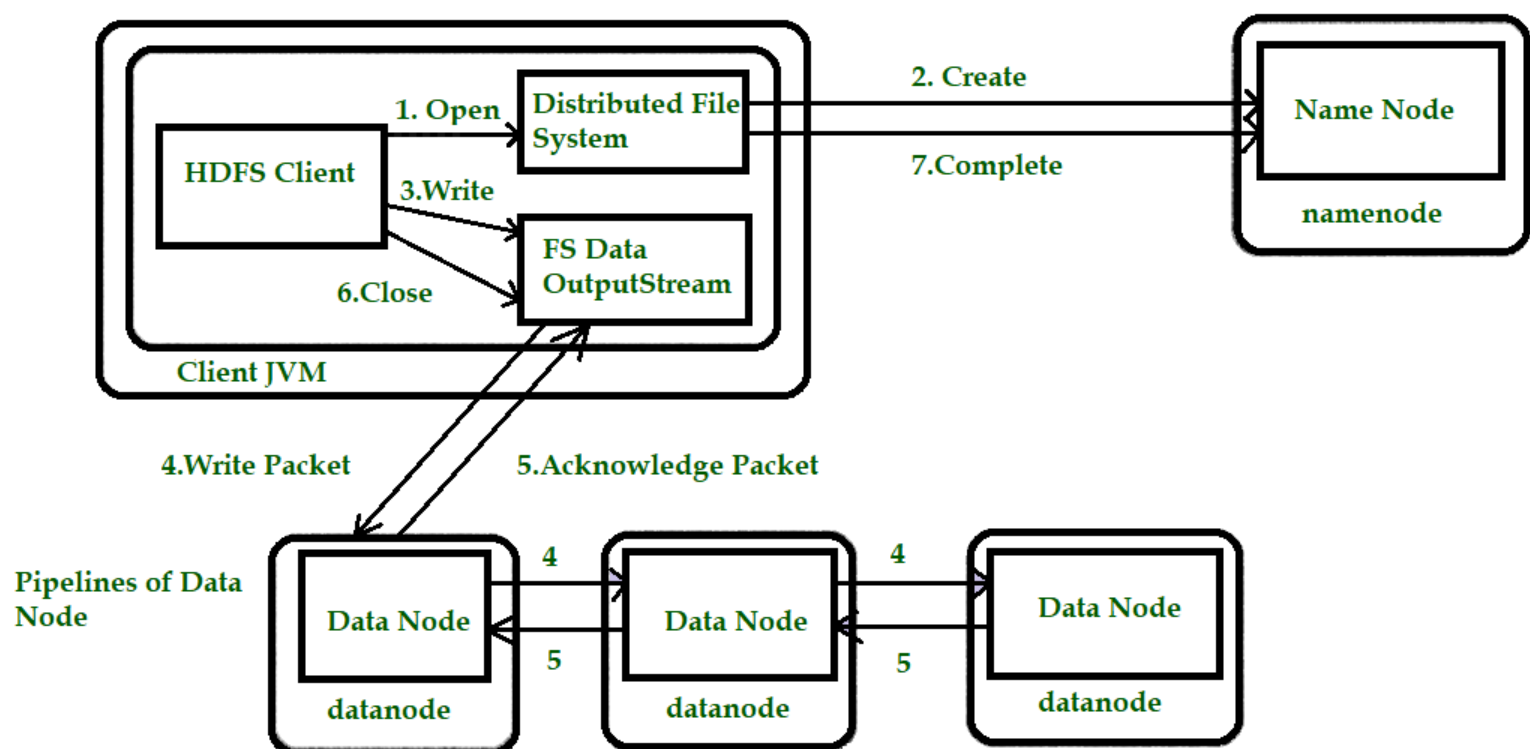
## Write

The name node confirms that the client has the right privileges.

