Department of CSE

COURSE NAME: DBMS
COURSE CODE:23AD2102R

Topic: Distributed Storage and Processing Framework (Hadoop)

Session – 3

AIM OF THE SESSION

To familiarize students with the basic concept of BigData

INSTRUCTIONAL OBJECTIVES

This Session is designed to: discuss and study the concepts of BigData

Distributed Storage and Processing Framework (Hadoop)

LEARNING OUTCOMES

At the end of this session, you should be able to: understand Hadoop Framework

Distributed Storage and Processing Framework (Hadoop)

- **Distributed storage** is a method of storing data across multiple devices connected by a network, instead of on a single server. It's a software-defined system that allows data to be accessed from anywhere, by anyone, and whenever needed.
- **A framework** is a tool that provides a set of templates and functions to help developers build applications. Frameworks can include libraries, compilers, code libraries, toolsets, and APIs.
- **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.

Hadoop has several key features that make it well-suited for big data processing:

- **Distributed Storage**: Hadoop stores large data sets across multiple machines, allowing for the storage and processing of extremely large amounts of data.
- **Scalability:** Hadoop can scale from a single server to thousands of machines, making it easy to add more capacity as needed.
- **Fault-Tolerance**: Hadoop is designed to be highly fault-tolerant, meaning it can continue to operate even in the presence of hardware failures.
- **Data locality:** Hadoop provides data locality feature, where the data is stored on the same node where it will be processed, this feature helps to reduce the network traffic and improve the performance.

•**High Availability**: Hadoop provides High Availability feature, which helps to make sure that the data is always available and is not lost.

• **Flexible Data Processing**: Hadoop's MapReduce programming model allows for the processing of data in a distributed fashion, making it easy to implement a wide variety of data processing tasks.

• **Data Integrity**: Hadoop provides built-in checksum feature, which helps to ensure that the data stored is consistent and correct.

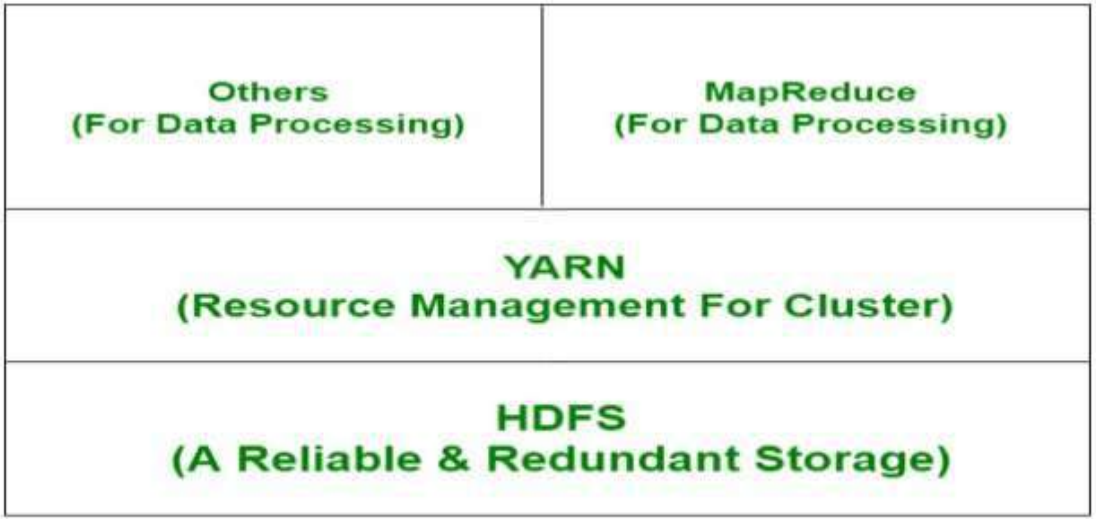
• **Data Replication**: Hadoop provides data replication feature, which helps to replicate the data across the cluster for fault tolerance

•**Data Compression**: Hadoop provides built-in data compression feature, which helps to reduce the storage space and improve the performance.

• **YARN:** A resource management platform that allows multiple data processing engines like real-time streaming, batch processing, and interactive SQL, to run and process data stored in HDFS

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.



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.

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**

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.

Big Data and Its Challenges

Big Data refers to the massive amount of data that cannot be stored, processed, and analyzed using traditional ways.

