



BITS Pilani
Pilani Campus

BITS Pilani presentation

Dr. Vivek V. Jog
Dept. Of Computer Engineering





BITS Pilani
Pilani Campus



Big Data Systems (S1-24_CCZG522)

Lecture No.3

INDEX

- Big Data Growth Drivers
- What is Big Data?
- Hadoop Introduction
- Hadoop Master/Slave Architecture
- Hadoop Core Components
- HDFS Data Blocks
- HDFS Read/Write Mechanism
- What is MapReduce
- MapReduce Program
- MapReduce Job Workflow
- Hadoop Ecosystem



Five V's

"Big data is the term for a collection of data sets so **large and complex** that it becomes difficult to process using on-hand database management tools or traditional data processing applications"

Volume



Processing increasing huge data sets

Variety



Processing different types of data

Velocity



Data is being generated at an alarming rate

Value



Finding correct meaning out of the data

Veracity

2010	2011	2012	2013
42	?	548	612
14	44	100	140
—	28	120	140
9.1	22	?	119

Uncertainty and inconsistencies in the data

Traditional System V/s Big Data

Traditional Scenario:

2 orders per hour

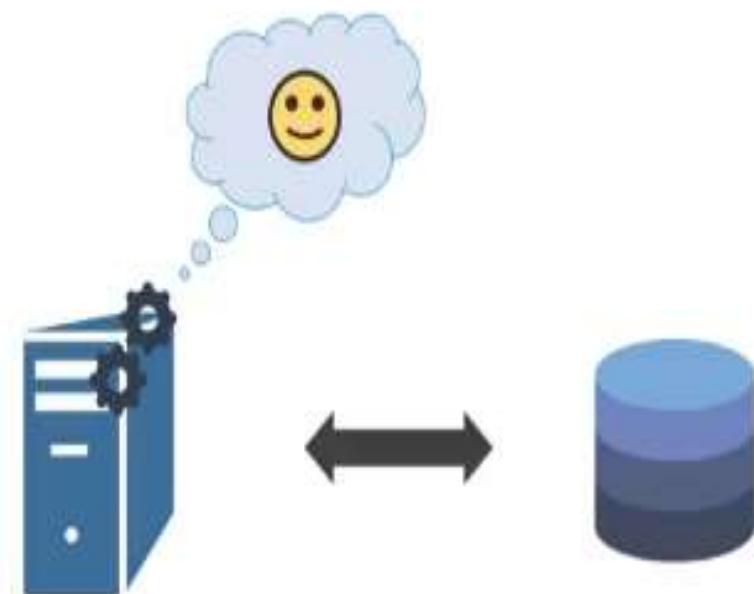


Single Cook

Food Shelf

Traditional Scenario:

Data is generated at a steady rate and is structured in nature



Traditional Processing System

RDBMS

Contd..

Scenario 2:

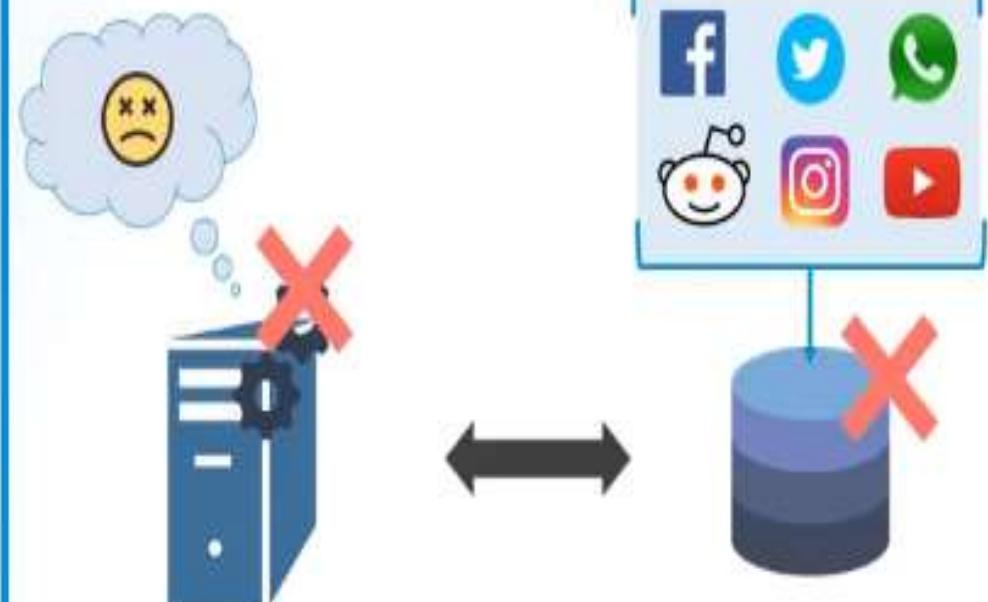
- They started taking Online orders
- 10 orders per hour



Single Cook
(Regular Computing System)

Big Data Scenario:

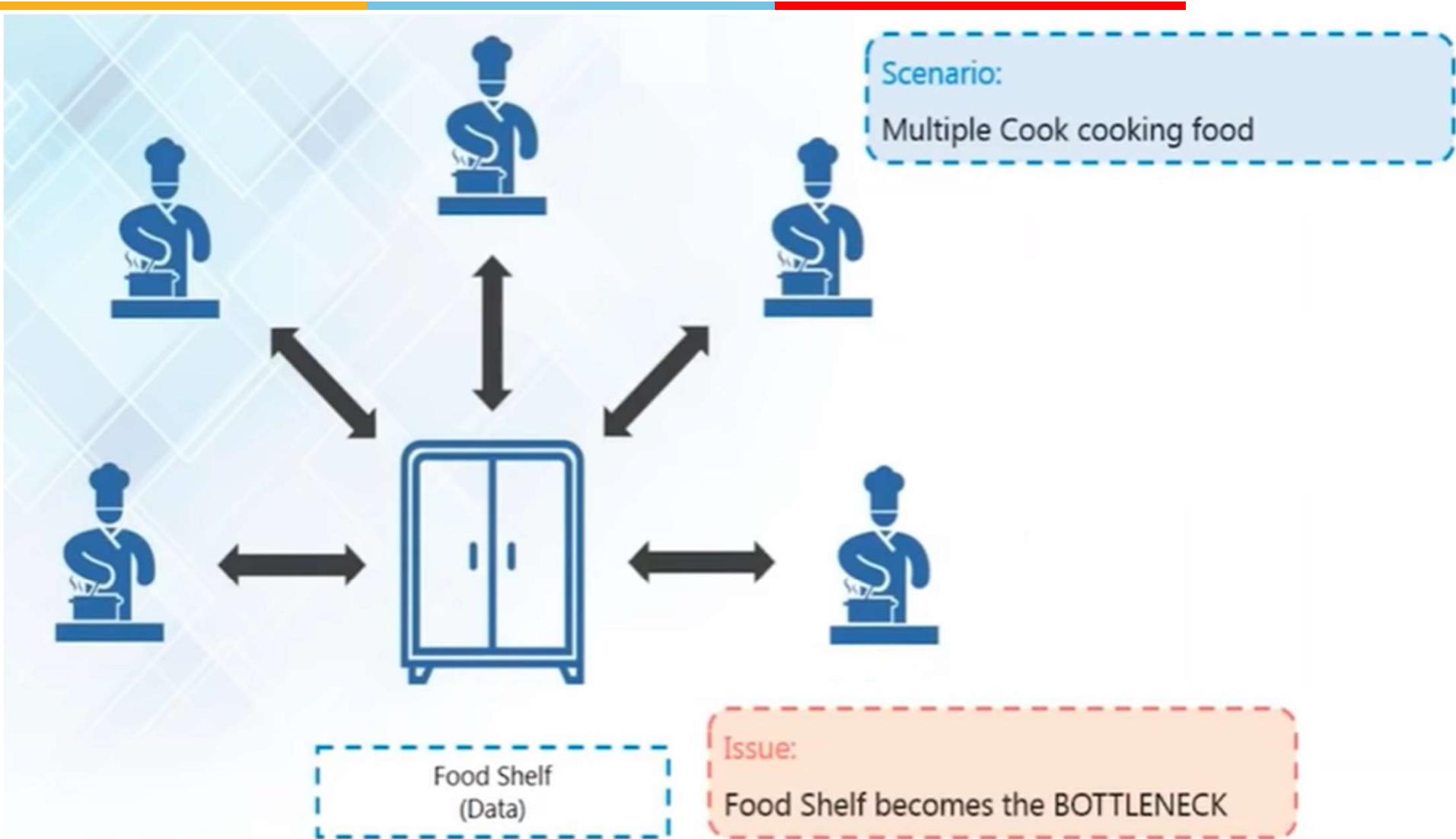
Heterogenous data is being generated at an alarming rate by multiple sources