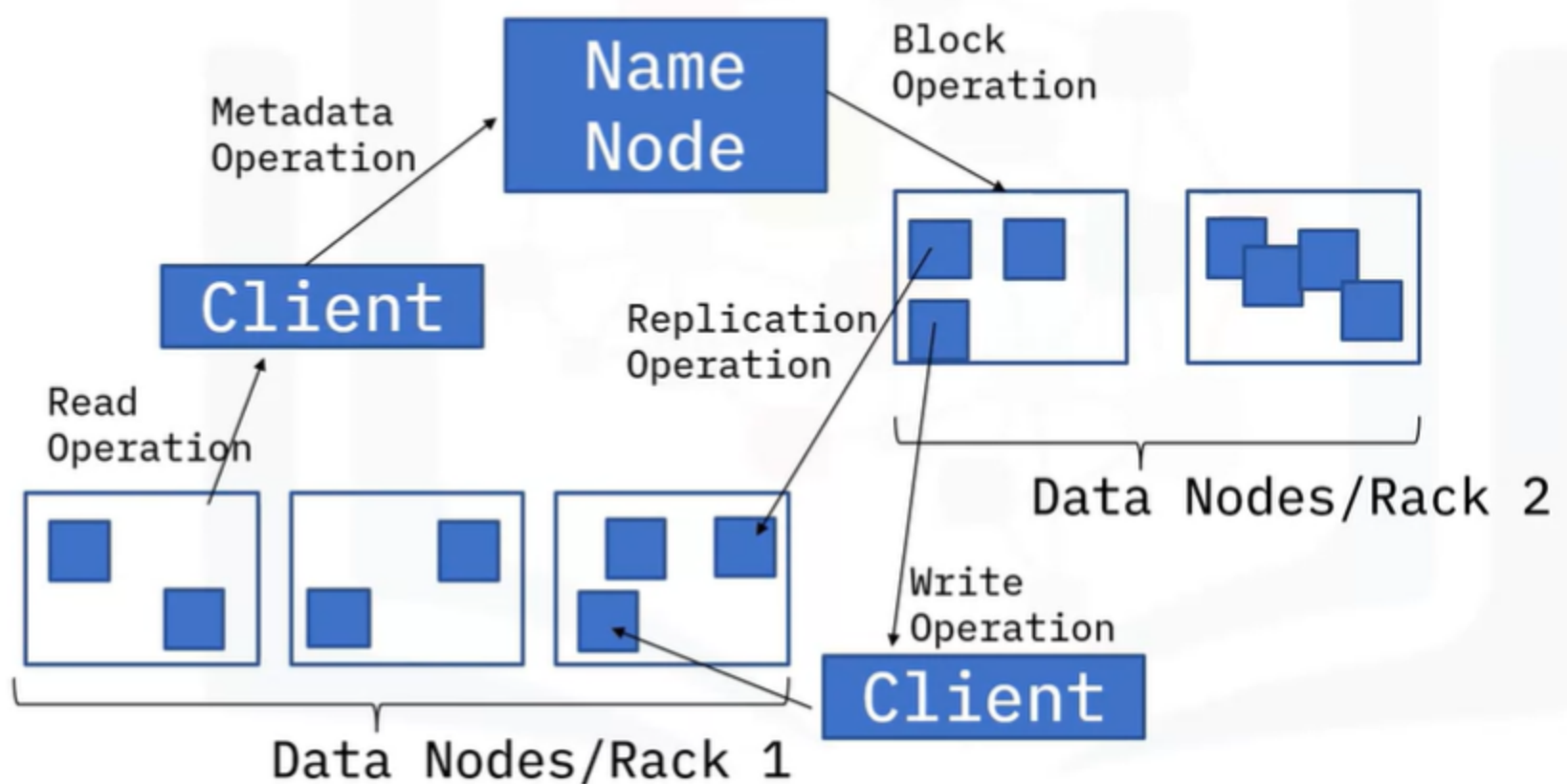
The name node makes sure to check that the file doesn't exist in the system.

- If the file already exists, the client will receive an IO exception.
- If the file doesn't exist, the client receives a write permission together with the data nodes.

Once the client is done, the data nodes start creating replicas and sends a confirmation to the client.



## HDFS Architecture



Hadoop follows the concept of a primary/secondary node architecture. The primary node is the name node.

The architecture is such that per cluster, there is one name node and multiple data nodes, which are the secondary nodes.

Internally, a file is split into one or more blocks and these blocks are stored in a set of data nodes.

The name node oversees opening, closing, renaming file operations, and mapping file blocks to the data node.

The data nodes are responsible for read and write requests from the client and perform the creation, replication, and deletion of file blocks based on instructions from the name node.

## Conclusion


- Key HDFS benefits include its cost efficiency, scalability, data storage expansion, and data replication capabilities.
- A block is the minimum size of a file.
- Replication creates a copy of data blocks for backup purposes.
- Rack awareness helps reduce the network traffic and improve cluster performance.
- HDFS enables write once, read many operations.

---

# Hive


## Hive


Hive is a data warehouse software within Hadoop that is designed to read, write, and manage large and tabular-type datasets and data analysis.