The main elements of Big Data are:

- Volume - There is a massive amount of data generated every second.
- Velocity - The speed at which data is generated, collected, and analyzed
- Variety - The different types of data: structured, semi-structured, unstructured
- Value - The ability to turn data into useful insights for your business
- Veracity - Trustworthiness in terms of quality and accuracy

Who Uses Hadoop?

Hadoop is a popular big data tool, used by many companies worldwide. Here's a brief sample of successful Hadoop users:

- British Airways
- Uber
- The Bank of Scotland
- Netflix
- The National Security Agency (NSA), of the United States
- The UK's Royal Mail system
- Expedia
- Twitter

Now that we have some idea of Hadoop's popularity, it's time for a closer look at its components to gain an understanding of what is Hadoop.

COMPONENTS OF HADOOP

Hadoop is a framework that uses **distributed storage** and **parallel processing** to store and manage Big Data. It is the most commonly used software to handle Big Data. There are **three** components of Hadoop.

1.Hadoop HDFS - Hadoop Distributed File System (HDFS) is the storage unit of Hadoop.

2.Hadoop MapReduce - Hadoop MapReduce is the processing unit of Hadoop.

3.Hadoop YARN - Hadoop **YARN** is a resource management unit of Hadoop.

Let us take a detailed look at Hadoop HDFS in this part of the What is Hadoop article.

Hadoop HDFS

- Data is stored in a distributed manner in HDFS.
- There are two components of HDFS - **name node** and **data node**
- While there is only **one name node**, there can be **multiple data nodes.**
- HDFS is specially designed for storing huge datasets in commodity hardware

Features of HDFS

•Provides distributed storage

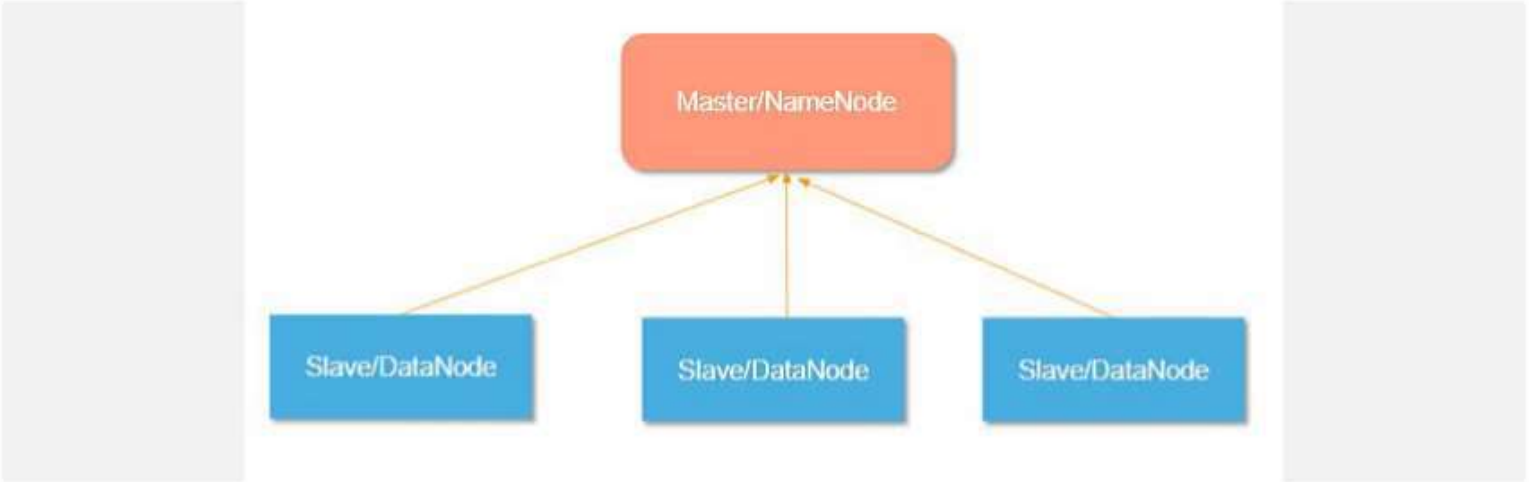
•Can be implemented on commodity hardware

•Provides data security

•Highly fault-tolerant - If one machine goes down, the data from that machine goes to the next machine