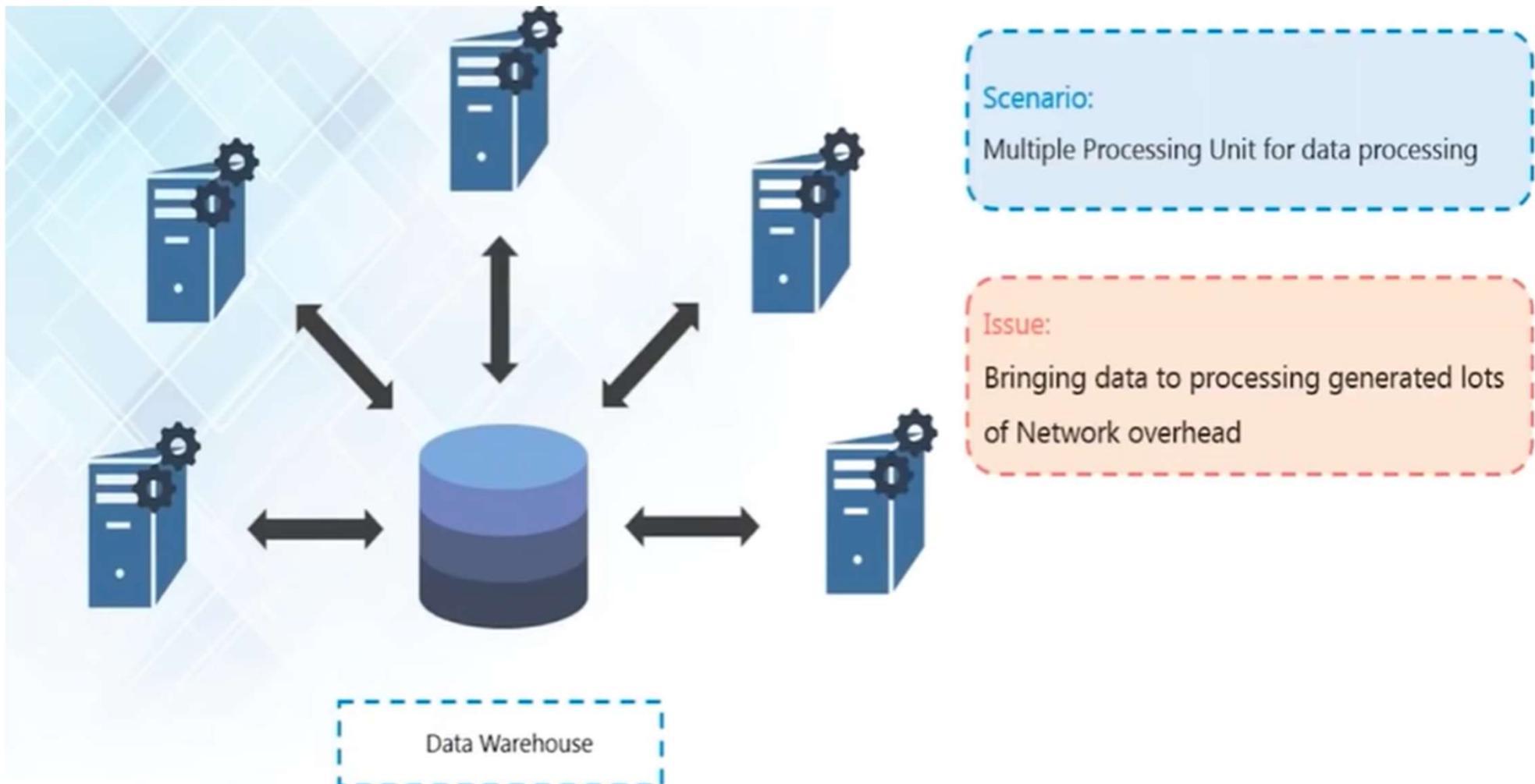
Traditional Processing System

RDBMS

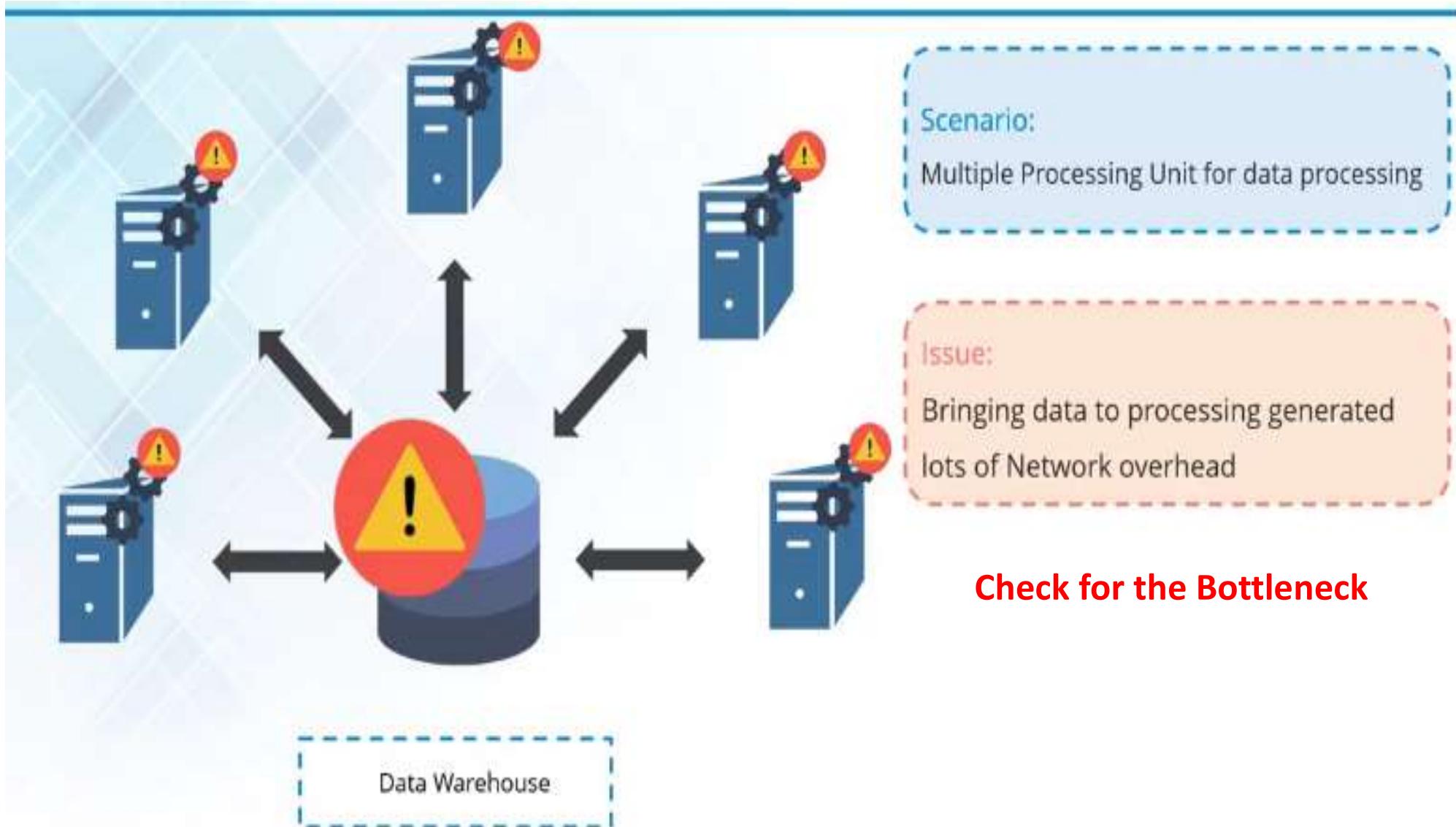
Bottleneck



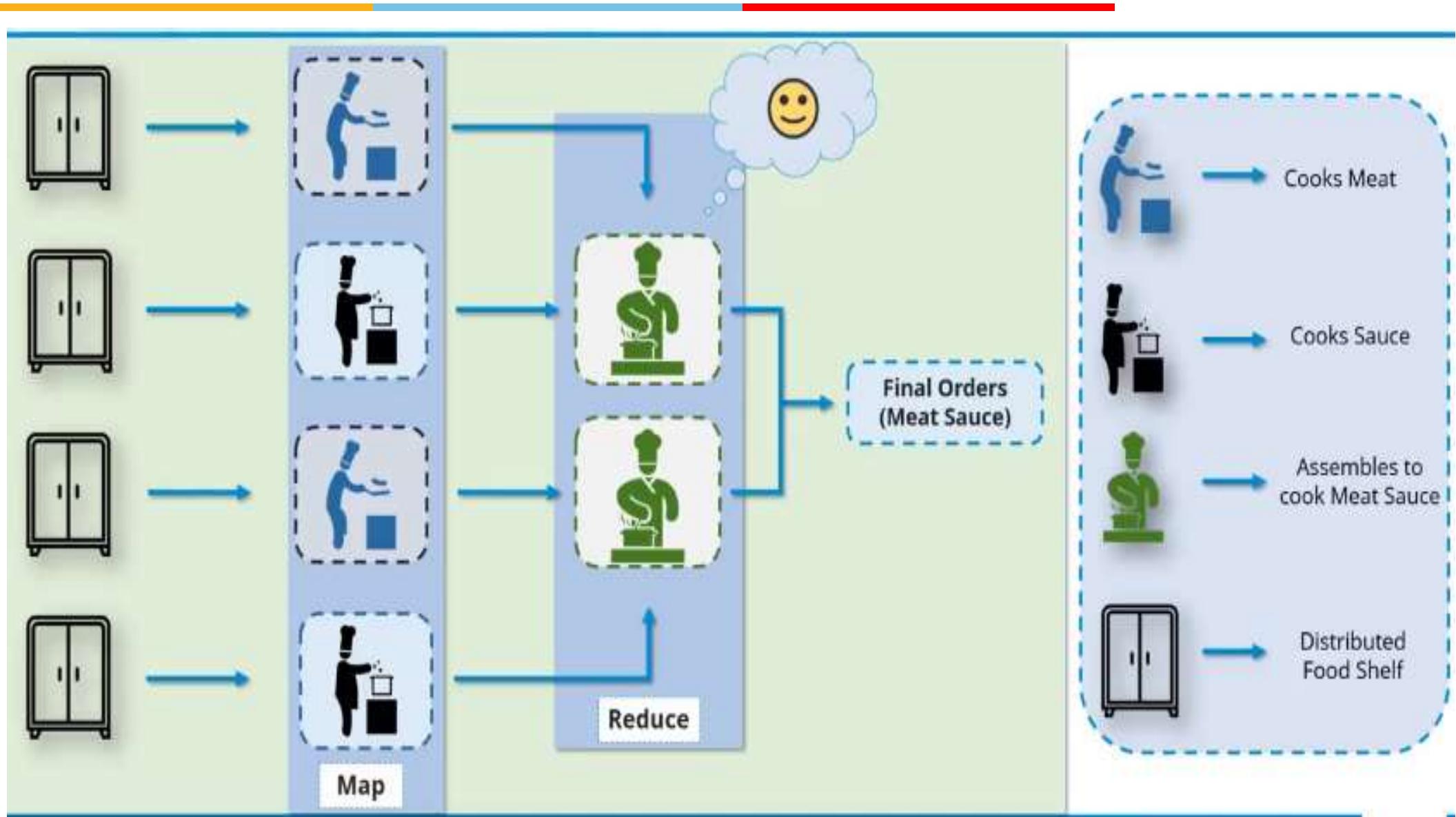
Bottleneck



More number of orders = More number of cooks



Effective Solution



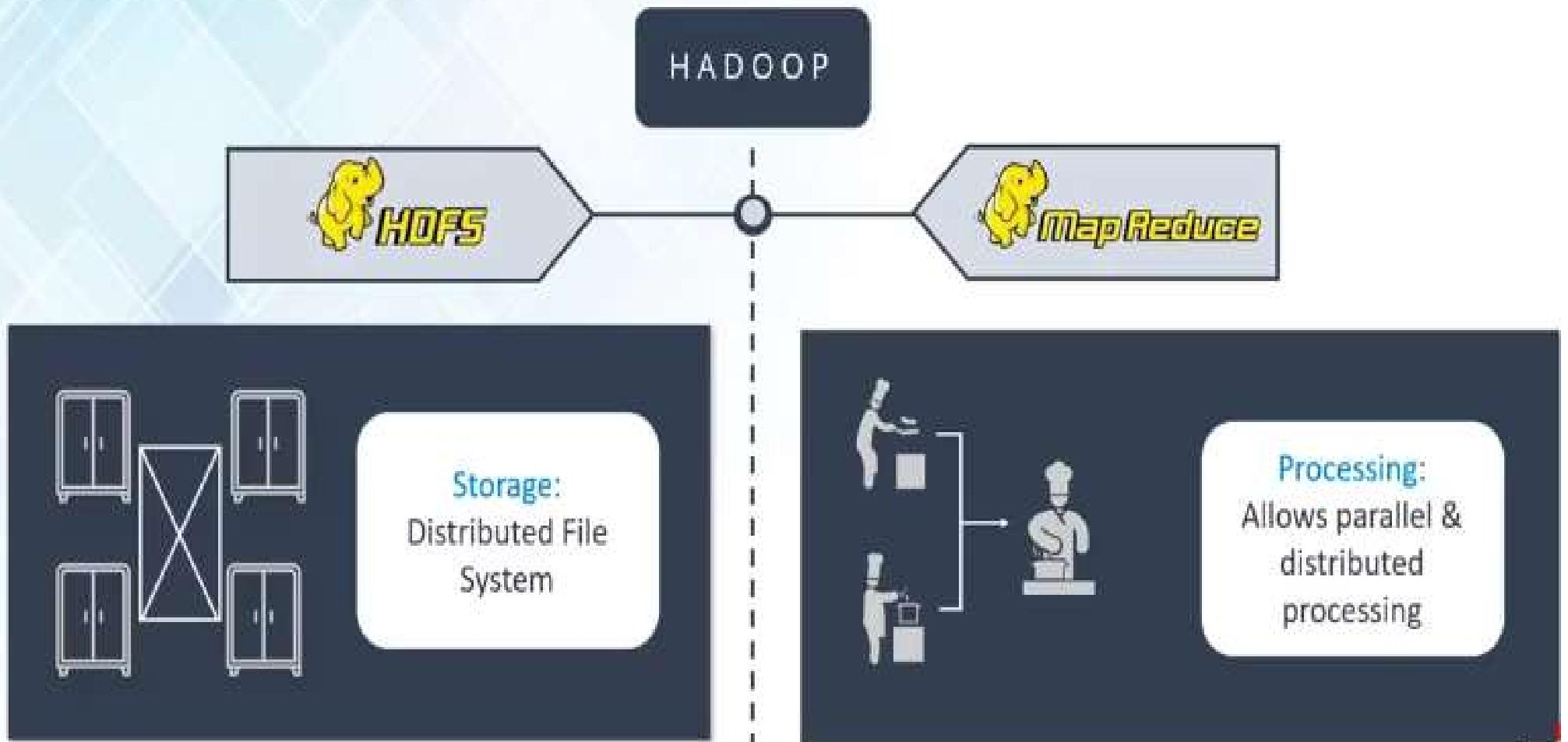


Solution Framework

WE NEED SOME FRAMEWORK THAT CAN PROVIDES
SOLUTION

Apache Hadoop

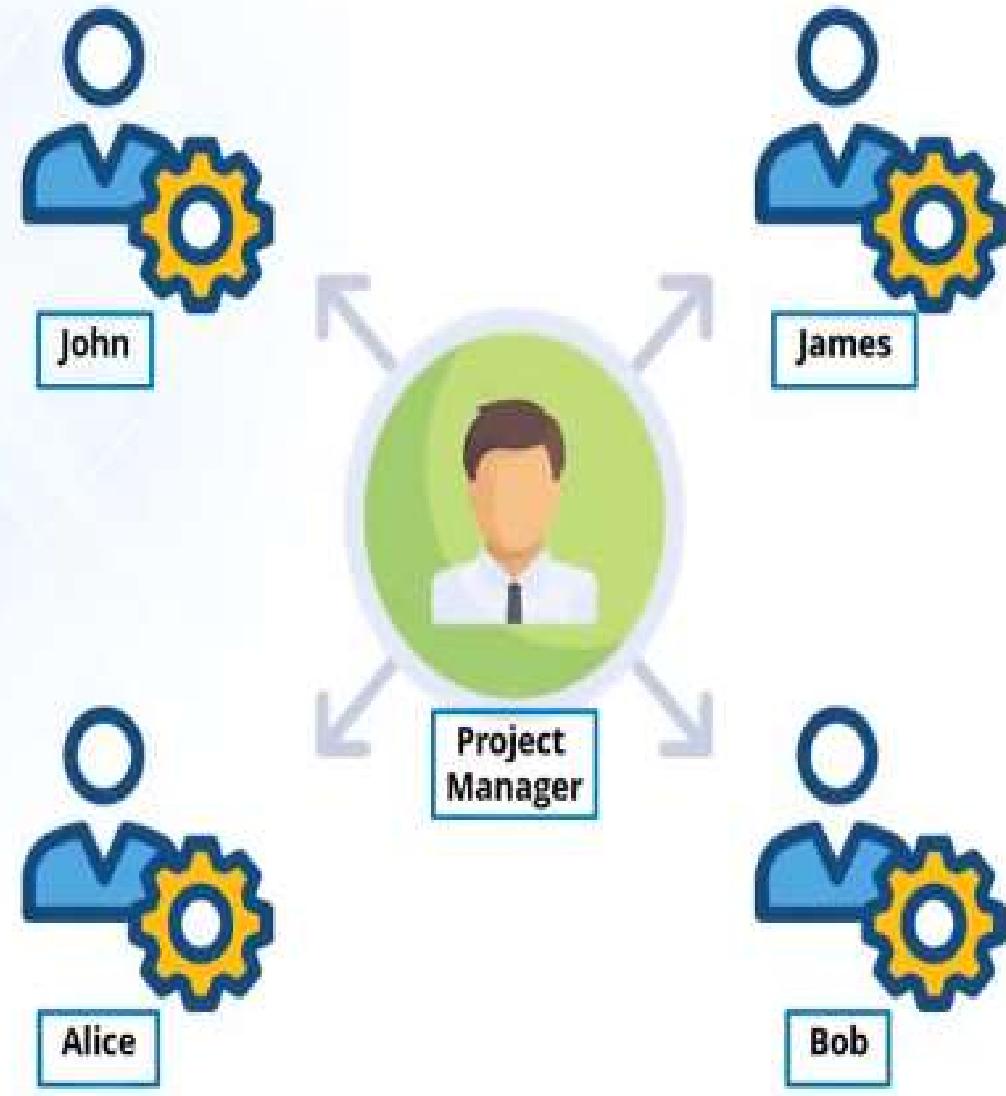
Hadoop is a framework that allows us to **store** and **process** large data sets in **parallel** and **distributed** fashion