 A data warehouse stores historical data from many different sources so that you can analyze and extract insights from it. These insights are used for reporting.

- Hive is scalable and fast because it is designed to work on petabytes of data.
- It is easy to use if you are familiar with SQL because Hive query language, or Hive QL, is based on SQL. This makes learning quicker because users who already deal with relational databases can easily grasp the concepts.
- Hive supports the following file formats:
  - Sequence files consisting of. binary key value pairs,
  - Record columnar files that store columns of a table in a columnar database,
  - Text or flat files.
- Hive allows for data cleaning and filtering tasks according to users' requirements.

## Hive and traditional RDBMS compared

 RDBMS is a type of database management system specifically designed for relational databases. RDBMS is the acronym for Relational Database Management System. A relational database is a database that stores data in a structured format using rows and columns known as a table.

 Partitioning means dividing the table into parts based on the values of a particular column such as date or city

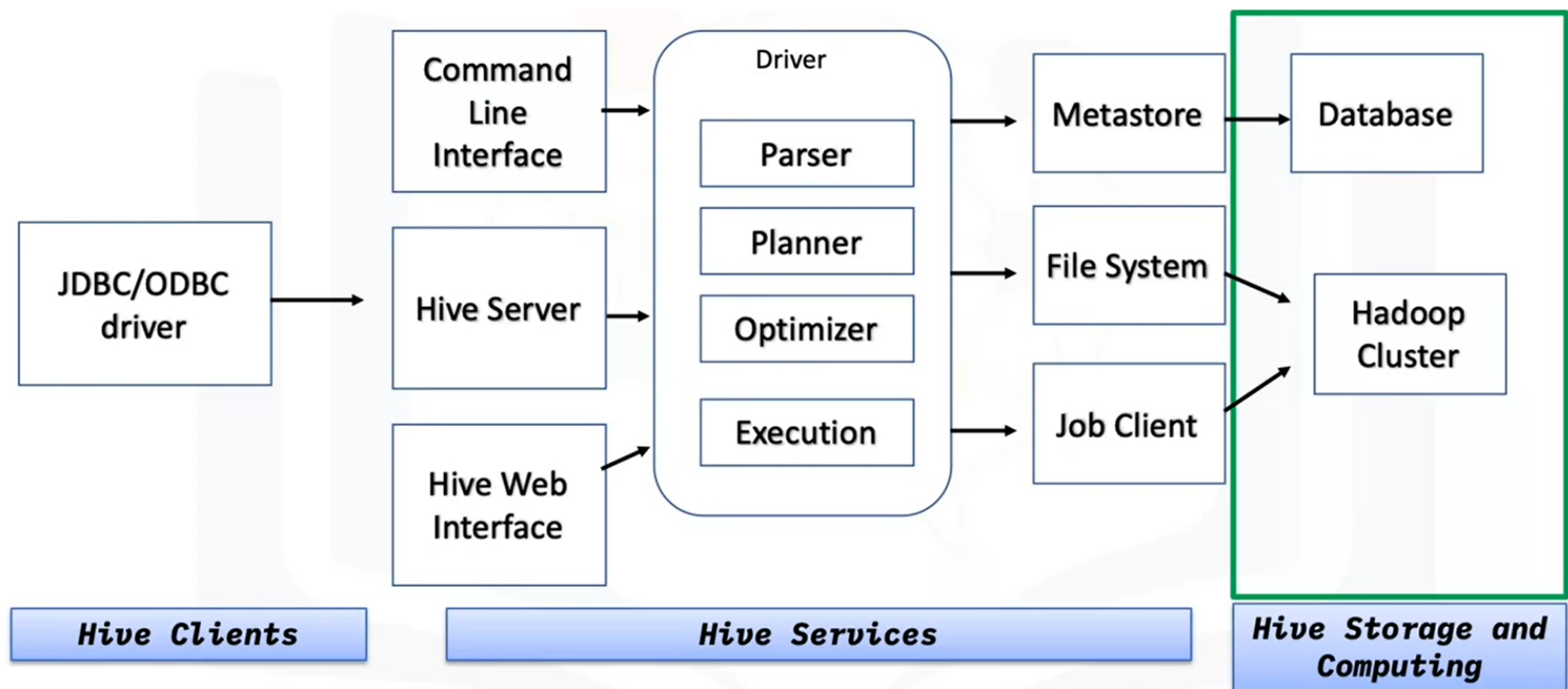
### Traditional RDBMS

Used to maintain a database and uses SQL  
Suited for real-time data analysis, like data from sensors  
Allows for as many read and write operations a user needs  
Handle up to terabytes of data  
Enforces that the schema must verify loading data before it can proceed  
not always have built-in for support data partitioning