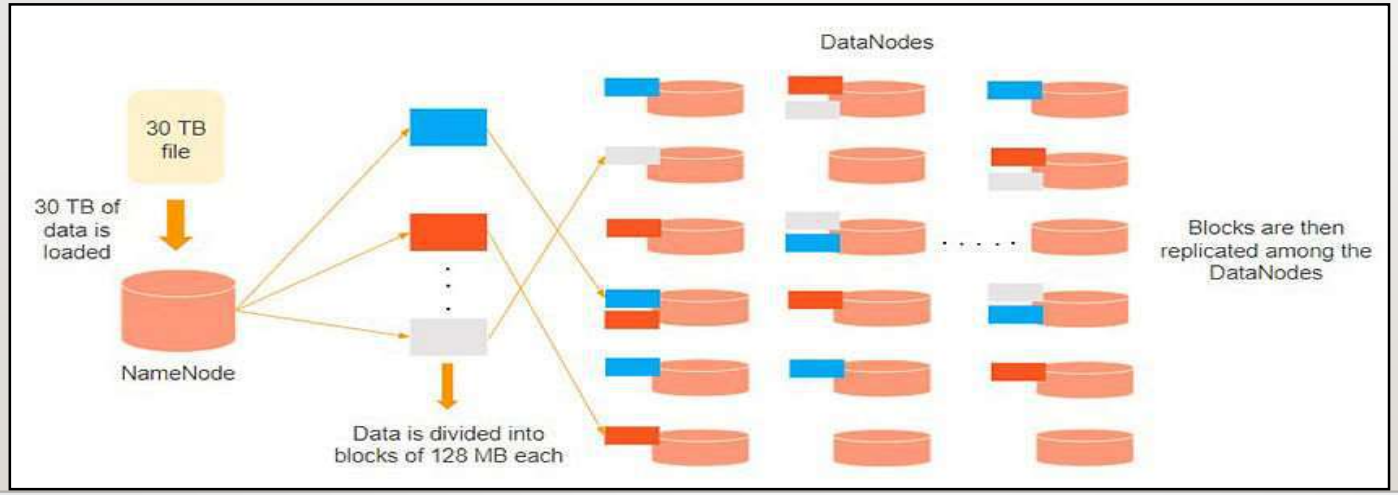
Master and Slave Nodes

Master and slave nodes form the HDFS cluster. The name node is called the master, and the data nodes are called the slaves.



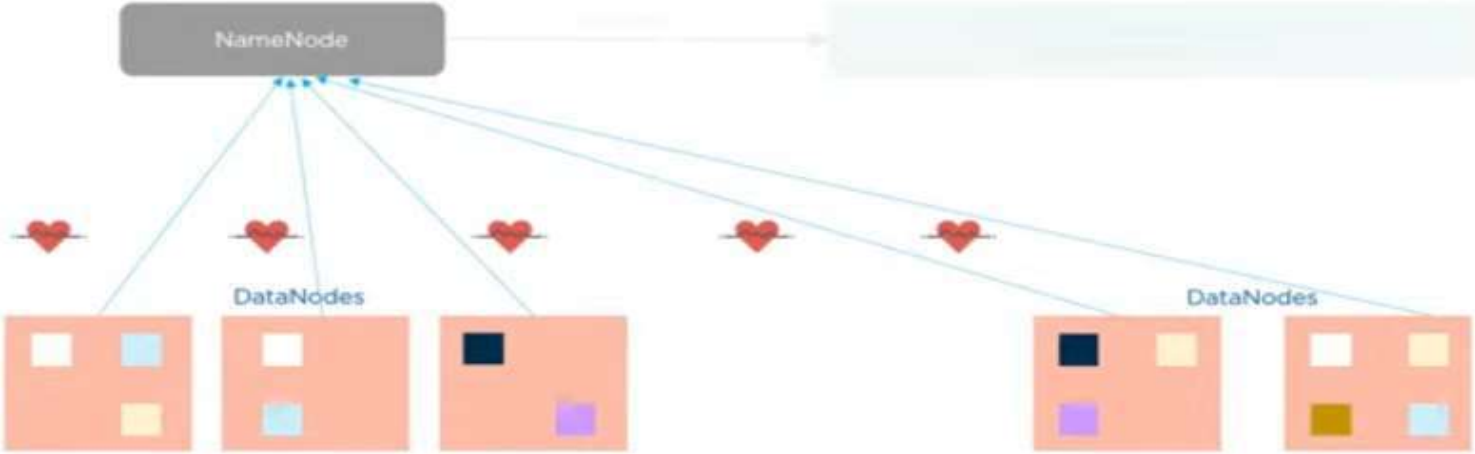
The name node is responsible for the workings of the data nodes. It also stores the metadata.

- The data nodes **read**, **write**, **process**, and **replicate** the data.
- They also send signals, known as **heartbeats**, to the name node
- These heartbeats show the **status of the data node**.