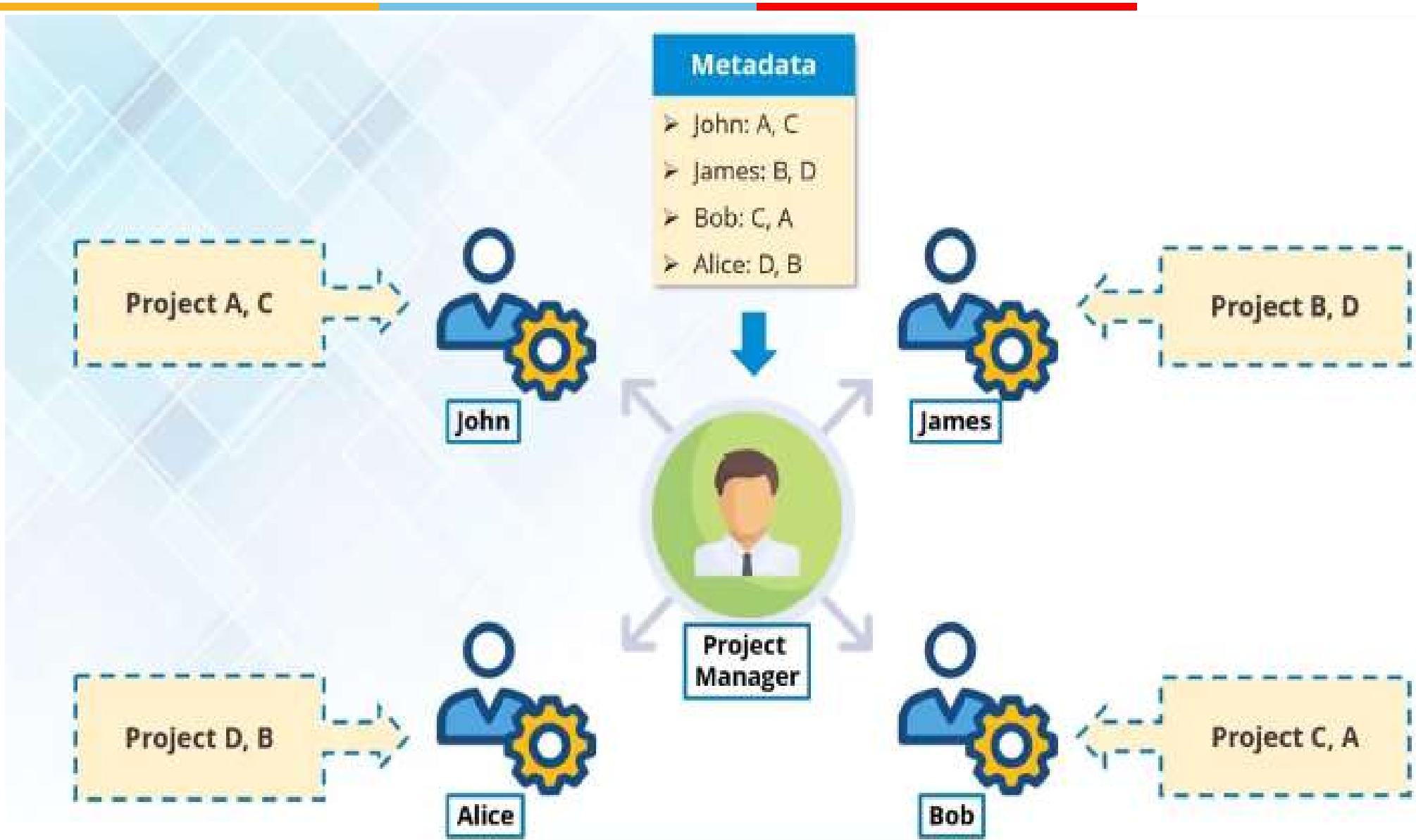
Master/Slave Approach

Scenario:

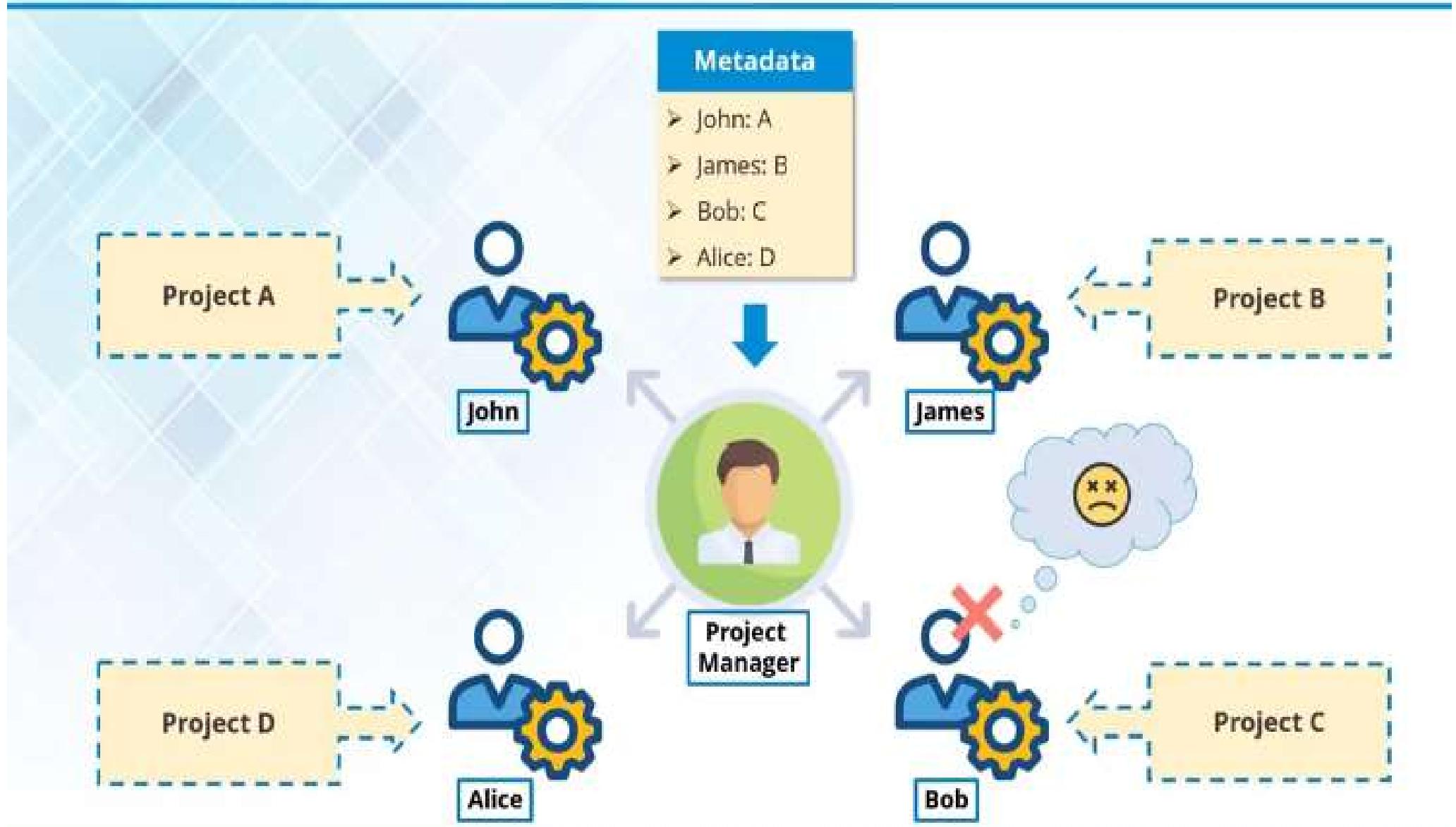
A project Manager managing a team of four employees. He assigns project to each of them and tracks the progress