### Hive

Used to maintain a DWH and uses the HQL  
Suited for static data analysis  
Based on the write once, read many methodology  
Handle as much as petabytes of data  
Doesn't enforce the schema to verify loading data  
Supports partitioning

## Hive architecure



There are three main parts of the architecture:

- **The Hive client** - Hive provides different drivers for communication depending on the type of application. For example, for Java based applications, it uses JDBC drivers, and other types of applications will use ODBC drivers. These drivers communicate with the servers.
- **Hive services** - Client interactions are done through the Hive services. Any query operations are done here.
  - The **command line interface** acts as an interface for the Hive service.
  - The **driver** takes in query statements, monitors the progress and life cycle of each session, and stores metadata generated from the query statements.
  - The **meta store** stores the metadata, the data and information about each table, such as the location and schema.
  - **The meta store, file system, and job client** in turn communicate with Hive storage and computing
- **Hive storage and computing** - perform the following: Metadata information of tables are stored in some sort of database and query results and data loaded in the tables are stored in Hadoop cluster or HDFS.

## Hive concepts

### Hive client - JDBC/ODBC driver

- The JDBC client, which allows Java-based applications to connect to Hive
- The ODBC client, which allows applications based on the ODBC protocol to connect to Hive.

### Hive services

The Hive server is used to run queries and allows multiple clients to submit requests.

It is built to support the JDBC and ODBC clients.

**The driver receives query statements submitted through the command line** and sends the query to the compiler after initiating a session.

The **optimizer performs transformations on the execution plan and splits the tasks** to help speed up and improve efficiency.

The **executor executes tasks** after the optimizer has split the tasks.

The **meta store is storage for the metadata**, which is information about the table.

The meta store is in charge of keeping this information in a central place.

## Conclusion

- Hive is a data warehouse software for reading, writing, and managing datasets.
- Although Hive works very similarly to traditional RDBMS, they are slightly different.
- The three main parts of the Hive architecture are *Hive Client*, *Hive Services*, and *Hive Storage and Computing*



# HBASE

## HBase

- HBase is a column-oriented non-relational database management system
- runs on top of HDFS.
- Provides a fault-tolerant way of storing sparse data sets.
- It works well with real-time data and random read and write access to Big Data.



Fault tolerance refers to the working ability of a system or computer to continue working even in unfavorable conditions such as when a server crashes.

## HBase

HBase is used for write-heavy applications, to store large amounts of datasets, and process and provide analytics in real time.

- It is referred to as linearly scalable because it supports scalability in both linear and modular form.
- HBase is a backup support for MapReduce jobs.
- HBase can be used for high-speed requirements because it offers consistent reads and writes.
- HBase has no fixed columns and is only defined by column families.
- It provides an easy-to-use Java API for client access.
- HBase supports data replication across clusters.

## Example

An HBase column represents an attribute of an object.

Patient Details		Heart Rate	Time Stamp
Patient Name	Patient Age	Heart Rate	Time Stamp
Patient A	28	120 BPM	8:50 AM
Patient A	28	110 BPM	10:10 AM
Patient B	77	95 BPM	11:00 AM
Patient C	45	150 BPM	11:30 AM
⋮			
Patient B	77	115 BPM	12:30 AM

The table is storing sensors from heart monitors in the hospital, each row might be a log record.

A typical column could contain information about the patient’s details or the time when the log record was taken.

HBase allows for many attributes to be grouped together into **column families** such that the elements of a column family are all stored together.

For example, “Patient Name” and “Patient Age” will be stored as ”Patient Details.” The columns in HBase will look like – Patient Details, Heart Rate, and Time Stamp.

Note:

- When creating schemas in HBase, you must predefine the table schema and specify the column families.
- New columns can be added to families at any time,
- making the schema flexible and adaptable to changing application requirements.
- A primary node manages the cluster and a region server stores portions of the tables and performs the work on the data.