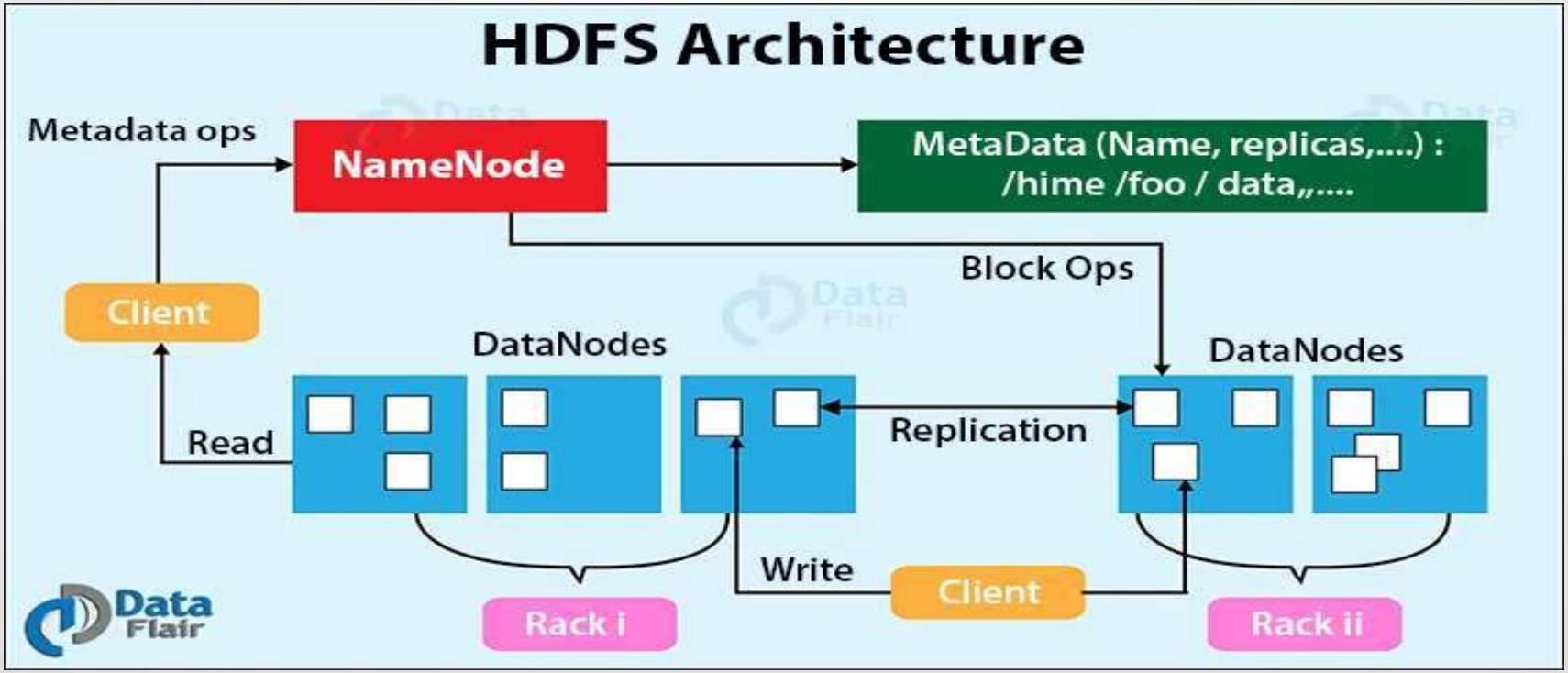


Consider that 30TB of data is loaded into the name node. The name node distributes it across the data nodes, and this data is replicated among the data notes. You can see in the image above that the blue, grey, and red data are replicated among the three data nodes.