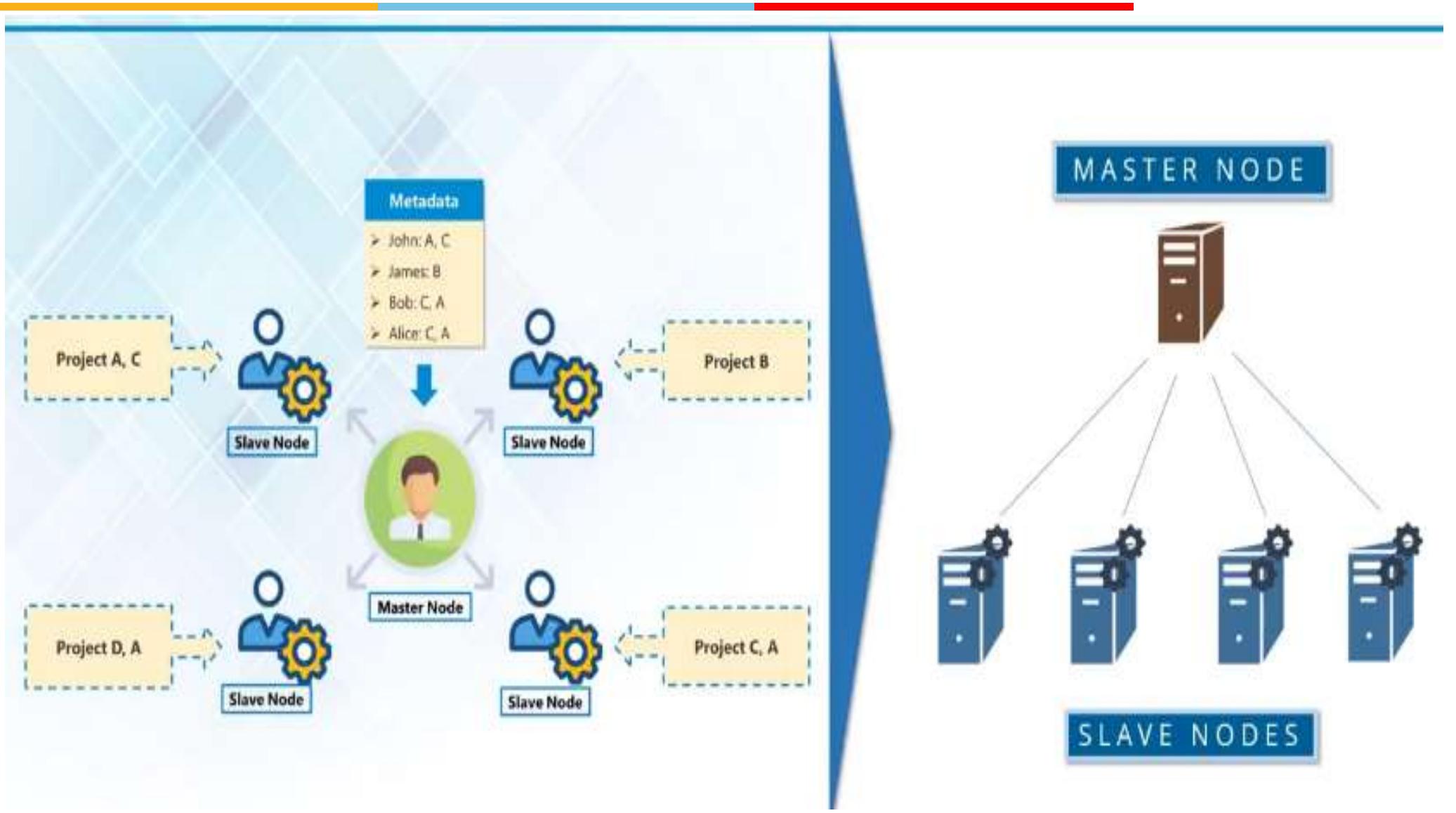
Maintain Metadata



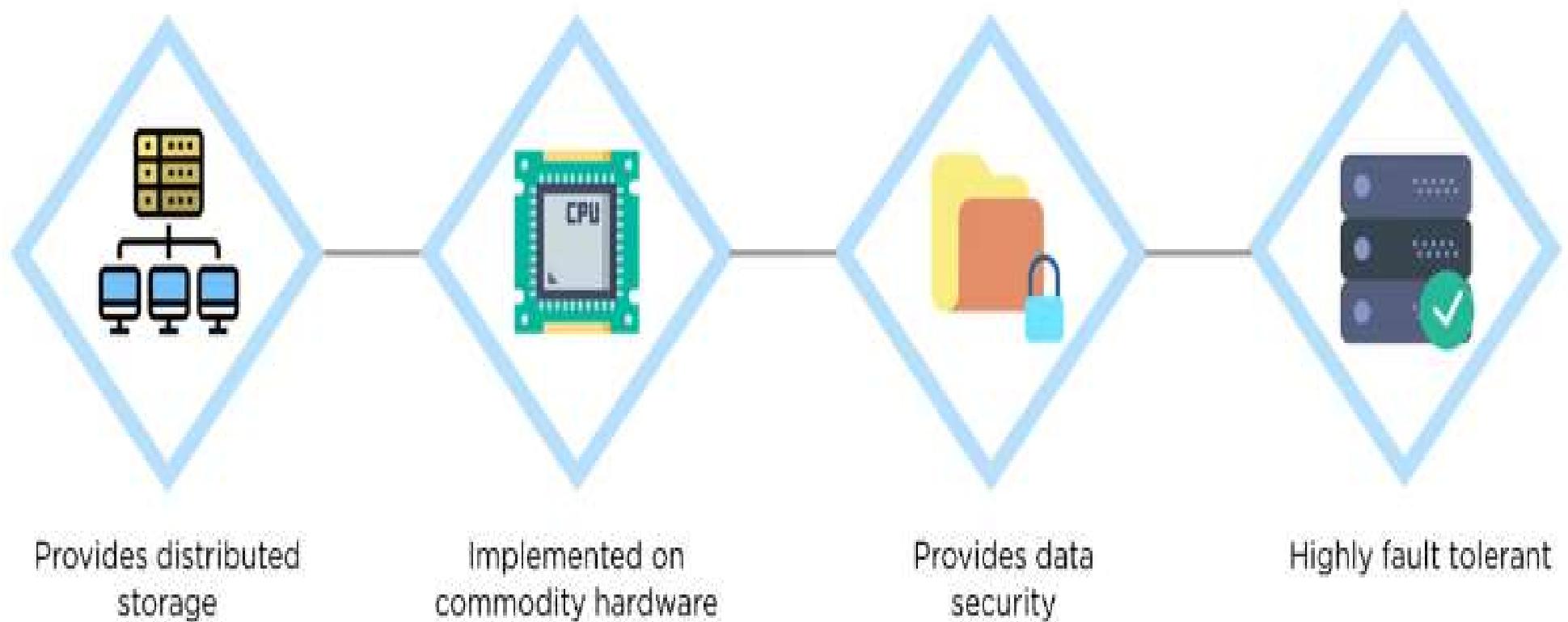
Backup Failover Solution



Evolution of Architecture



Evolution of HDFS



3 Core Components

01

NameNode

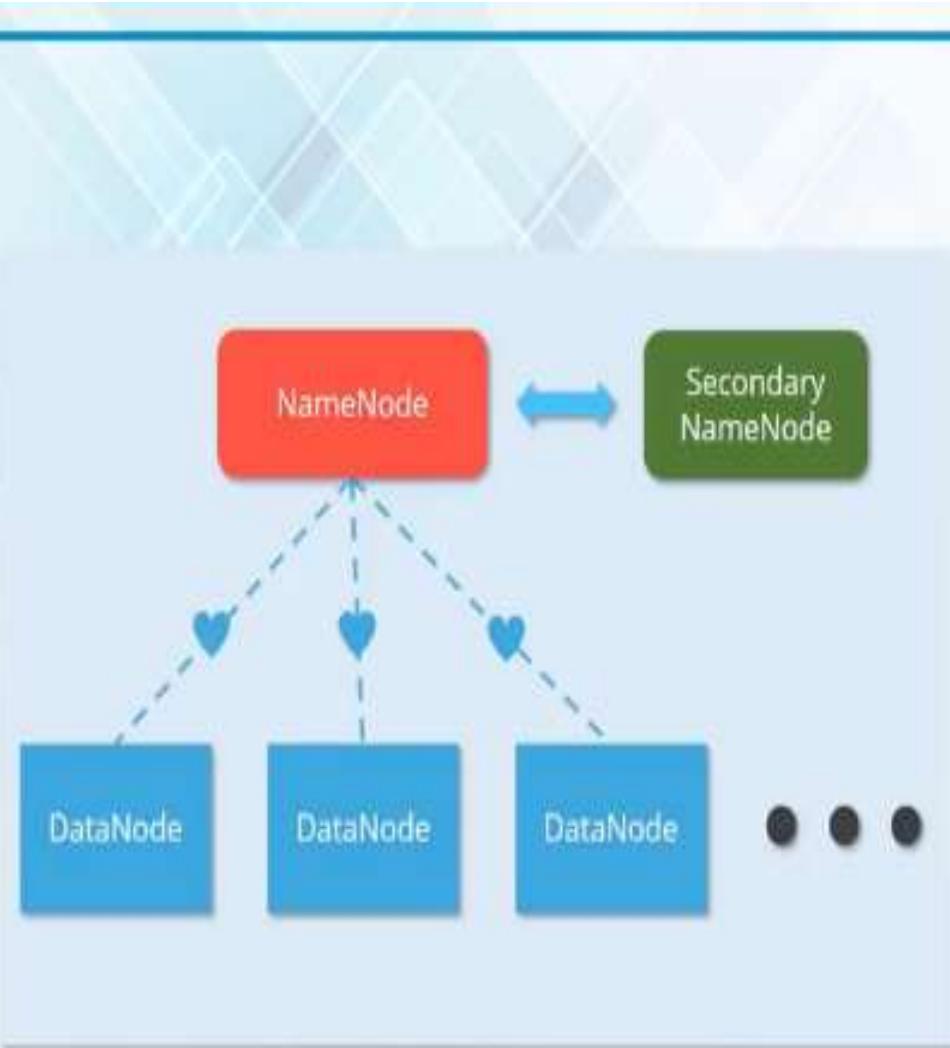
02

DataNode

03

Secondary
NameNode

Namenode & Datanode



NameNode:

- Maintains and Manages DataNodes
- Records metadata i.e. information about data blocks e.g. location of blocks stored, the size of the files, permissions, hierarchy, etc.
- Receives heartbeat and block report from all the DataNodes

DataNode:

- Slave daemons
- Stores actual data
- Serves read and write requests from the clients

Daemon Services

```
Terminal 11:56 AM
shriramkv@shriramkv:~/Desktop/OS Concepts
DS_2.mkv
shriramkv@shriramkv:~/Desktop/OS Concepts$ clear

shriramkv@shriramkv:~/Desktop/OS Concepts$ gedit Daemon.
shriramkv@shriramkv:~/Desktop/OS Concepts$ gedit Daemon.
Daemon.c      Daemon.c~      Daemon.odp
shriramkv@shriramkv:~/Desktop/OS Concepts$ gedit Daemon.c

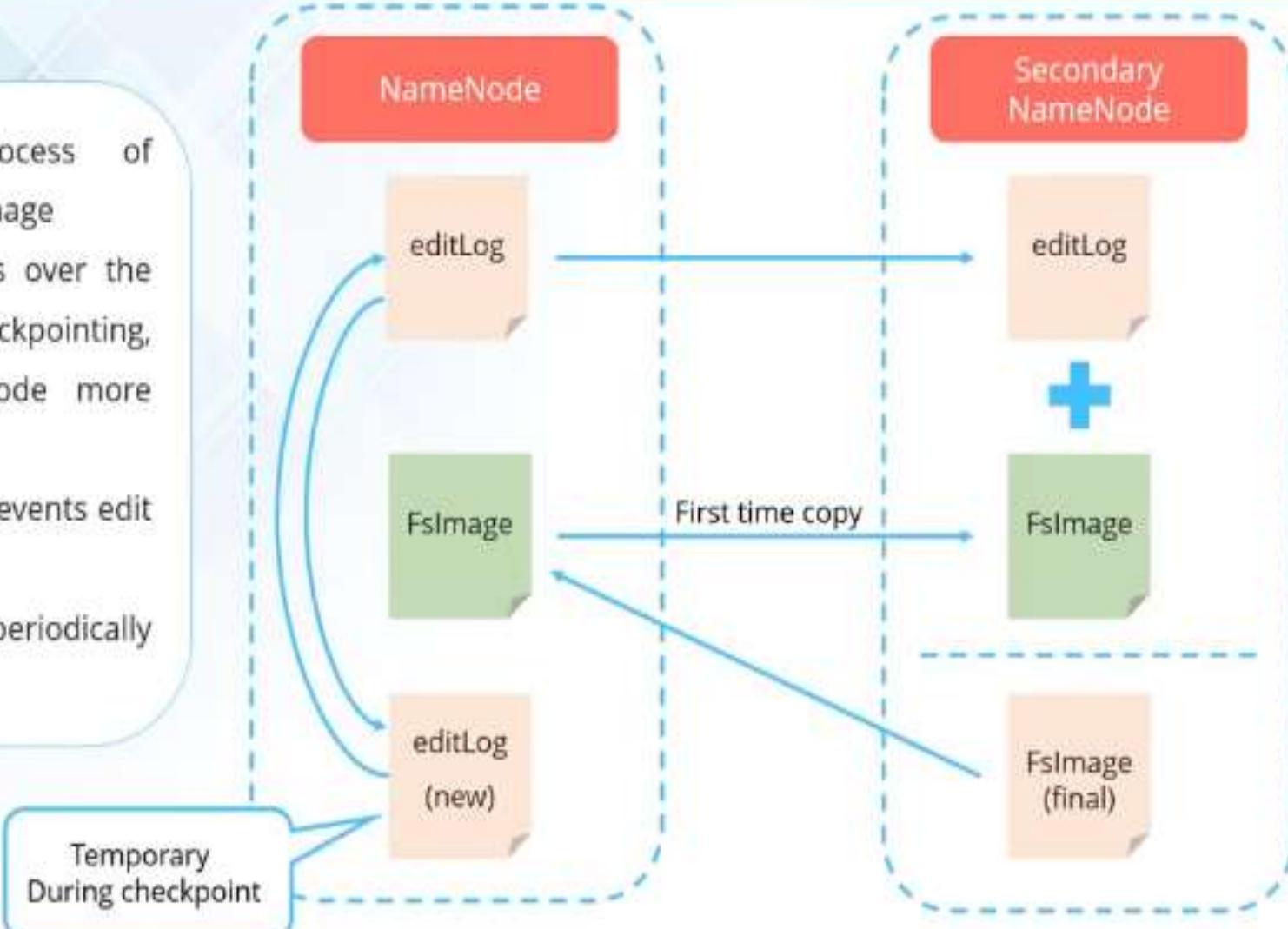
^C
shriramkv@shriramkv:~/Desktop/OS Concepts$ gedit Daemon.c &
[1] 2419
shriramkv@shriramkv:~/Desktop/OS Concepts$ ps -axl
F  UID  PID  PPID PRI  NI    VSZ   RSS WCHAN STAT TTY          TIME COMMAND
4  0     1     0  20   0    4544  2548 poll_s Ss  ?        0:01 /sbin/init
1  0     2     0  20   0      0  kthrea S  ?        0:00 [kthreadd]
1  0     3     2  20   0      0  smpboo S  ?        0:00 [ksoftirqd/0]
1  0     4     2  20   0      0  worker  S  ?        0:00 [kworker/0:0]
1  0     5     2   0 -20      0  worker  S< ?        0:00 [kworker/0:0]
1  0     6     2  20   0      0  worker  S  ?        0:00 [kworker/u16]
1  0     7     2  20   0      0  rcu_gp S  ?        0:00 [rcu_sched]
```

About Daemon Process

- It is a program.
- Runs in Unix/Linux without interruption or user initiation to happen.
- Executed in the background.
- Recollect, orphans.
- No terminal usage
- Command to see the daemons
- Ps -axl (results will have ? Under TTY, it tells Daemon's existence)

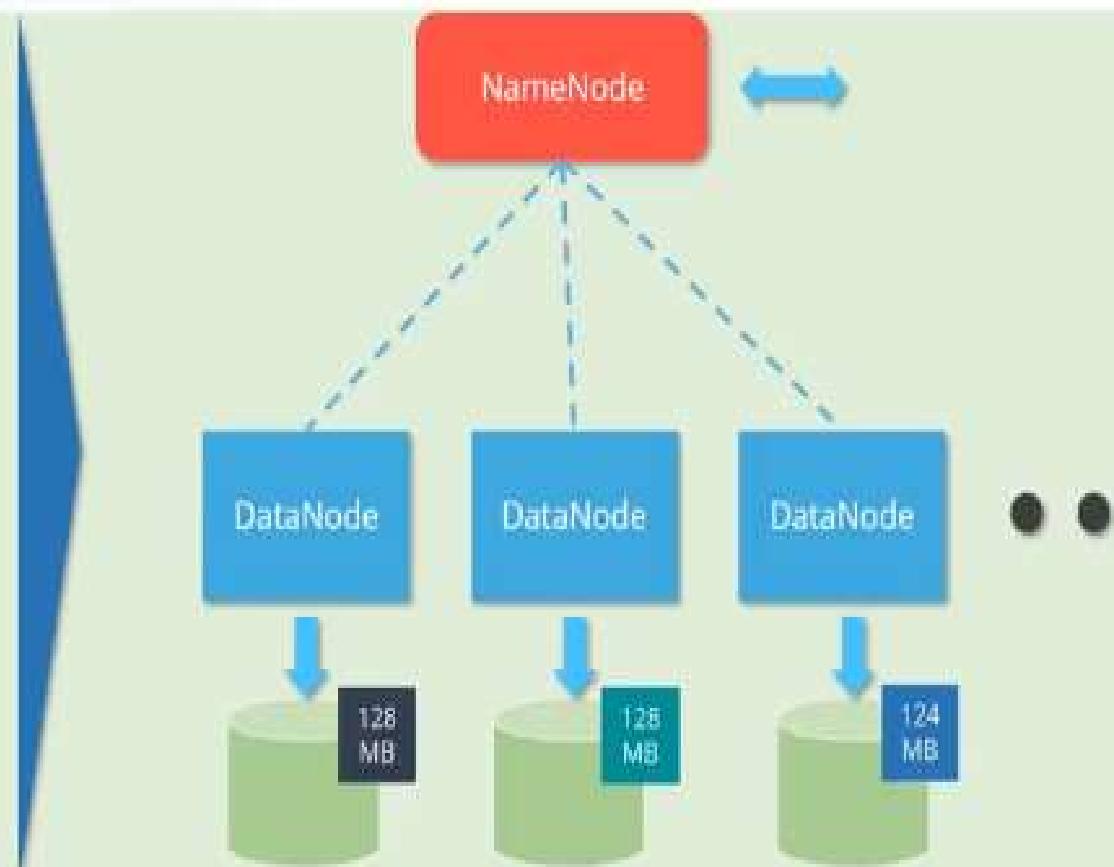
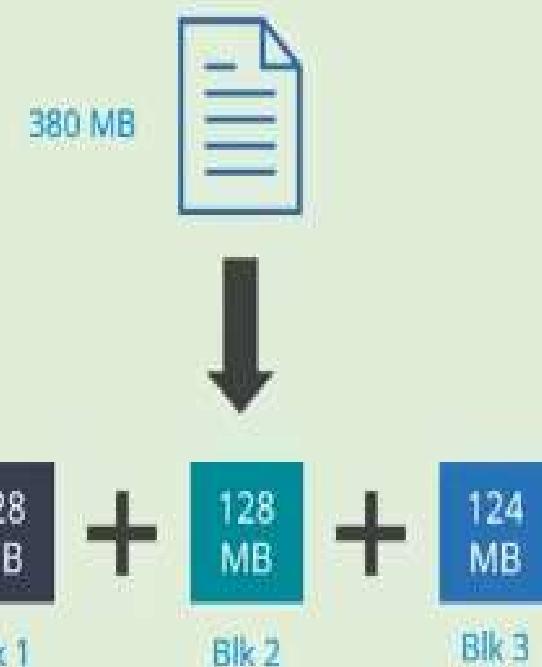
Secondary NameNode & Checkpoint

- Checkpointing is a process of combining edit logs with FsImage
- Secondary NameNode takes over the responsibility of checkpointing, therefore, making NameNode more available
- Allows faster Failover as it prevents edit logs from getting too huge
- Checkpointing happens periodically (default: 1 hour)