## Differences between HBase and HDFS

HBase and HDFS are both used to store massive amounts of data

### HBase

stores data in the form of columns and rows in a table

allows dynamic changes

suitable for read, write, update, and delete type workloads on a record level

allows for storing and processing of Big Data

### HDFS

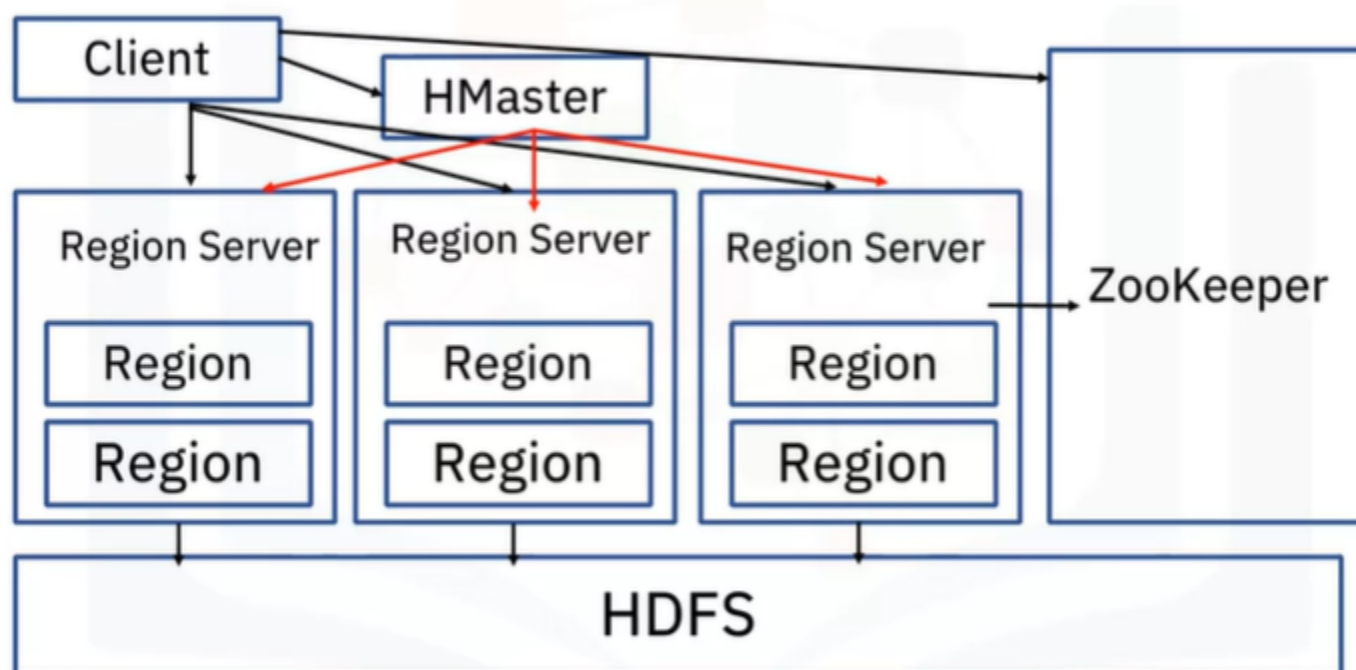
stores data in a distributed manner across different nodes on that network

has a rigid architecture that doesn't allow changes

suited for write once and read in batch workloads

is for storing only

## HBase architecture



The HBase architecture consists of four components:

1. HMaster
2. HRegionserver
3. HRegions
4. ZooKeeper

From the architecture, we can see that HBase sits and works on top of HDFS.

HDFS provides a distributed environment for the storage, and it is a file system designed to run on commodity hardware. It stores each file in multiple blocks, and to maintain fault tolerance, it replicates the blocks across a Hadoop cluster.

## HBase concepts

### HMaster

HMaster is the master server.

- Monitors the region server instances
- Assigns regions to region servers, and distributes services to different region servers,
- Manages any changes that are made to the schema and metadata operations.

### Region Servers

- Receive read and write requests from the client and assign the request to a specific region where the column family resides.
- Responsible for serving and managing regions that are present in a distributed cluster.
- Communicate directly with the client to facilitate requests.

### Region

- The basic building element and smallest unit of the HBase cluster that consist of column families.
- Contains multiple stores, one for each column family

- Two components – HFile and Memstore.

### ZooKeeper

ZooKeeper is a centralized service for maintaining configuration information to maintain healthy links between nodes.

Provides synchronization across distributed applications

Used to track server failure and network partitions by triggering an error message, and then starts repairing the failed nodes.