HDFS Architecture

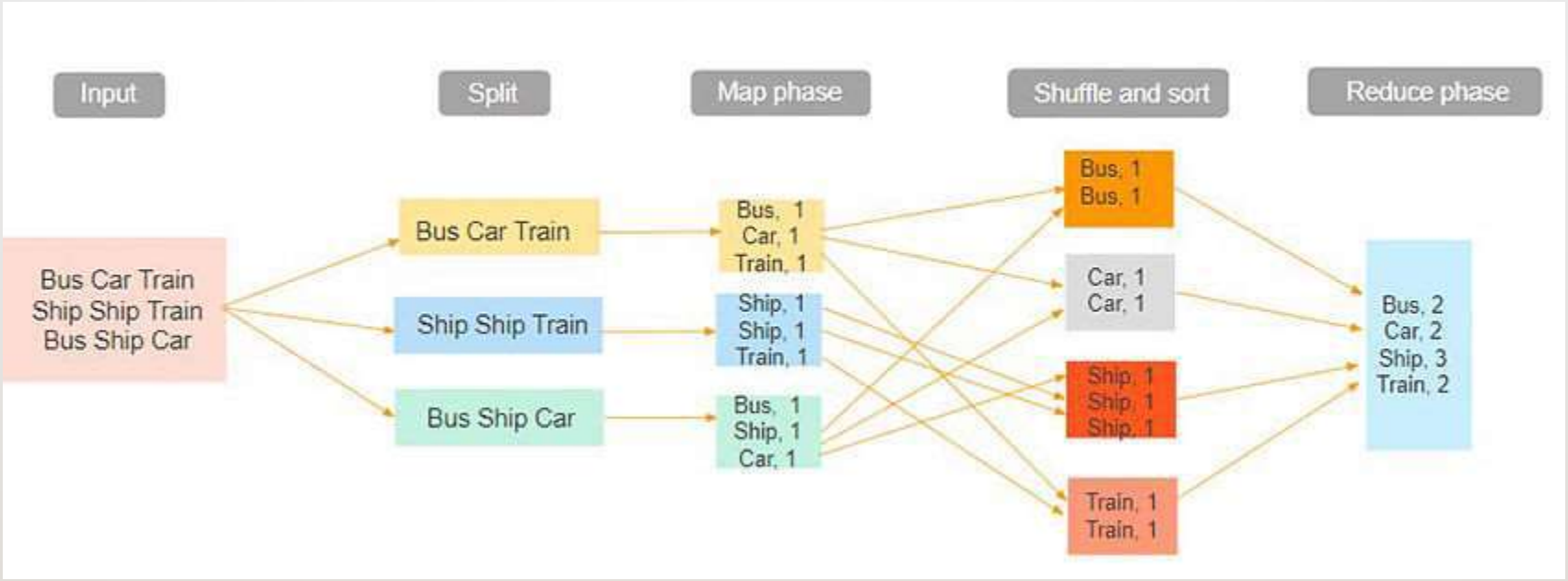


HeartBeat is the signal that DataNodes continuously send to the NameNode. This signal shows the status of the DataNode


HDFS Architecture
Metadata ops
NameNode
MetaData (Name, replicas,....) :
/hime /foo / data,,....
Block Ops
Client
DataNodes
DataNodes
Read
Replication
Write
Client
Rack i
Rack ii
Data Flair

Hadoop MapReduce

- Hadoop MapReduce is the **processing unit of Hadoop**.
- In the MapReduce approach, the **processing is done at the slave nodes**, and the final result is sent to the **master node**.
- A data containing code is used to process the entire data. This coded data is usually very small in comparison to the data itself. You only need to send a few kilobytes worth of code to perform a heavy-duty process on computers.


Input
Split
Map phase
Shuffle and sort
Reduce phase
Bus Car Train
Ship Ship Train
Bus Ship Car
Bus Car Train
Ship Ship Train
Bus Ship Car
Bus, 1
Car, 1
Train, 1
Ship, 1
Ship, 1
Train, 1
Bus, 1
Ship, 1
Car, 1
Bus, 1
Bus, 1
Car, 1
Car, 1
Ship, 1
Ship, 1
Ship, 1
Train, 1
Train, 1
Bus, 2
Car, 2
Ship, 3
Train, 2

- The input dataset is first split into **chunks of data**. In this example, the input has three lines of text with three separate entities - "bus car train," "ship ship train," "bus ship car." The dataset is then split into three chunks, based on these entities, and processed parallelly.
- In the **map phase**, the data is assigned a key and a value of 1. In this case, we have one bus, one car, one ship, and one train.
- These key-value pairs are then shuffled and sorted together based on their keys. At the **reduce phase,** the aggregation takes place, and the final output is obtained.