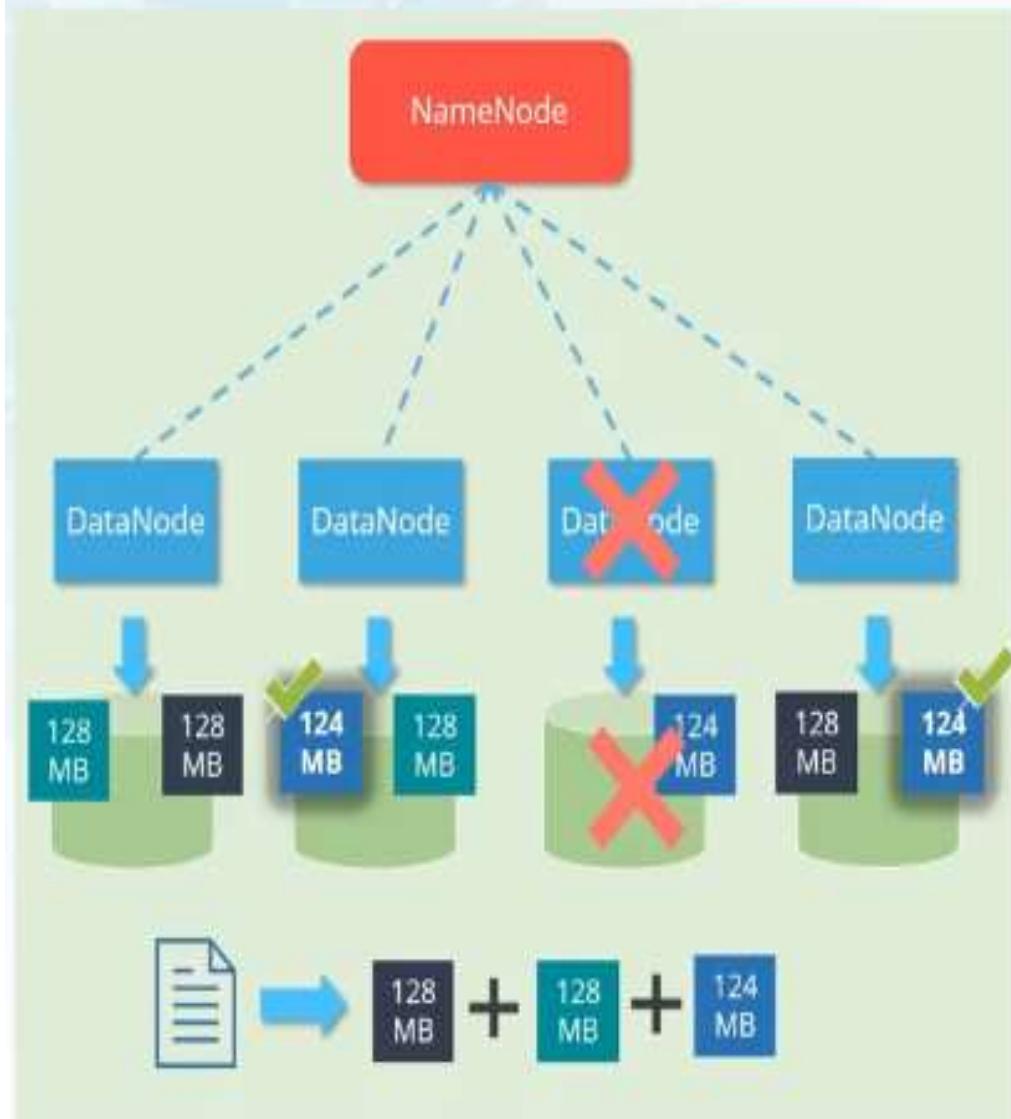


HDFS DataNode Blocks

- Each file is stored on HDFS as blocks
- The default size of each block is 128 MB in Apache Hadoop 2.x (64 MB in Apache Hadoop 1.x)



Replication

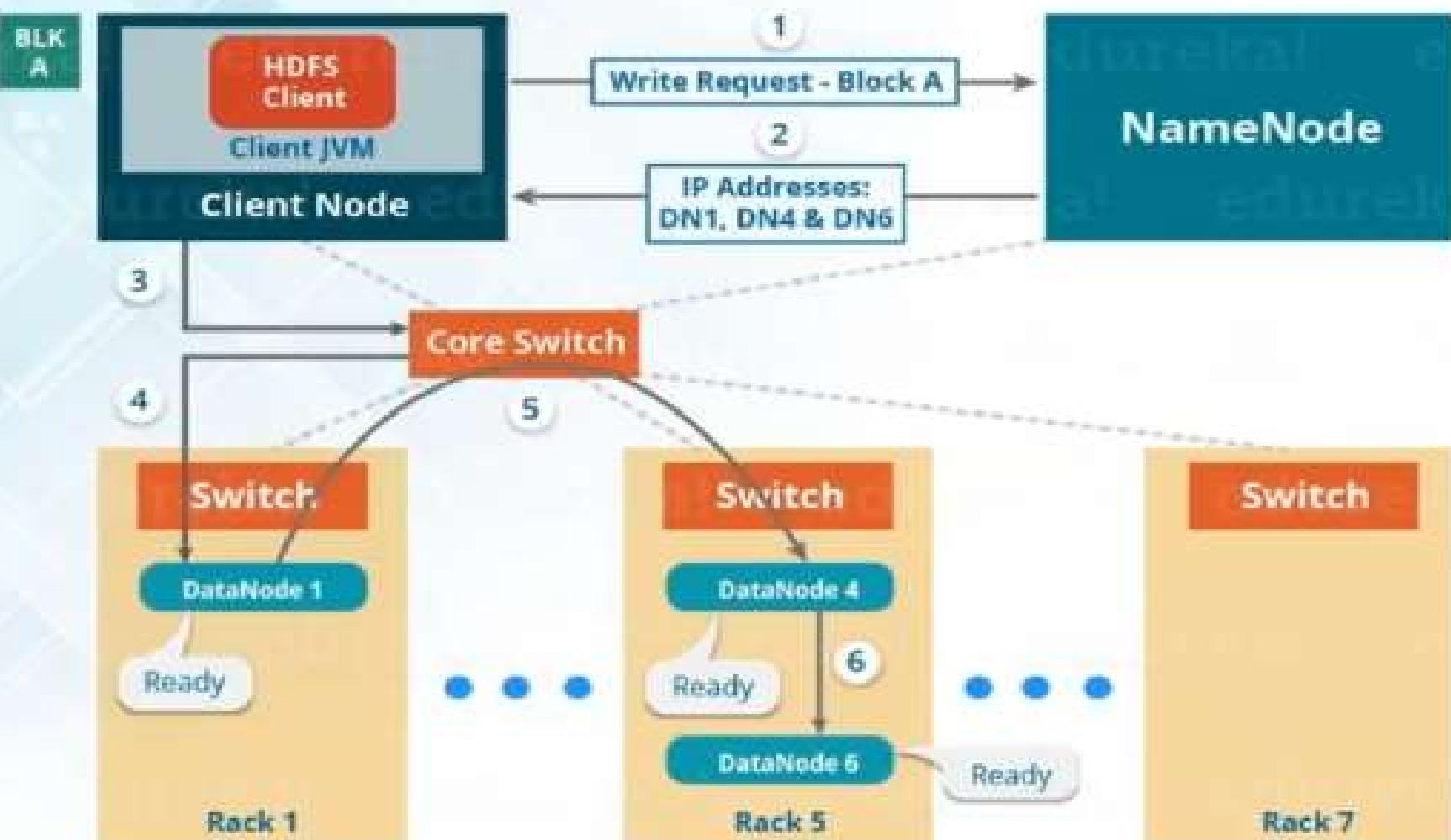


Solution:

Each data blocks are replicated (thrice by default) and are distributed across different DataNodes