## Conclusion

- HBase is a column-oriented non-relational database management system that runs on HDFS.
- HBase columns are defined by column families.
- HBase is linearly scalable, highly efficient and provides an easy-to-use Java API for client access.
- A key difference between HDFS and HBase is that HBase allows dynamic changes compared to the rigid architecture of HDFS
- HBase architecture consists of HMaster, Region servers, Region, Zookeeper and HDFS.

---

## Labs

**Hadoop MapReduce:** <https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-BD0225EN-SkillsNetwork/labs/HadoopMapReduce.md.html>

**Hadoop Cluster:** <https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-BD0225EN-SkillsNetwork/labs/SettingUpClusterNode.md.html>

---

## Summary & Highlights

In this module, you learned that:

- Hadoop is an open-source framework for Big Data that faced challenges when encountering dependencies and low-level latency.
- MapReduce, a parallel computing framework used in parallel computing, is flexible for all data types, addresses parallel processing needs for multiple industries and contains two major tasks, “map” and “reduce.”
- The four main stages of the Hadoop Ecosystem are Ingest, Store, Process and Analyze, and Access.
- Key HDFS benefits include its cost efficiency, scalability, data storage expansion and data replication capabilities. Rack awareness helps reduce the network traffic and improve cluster performance. HDFS enables “write once, read many” operations.
- Suited for static data analysis and built to handle petabytes of data, Hive is a data warehouse software for reading, writing, and managing datasets. Hive is based on the “write once, read many” methodology, doesn’t enforce the schema to verify loading data and has built-in partitioning support.
- Linearly scalable and highly efficient, HBase is a column-oriented non-relational database management system that runs on HDFS and provides an easy-to-use Java API for client access. HBase architecture consists of HMaster, Region servers, Region, Zookeeper and HDFS. A key difference between HDFS and HBase is that HBase allows dynamic changes compared to the rigid architecture of HDFS.

---

## Quiz

### Practice quiz: Introduction to Hadoop

#### Question 1

Which of the following Hadoop characteristics best describes the following statement: “Hadoop is an ecosystem that can handle processes and jobs in parallel or concurrently”?

- Set of open-source programs and procedures which can be used as the framework for Big Data operation

- Collection of computers working together at the same time to perform tasks
- **Hadoop allows for running applications on clusters**
- Processes massive amounts of data in distributed files systems that are linked together.

Correct, clusters are collections of computers working together at the same time to perform tasks.

## Question 2

MapReduce is a programming model used in Hadoop for processing Big Data. It's also a processing technique for what?

- Python
- System with multiple components
- Java
- **Distributed computing**

Correct, MapReduce is a processing technique for distributed computing.

## Question 3

Four external components make up the Hadoop ecosystem. Which external component uses the Pig and Hive tools?

- **Process and analyze**
- Ingest data
- Access data
- Store data

Correct, this component uses Pig and Hive to process and analyze the data.

## Question 4

The HDFS video shows six key features for HDFS. Which features ensure against data loss?

- **Fault tolerant**

Correct, this feature ensures that if one machine crashes, many other data copies are available somewhere else, so work continues.

- Portable
- Scalable
- **Replication**

Correct, this feature copies data onto multiple machines to provide replication.

## Question 5

Which of the following file formats does Hive support?

- **Sequence file**

Correct, Hive supports sequence files consisting of binary key value pairs.

- Structured format
- **Record columnar files**

Correct, Hive supports record columnar files that store columns of a table in a columnar database.

- **Flat files**

Correct, Hive supports both flat and text files.

# Graded Quiz: Introduction to Hadoop

## Question 1

MapReduce has two tasks, map and reduce. Together, what do these two tasks do?

- Sort data



- **Creates a unique key**
- Organize data
- Aggregates and computes

MapReduce keeps track of its task by creating a unique key.

## Question 2

Which external component is the first stage of operation in the Hadoop ecosystem?

- Access data
- Process and analyze
- **Ingest data**
- Store data

Correct, this is the first stage of Big Data processing in the Hadoop Ecosystem.

## Question 3

Nodes are single systems that are responsible for storing and processing data. Which node is known as a data node in HDFS?

- Name node
- Primary node
- Storage node
- **Secondary node**

Correct, the secondary node is also known as a data node.

## Question 4

What is the maximum data size Hive can handle?

- Gigabyte
- Terabyte
- **Petabyte**
- Unlimited

Correct, the maximum data size it can handle is petabytes.

## Question 5

Which HBase component has 2 sub components - HFile and Memstore?

- Region server
- Zookeeper
- HMaster
- **Region**

Correct, Region has 2 components - HFile and Memstore