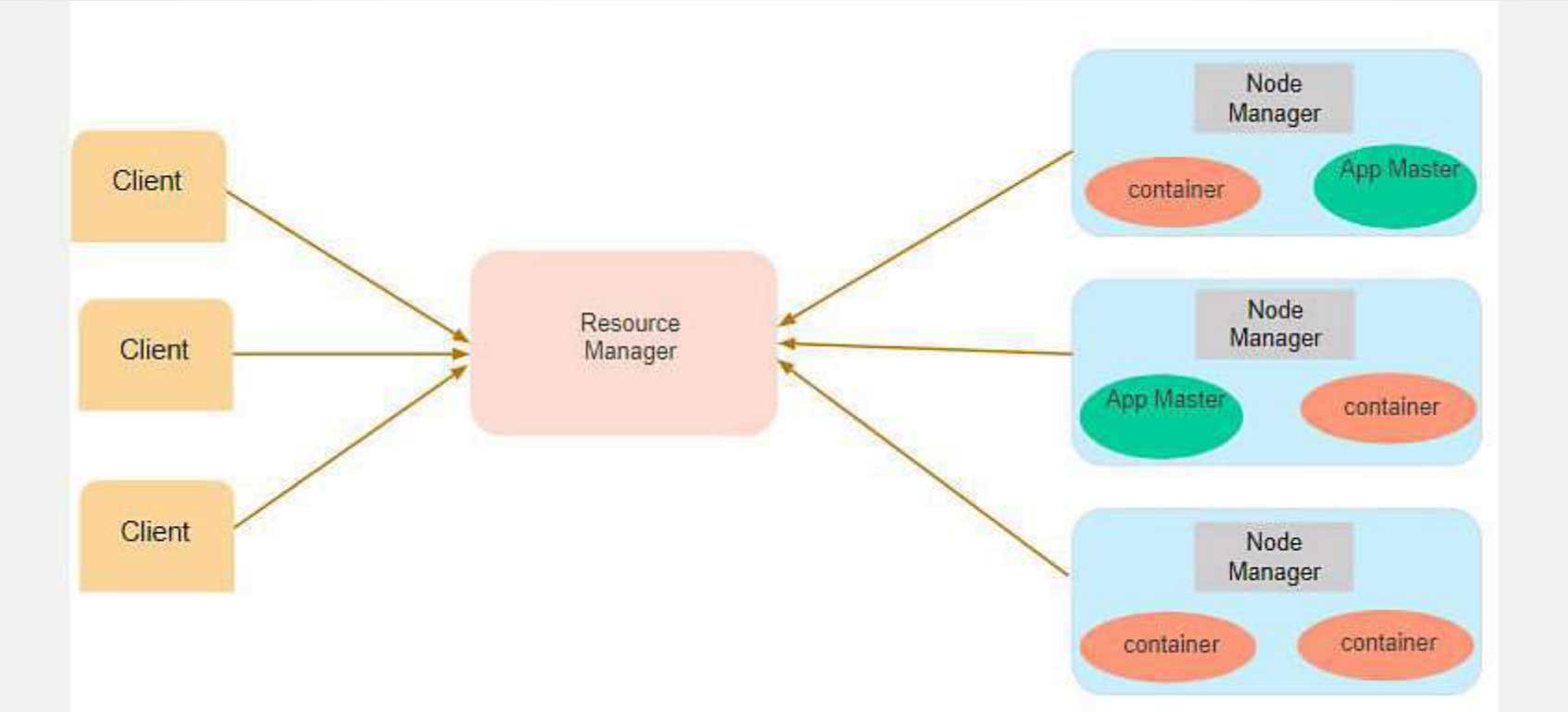
Hadoop YARN

- Hadoop **YARN** stands for Yet Another Resource Negotiator. It is the resource management unit of Hadoop and is available as a component of Hadoop version 2.

•Hadoop YARN acts like an OS to Hadoop. It is a file system that is built on top of HDFS.

•It is responsible for managing cluster resources to **make sure you don't overload one machine**.

•It performs **job scheduling** to make sure that the jobs are scheduled in the right place


Client
Client
Client
Resource Manager
Node Manager
container
App Master
Node Manager
App Master
container
Node Manager
container
container

- Suppose a **client machine** wants to do a query or fetch some code for data analysis. This job request goes to the **resource manager** (Hadoop Yarn), which is responsible for **resource allocation** and **management**.
- In the **node section**, each of the nodes has its node managers.
- These **node managers** manage the **nodes** and **monitor** the resource usage in the node.
- The **containers** contain a collection of physical resources, which could be **RAM**, **CPU**, or **hard drives.** Whenever a job request comes in, the app master requests the container from the node manager. Once the node manager gets the resource, it goes back to the Resource Manager.

How Does Hadoop Work?

The primary function of Hadoop is to process the data in an organised manner among the cluster of commodity software. The client should submit the data or program that needs to be processed. Hadoop HDFS stores the data. YARN, MapReduce divides the resources and assigns the tasks to the data. Let's know the working of Hadoop in detail.