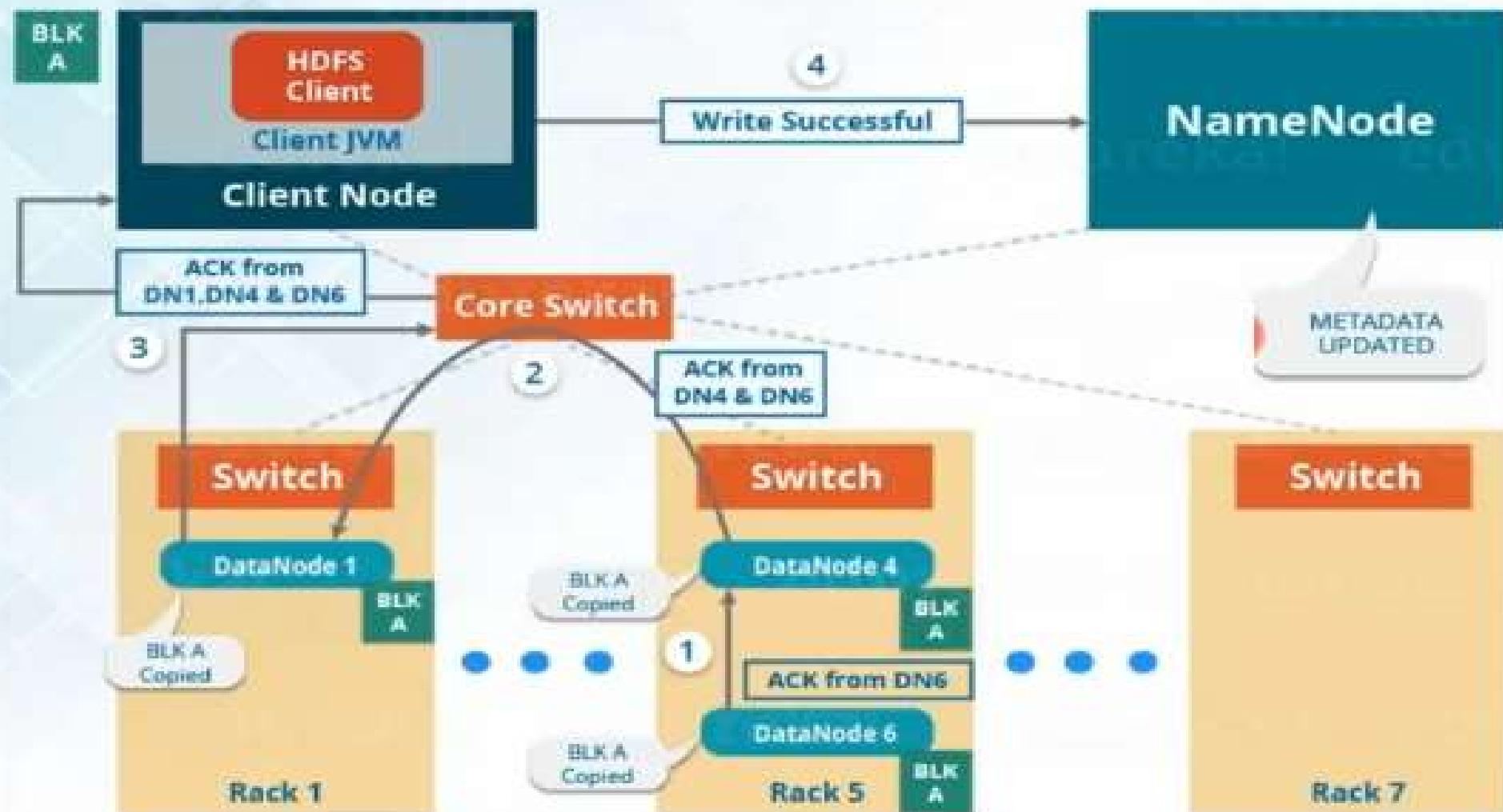
HDFS Write

Setting up HDFS - Write Pipeline



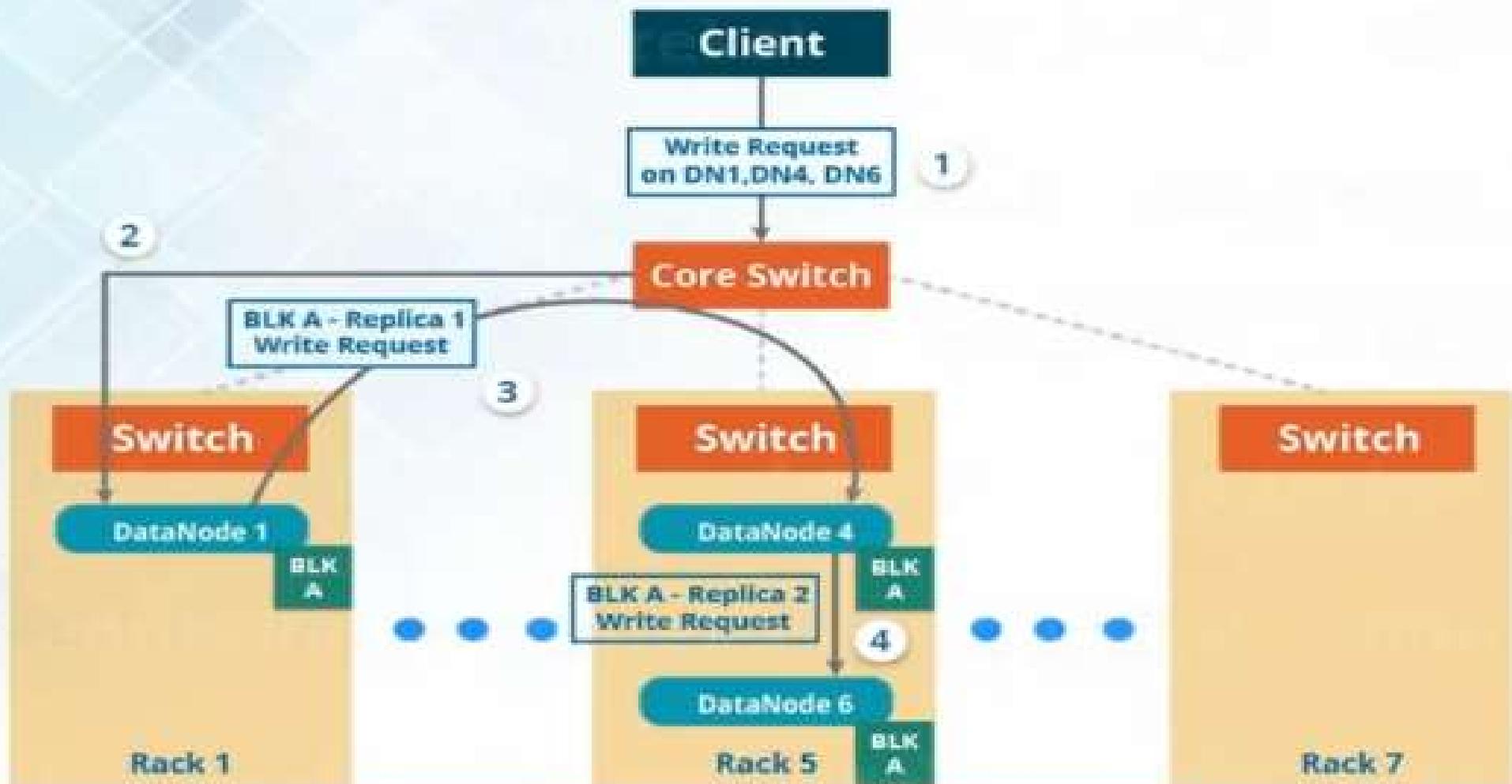
HDFS Write - Ack

Acknowledgement in HDFS - Write



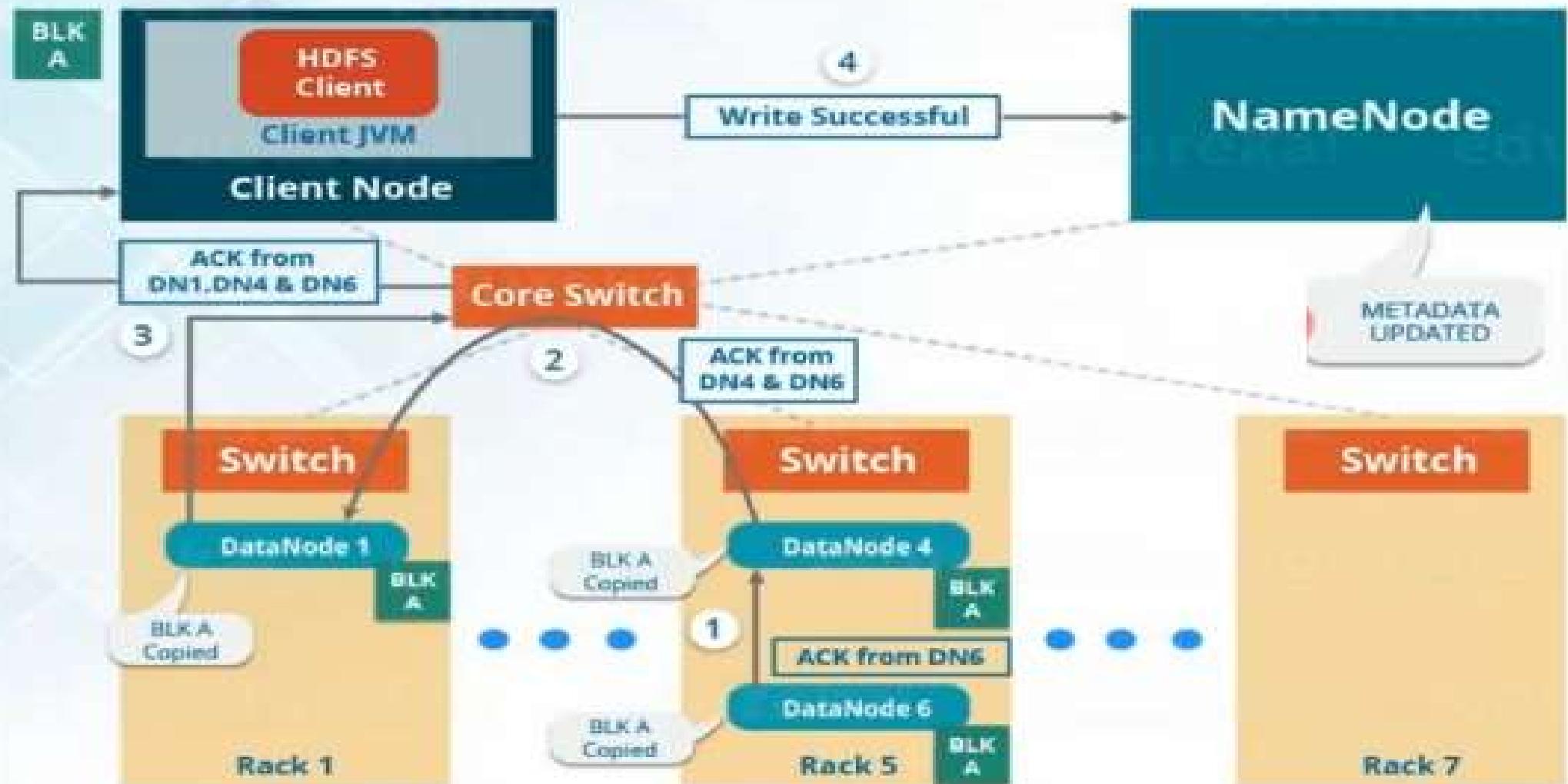
Write Request

HDFS - Write Pipeline

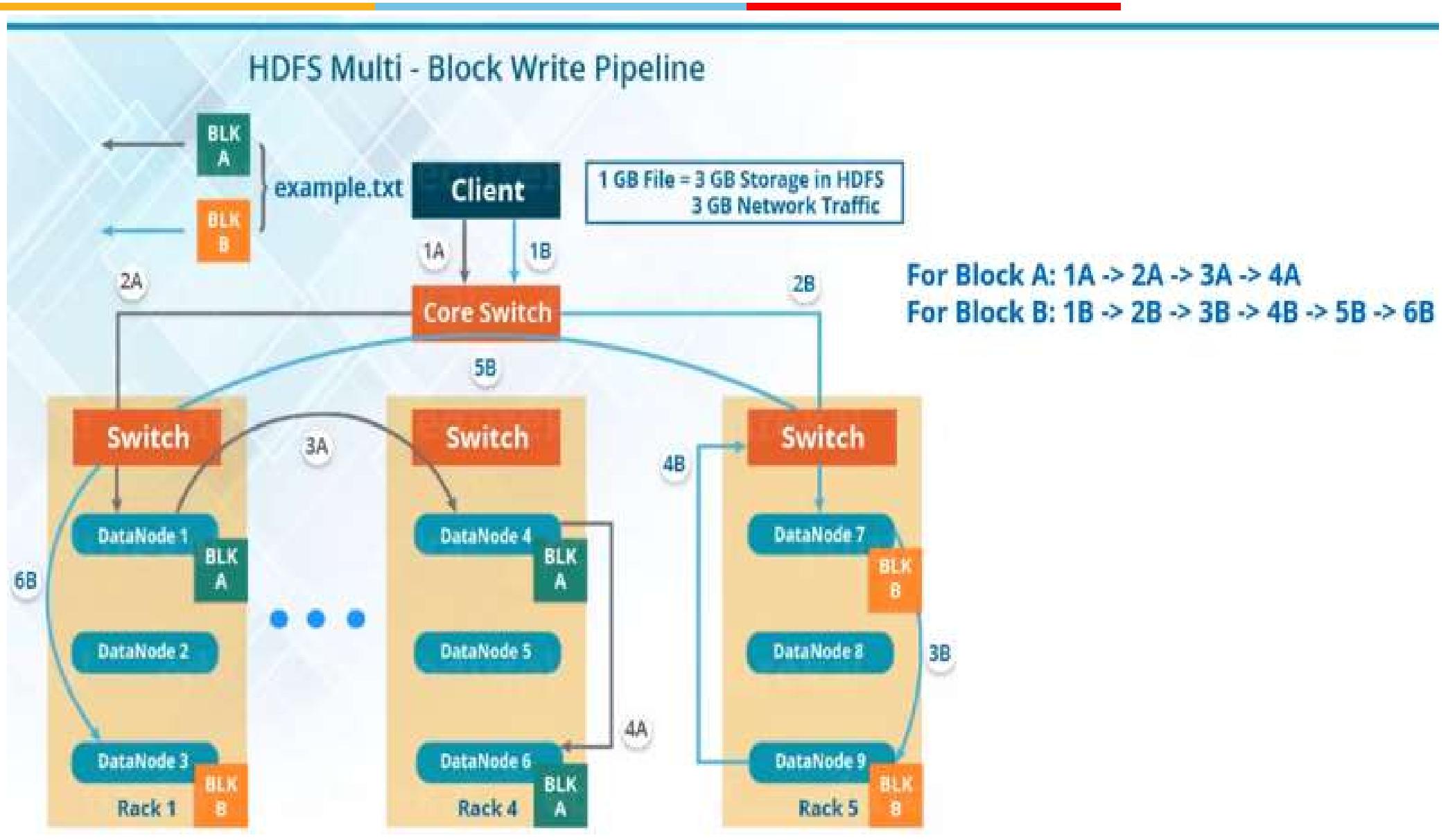


Ack

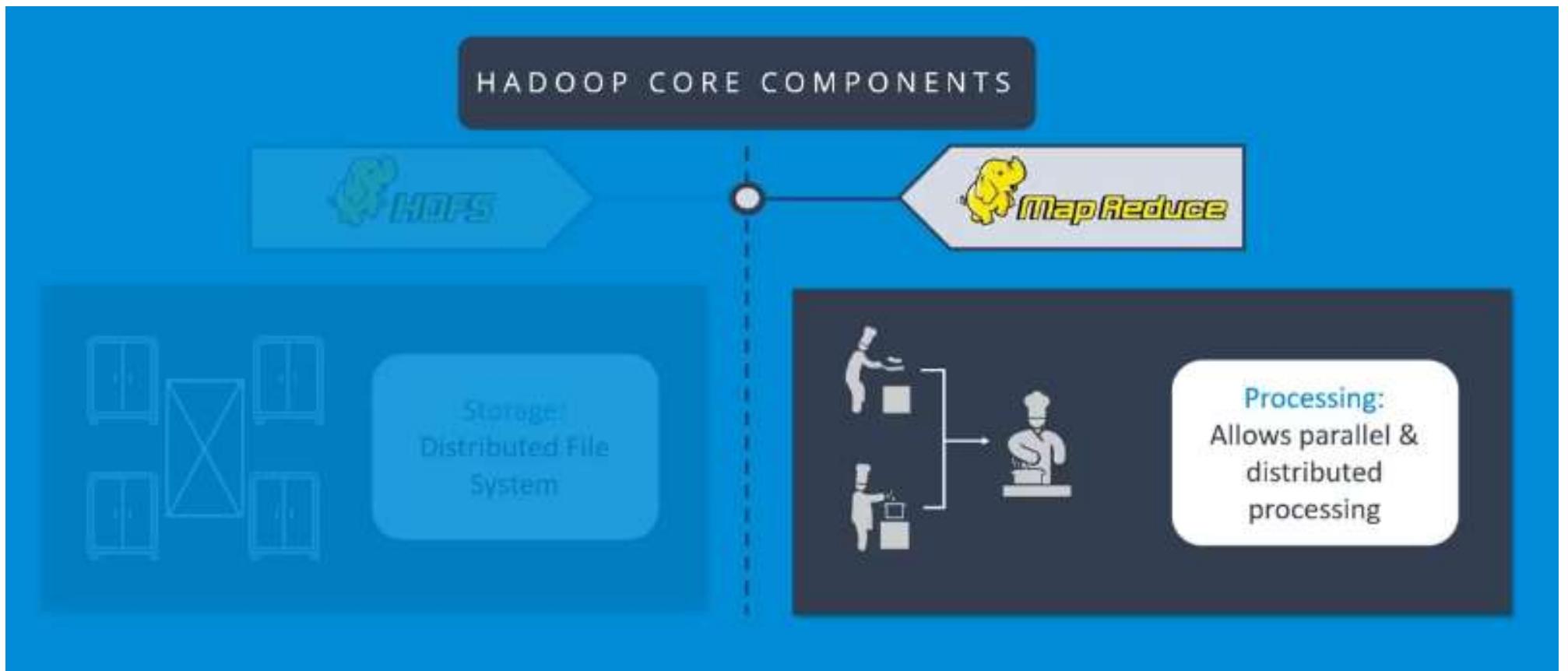
Acknowledgement in HDFS - Write



Multi Data Block write



MapReduce Layer



Why Map-Reduce?

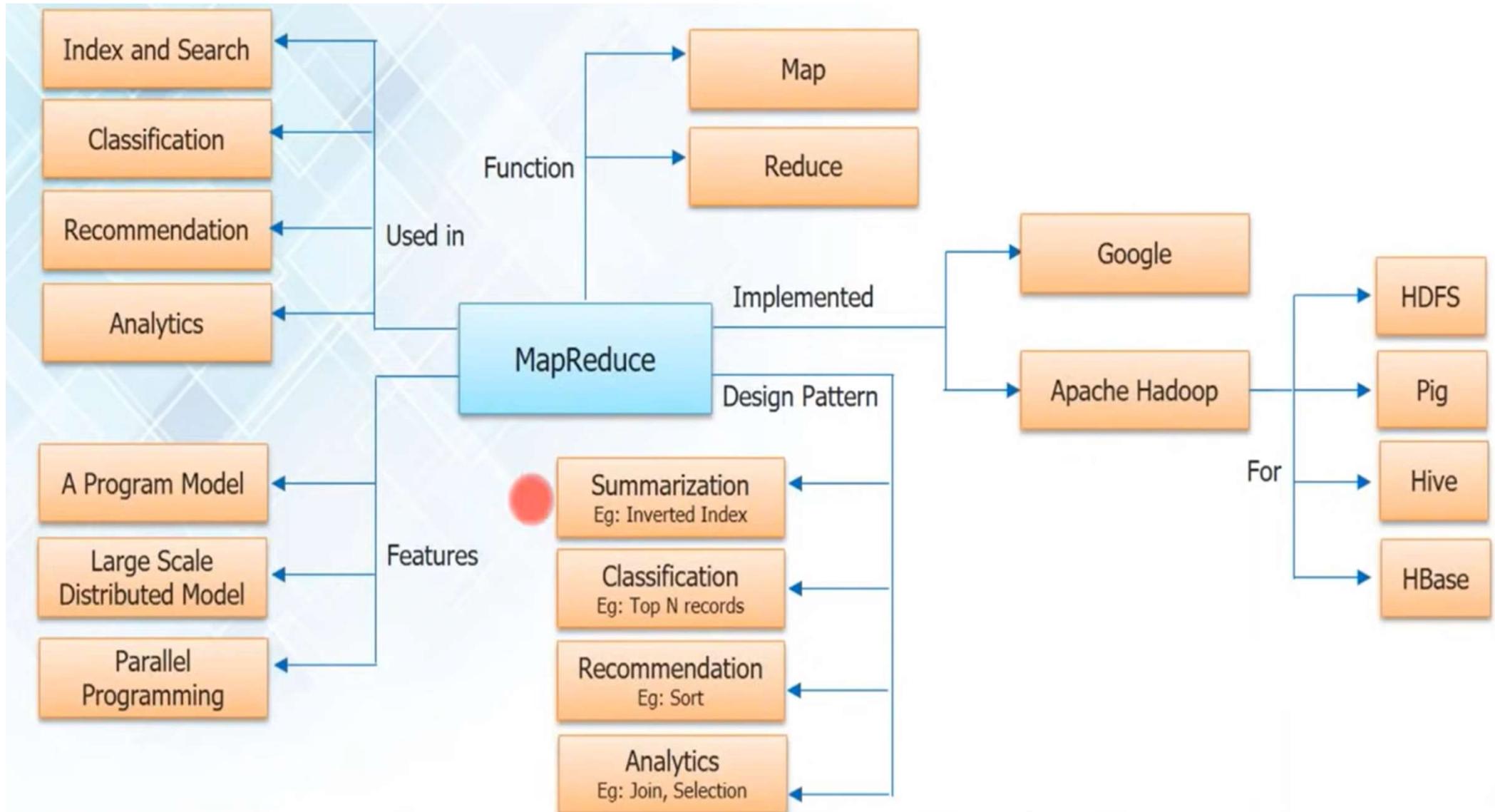


Process Big Data that resides in Hadoop HDFS is not stored in traditional fashion. Complete data is no more available at one place. So we need some processing mechanism that can go to all those locations where the data is available and process is required



- *Hadoop MapReduce is the processing component of Apache Hadoop*
- *It processes data parallelly in distributed environment*