- The client input data is divided into 128 MB blocks by HDFS. Blocks are replicated according to the replication factor: various DataNodes house the unions and their duplicates.
- The user can process the data once all blocks have been put on HDFS DataNodes.
- The client sends Hadoop the MapReduce programme to process the data.
- The user-submitted software was then scheduled by ResourceManager on particular cluster nodes.
- The output is written back to the HDFS once processing has been completed by all nodes.

5 Advantages of Hadoop for Big Data

Hadoop was created to deal with big data, so it's hardly surprising that it offers so many benefits. The five main benefits are:

- Speed. Hadoop's concurrent processing, MapReduce model, and HDFS lets users run complex queries in just a few seconds.
- Diversity. Hadoop's HDFS can store different data formats, like structured, semi-structured, and unstructured.
- Cost-Effective. Hadoop is an open-source data framework.
- Resilient. Data stored in a node is replicated in other cluster nodes, ensuring fault tolerance.
- Scalable. Since Hadoop functions in a distributed environment, you can easily add more servers.

To run Hadoop framework in windows follow the steps

1. Software Requirements

- **Hadoop Binary Distribution**: Download a stable version of Hadoop (e.g., Hadoop 3.x) from the Apache Hadoop website.
- **Java Development Kit (JDK)**: Hadoop requires JDK 8 or JDK 11, which should be installed and added to the system's `PATH`.
- **Windows Subsystem for Linux (WSL)** or **Cygwin**: If you want to simulate a Linux environment, WSL or Cygwin can help, though Hadoop can still run with native Windows commands.
- **WinRAR or 7-Zip**: Used to extract `.tar.gz` Hadoop binaries.
- **Optional**: Docker or a Virtual Machine with Ubuntu if you want a full Linux environment for Hadoop without compatibility issues.

Start Hadoop

- **Format the HDFS (Hadoop Distributed File System)**:

```bash
hdfs namenode -format
```

This command initializes HDFS before using it for the first time.

- **Start HDFS and YARN**:

```bash
start-dfs.cmd
start-yarn.cmd
```

These commands start Hadoop's distributed filesystem and resource manager (YARN) services.

Basic HDFS Commands

- **List Files in HDFS:**

  bash

  Copy code

  ```
  hdfs dfs -ls /
  ```

  This command lists files in the root directory of HDFS.

- **Make a Directory:**

  bash

  Copy code

  ```
  hdfs dfs -mkdir /user/students
  ```

  Creates a new directory named "students" under `/user`.

- **Upload Files to HDFS**:

```bash
hdfs dfs -put localfile.txt /user/students/
```

Copies `localfile.txt` from your Windows machine into the HDFS directory `/user/students/`.

- **Read a File in HDFS**:

```bash
hdfs dfs -cat /user/students/localfile.txt
```

Displays the contents of the file directly from HDFS.

- **Remove Files in HDFS:**

```bash
hdfs dfs -rm /user/students/localfile.txt
```

Deletes `localfile.txt` from HDFS.

Intermediate HDFS Commands

- **Check HDFS Disk Usage**:

```bash
hdfs dfs -du -s /user/students
```

Displays the disk space used by files under the specified directory.

- **Copy from HDFS to Local File System**:

```bash
hdfs dfs -get /user/students/localfile.txt C:\localpath\
```

Copies a file from HDFS back to the Windows file system.

MapReduce Commands

- Run a WordCount Example:

```
yarn jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar wordcount /input /output
```

This example job takes an HDFS `/input` directory and processes it, saving the output to `/output`.

- **Check MapReduce Job Status:**

```bash
yarn application -list
```

Lists all active YARN applications, including MapReduce jobs.

- **Create Nested Directories:**

```bash
hdfs dfs -mkdir -p /user/students/classA/assignments
```

Creates a nested directory structure in HDFS.

- **Copy Files Between HDFS Directories:**

```bash
hdfs dfs -cp /user/students/classA/file1.txt /user/students/classB/
```

Copies `file1.txt` from one HDFS directory to another.

- **Move Files in HDFS**:

```bash
hdfs dfs -mv /user/students/classA/file1.txt /user/students/classB/
```

Moves `file1.txt` from one directory to another.

- **Count Files, Directories, and Bytes**:

```bash
hdfs dfs -count /user/students
```

Displays the count of files, directories, and bytes in the specified directory.

- **HDFS File Checksum**:

```bash
hdfs dfs -checksum /user/students/file1.txt
```

Shows the checksum of a file, which can be useful for verifying file integrity.

- **View File Permissions**:

```bash
hdfs dfs -ls -R /user/students
```

Recursively lists files and directories with permissions.

- **Change File Permissions**:

```bash
hdfs dfs -chmod 755 /user/students/file1.txt
```

Modifies the permissions of the file to `755` (owner read, write, execute; group and others read, execute).

- **Change File Ownership**:

```bash
hdfs dfs -chown new_owner /user/students/file1.txt
```

Changes the file owner to `new_owner`.

- **Check HDFS Health:**

```bash
hdfs fsck / -files -blocks -racks
```

Checks the health of the HDFS filesystem, listing files, blocks, and rack awareness.

YARN (Yet Another Resource Negotiator) Commands

View Running Applications in YARN:

```bash
yarn application -list
```

Shows all currently running applications on YARN.

Kill a Running Application:

```bash
yarn application -kill application_id
```

Stops a specific YARN application using its application ID.

- **View YARN Node Status**:

```bash
yarn node -list
```

Lists all nodes managed by YARN, along with their status.

- **Check Application Logs**:

```bash
yarn logs -applicationId application_id
```

Retrieves logs for a specific YARN application.

Administrative Hadoop Commands

View HDFS Cluster Summary:

```bash
hdfs dfsadmin -report
```

Provides an overview of the HDFS cluster, including disk capacity and utilization.

Apache Pig and Apache Hive

- **Apache Pig** and **Apache Hive** provide high-level frameworks for processing and analyzing large datasets stored in HDFS.
- Each has distinct strengths and use cases, designed to simplify working with big data by abstracting away from **Java-based MapReduce programming**.

1. Apache Pig

- **Purpose**: Apache Pig is a high-level data flow scripting language that provides a simple way to process and analyze large data sets.
- **Language**: Uses **Pig Latin**, a procedural language designed to simplify the creation of data processing workflows.
- **Advantages**:
  - **Flexible Data Model**: Pig can handle structured, semi-structured, and unstructured data, making it suitable for diverse datasets.
  - **Simplified Processing**: Pig Latin scripts break down data flows into multiple steps, which Hadoop translates into a series of MapReduce jobs automatically.
- **Common Use Cases**:
  - Data transformations (filtering, grouping, joining, sorting)
  - Extract, Transform, Load (ETL) processes
  - Iterative data processing tasks

- **Basic Pig Commands:**
  - **Load Data:**

```pig
data = LOAD 'data.txt' USING PigStorage(',');
```

  - **Filter Data:**

```pig
filtered_data = FILTER data BY $0 == 'specific_value';
```

  - **Group Data:**

```pig
grouped_data = GROUP data BY $0;
```

2. Apache Hive

- **Purpose**: Apache Hive is a data warehousing and SQL-like framework that enables querying and managing large datasets.
- **Language**: Uses **HiveQL (HQL)**, an SQL-like query language optimized for batch processing in Hadoop.
- **Advantages**:
  - **SQL Compatibility**: Familiar to users with SQL experience, allowing them to perform queries without needing to write MapReduce code.
  - **Schema on Read**: Hive applies schema when querying data, making it versatile and suitable for both structured and semi-structured data.
  - **Integration with BI Tools**: Hive's structure is compatible with Business Intelligence (BI) tools for data analytics and reporting.
- **Common Use Cases**:
  - Data summarization and analysis
  - Data mining tasks
  - Batch data processing for reporting

- Basic Hive Commands:
  - Create Database and Table:

```
CREATE DATABASE student_db;
USE student_db;
CREATE TABLE students (id INT, name STRING, age INT) ROW FORMAT
DELIMITED FIELDS TERMINATED BY ',';
```

- **Load Data into Table:**

```sql
LOAD DATA INPATH '/path/to/data.csv' INTO TABLE students;
```

- **Query Data:**

```sql
SELECT name, age FROM students WHERE age > 18;
```

Comparison of Pig and Hive

Feature	Apache Pig	Apache Hive
Language	Pig Latin (procedural)	HiveQL (declarative SQL-like)
Ideal User	Data engineers familiar with scripting	Analysts familiar with SQL
Data Type Compatibility	Structured, semi-structured, unstructured	Mostly structured and semi-structured
Execution	MapReduce jobs	MapReduce, Spark, or Tez
Primary Use Cases	Data transformation, ETL, batch processing	Data warehousing, batch analytics, reporting