Bird Eye View



Parallel Processing scenario

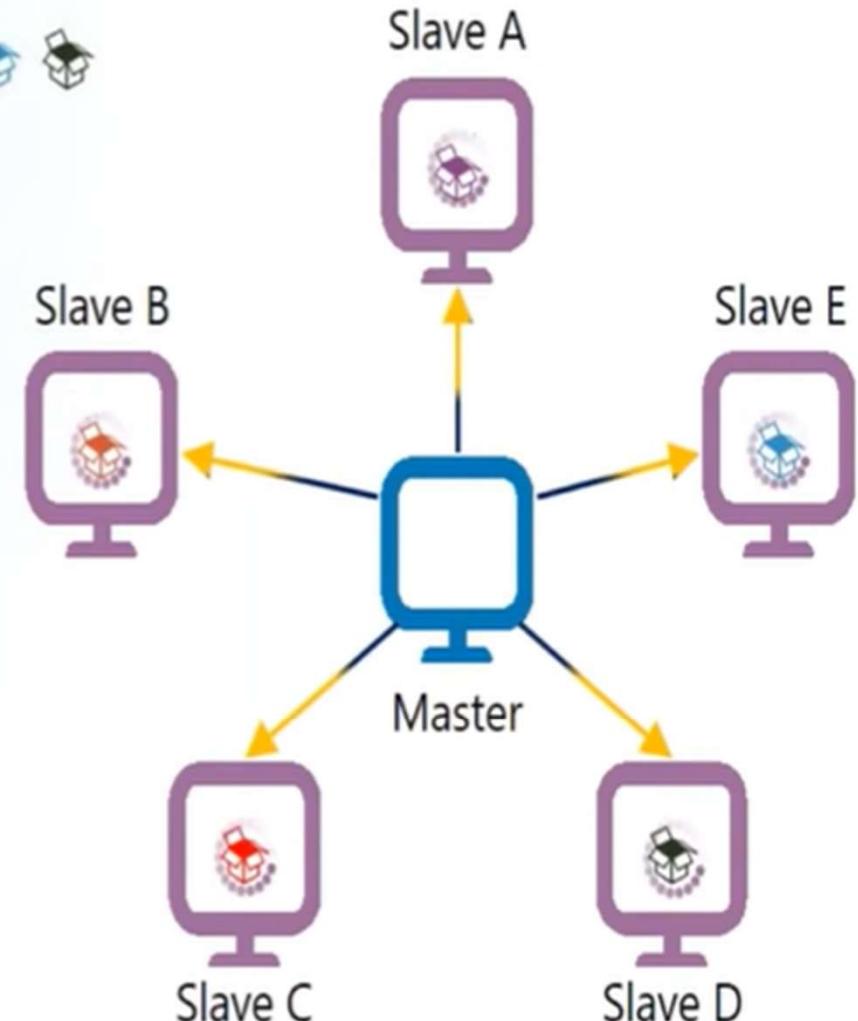
Data → 

Map Reduce Follow Data Locality Concept

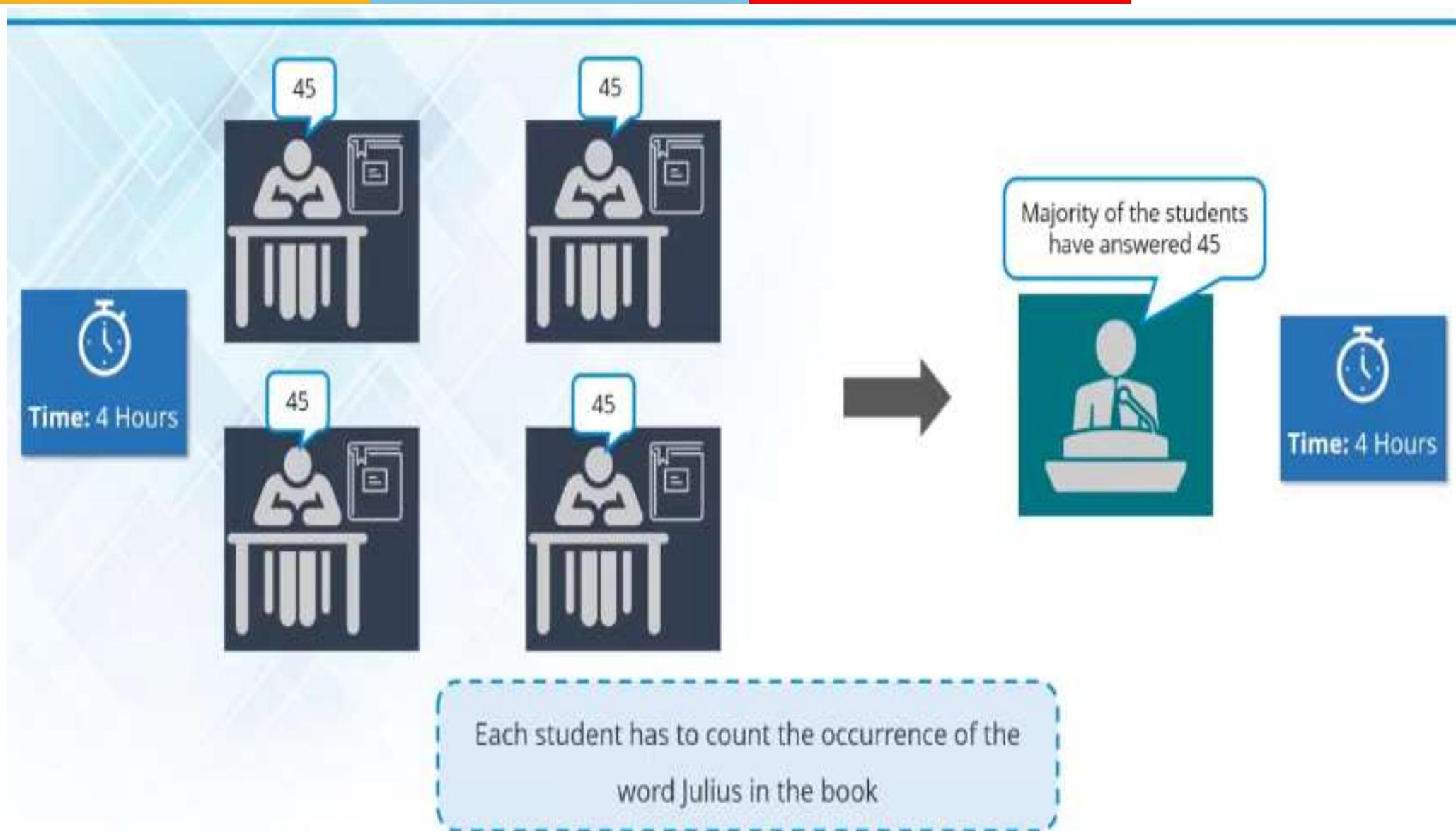
- Data is processed in parallel
- Processing becomes fast

Moving the Processing Logic to
slave machines.

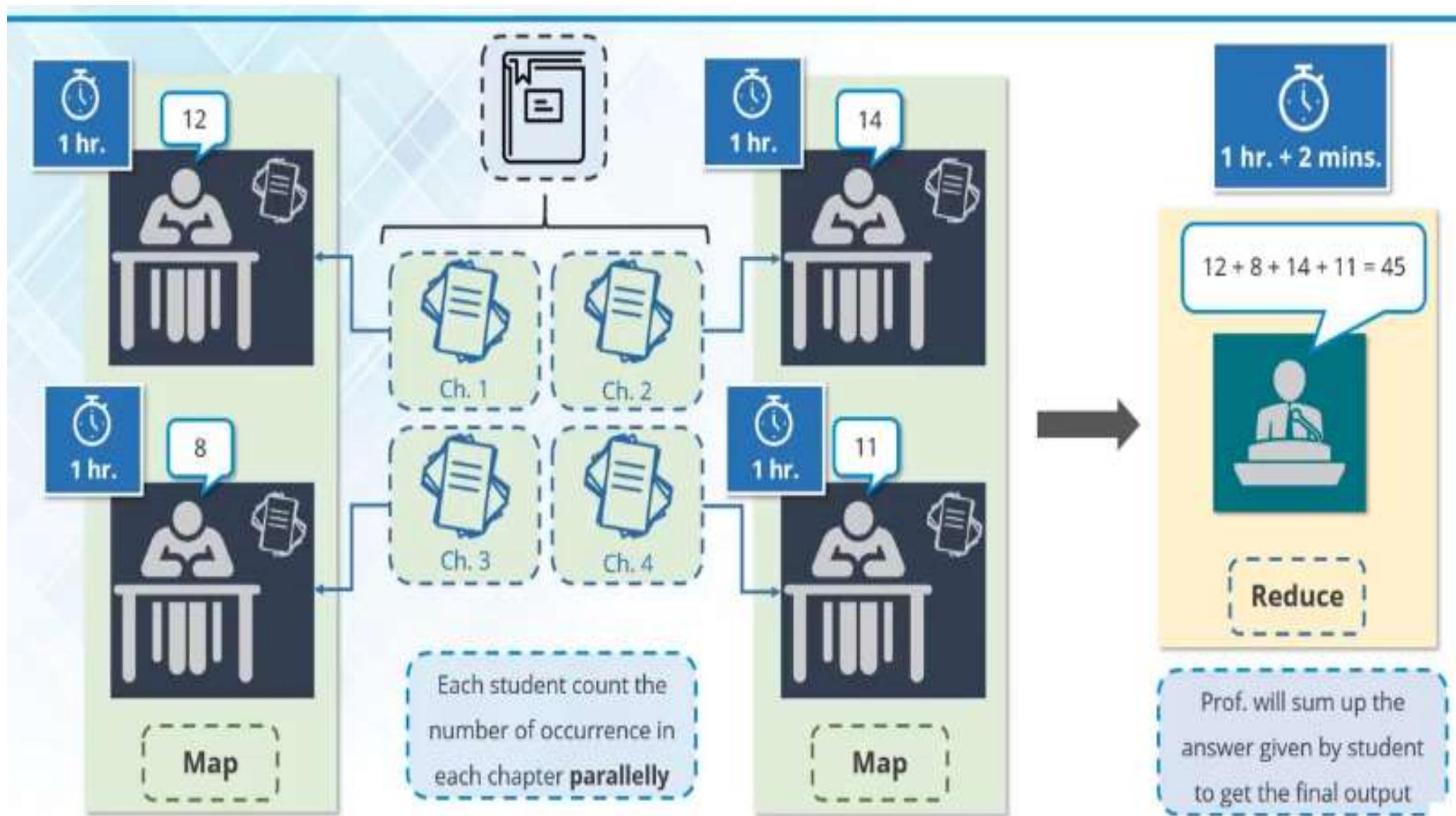
This saves lot of bandwidth



Word Count in a Book – Normal mode

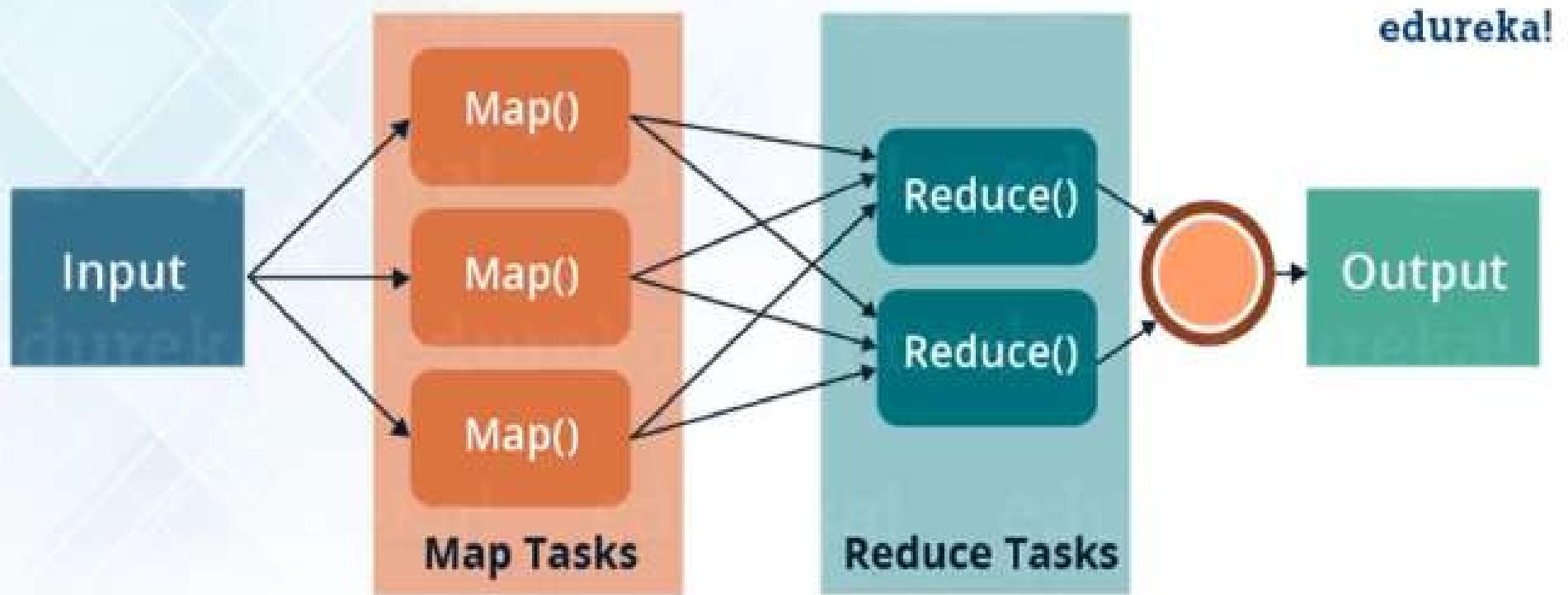


MapReduce Operation - Example



Exact Flow - MapReduce

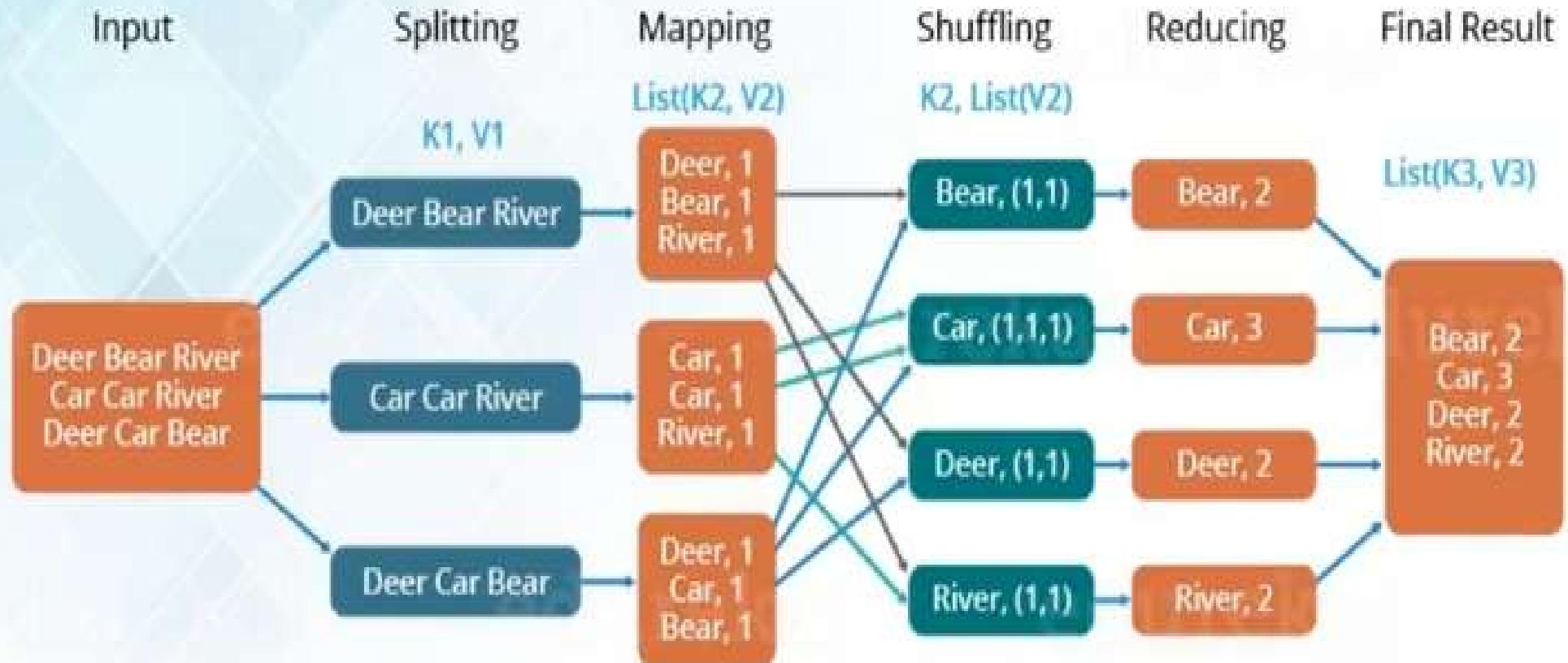
MapReduce is a **programming framework** that allows us to perform **distributed** and **parallel** processing on large data sets in a distributed environment



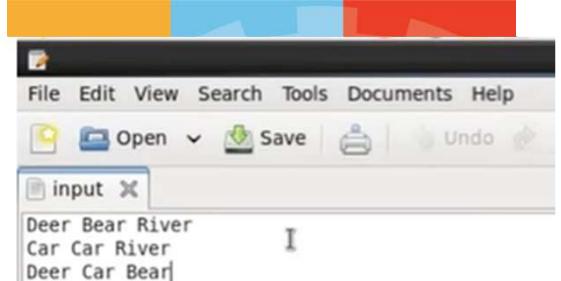
Steps with example

The Overall MapReduce Word Count Process

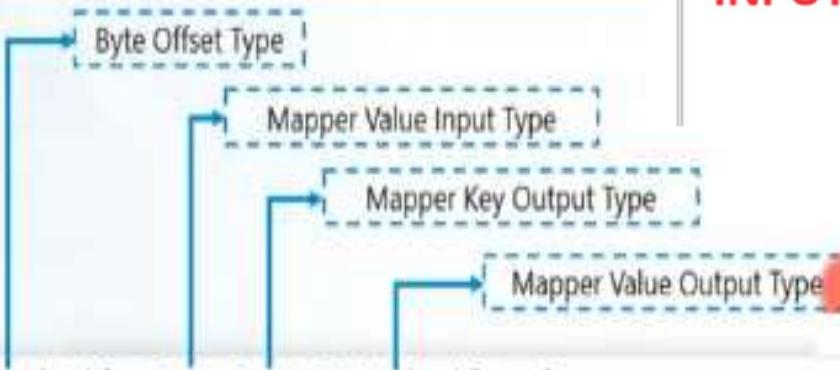
edureka!



Mapper Code –Word Count Example



INPUT FILE



```
public static class Map extends Mapper<LongWritable, Text, Text, IntWritable> {

    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {

        String line = value.toString();
        StringTokenizer tokenizer = new StringTokenizer(line);
        while (tokenizer.hasMoreTokens()) {
            value.set(tokenizer.nextToken());
            context.write(value, new IntWritable(1));
        }
    }
}
```

Mapper Input:

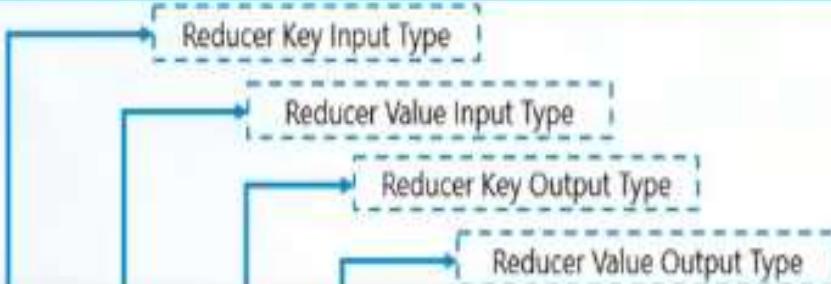
- The key is nothing but the offset of each line in the text file: *LongWritable*
- The value is each individual: *Text*

Mapper Output:

- The key is the tokenized words: *Text*
- We have the hardcoded value in our case which is 1: *IntWritable*
- Example – Dear 1, Bear 1, etc.

Reducer Code –Word Count Example

```
public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable> {  
  
    public void reduce(Text key, Iterable<IntWritable> values,Context context)  
    throws IOException,InterruptedException {  
  
        int sum=0;  
        for(IntWritable x: values)  
        {  
            sum+=x.get();  
        }  
        context.write(key, new IntWritable(sum));  
    }  
}
```



Reducer Input:

- Keys are unique words which have been generated after the sorting and shuffling phase: Text
- The value is a list of integers corresponding to each key: IntWritable
- Example: Bear, [1, 1], etc.

Reducer Output:

- The key is all the unique words present in the input text file: Text
- The value is the number of occurrences of each of the unique words: IntWritable
- Example: Bear, 2; Car, 3, etc. .

Driver Code for Configuration

In the driver class, we set the configuration of our MapReduce job to run in Hadoop

```
Configuration conf= new Configuration();
Job job = new Job(conf,"My Word Count Program");
job.setJarByClass(WordCount.class);
job.setMapperClass(Map.class);
job.setReducerClass(Reduce.class);
job.setOutputKeyClass(Text.class);

job.setOutputValueClass(IntWritable.class);
job.setInputFormatClass(TextInputFormat.class);
job.setOutputFormatClass(TextOutputFormat.class);
Path outputPath = new Path(args[1]);

//Configuring the input/output path from the filesystem into the job
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
```

- Specify the name of the job , the data type of input/output of the mapper and reducer
- Specify the names of the mapper and reducer classes.
- Path of the input and output folder
- The method setInputFormatClass () is used for specifying the unit of work for mapper
- Main() method is the entry point for the driver

NameNode Summary

Namenode information	
localhost:50070/dshealth.html#tab-overview	
Heap Memory used 36.64 MB of 119.94 MB Heap Memory. Max Heap Memory is 966.69 MB.	
Non Heap Memory used 55.88 MB of 56.96 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.	
Configured Capacity:	35.66 GB
DFS Used:	18.16 MB (0.05%)
Non DFS Used:	9.32 GB
DFS Remaining:	26.33 GB (73.83%)
Block Pool Used:	18.16 MB (0.05%)
DataNodes usages% (Min/Median/Max/stdDev):	0.05% / 0.05% / 0.05% / 0.00%
Live Nodes	1 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	12

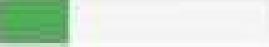
Live Node Details

Namenode information x +

localhost:50070/dfshealth.html#tab-datanode

In operation

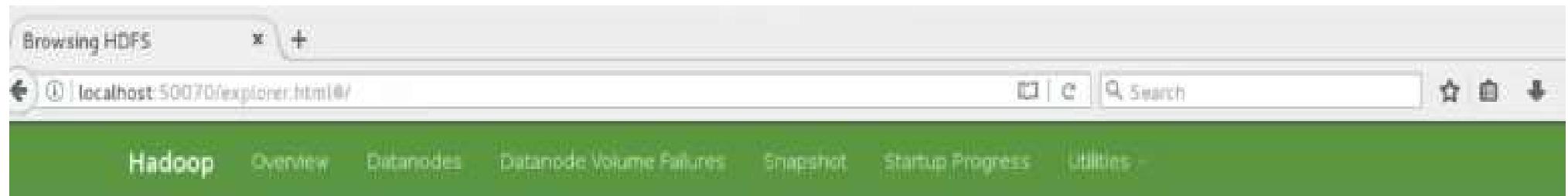
Show entries Search

Node	Http Address	Last contact	Capacity	Blocks	Block pool used	Version
localhost:50010 (127.0.0.1:50010)	localhost:50075	1s	35.06 GB 	19	18.17 MB (0.05%)	2.8.1

Showing 1 to 1 of 1 entries

Previous 1 Next

NameNode Files



Browsing HDFS x +

localhost:50070/explorer.html#/

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

Browse Directory

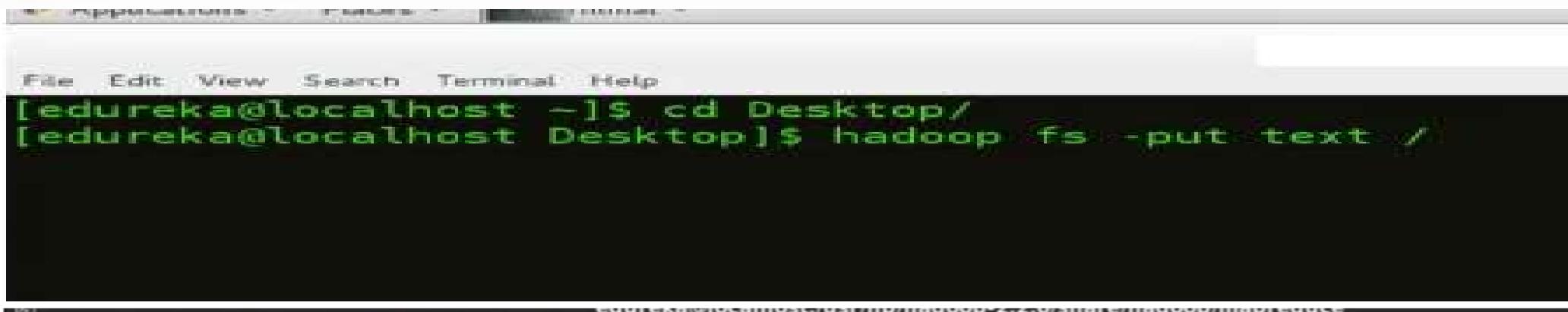


/

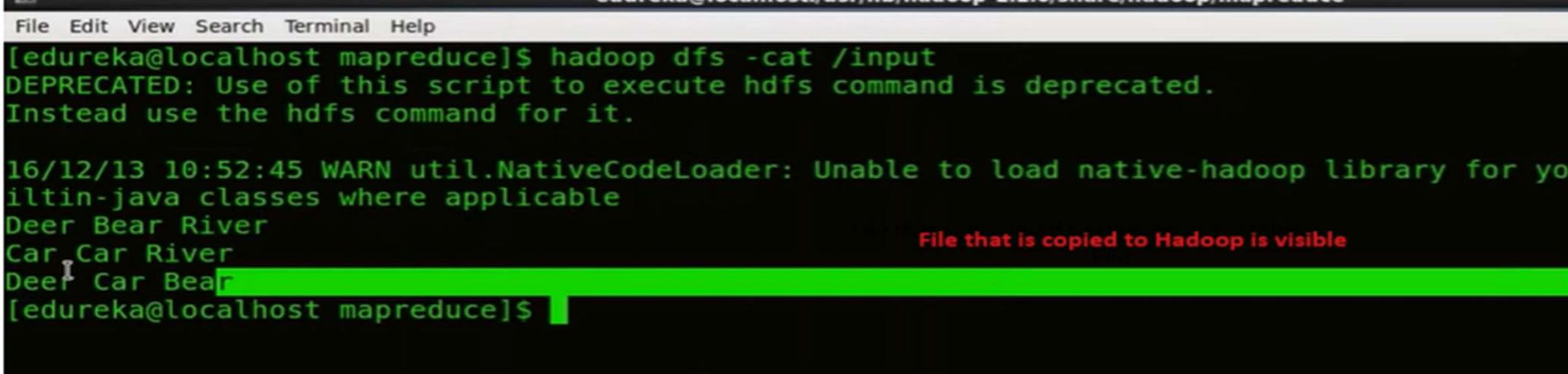
Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	Actions
-rw-r--r--	root	supergroup	499.2 kB	Dec 31 15:01	1	128 MB	olympix_data.csv	
-rw-r--r--	edureka	supergroup	45 B	Jan 03 16:26	1	128 MB	text	
drwxr-xr-x	edureka	supergroup	0 B	Oct 17 09:53	0	0 B	system	
drwxrwxrwx	root	supergroup	0 B	Dec 31 14:58	0	0 B	tmp	

Sample Text File



```
[edureka@localhost ~]$ cd Desktop/
[edureka@localhost Desktop]$ hadoop fs -put text /
```



```
[edureka@localhost mapreduce]$ hadoop dfs -cat /input
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

16/12/13 10:52:45 WARN util.NativeCodeLoader: Unable to load native-hadoop library for yo
iltin-java classes where applicable
Deer Bear River
Car Car River
Deer Car Bear
[edureka@localhost mapreduce]$
```

File that is copied to Hadoop is visible

Execution

```
edureka@localhost:/usr/lib/hadoop-2.2.0/share/hadoop/mapreduce
File Edit View Search Terminal Help
[edureka@localhost mapreduce]$ hadoop jar hadoop-mapreduce-examples-2.2.0.jar wordcount /input /firstExampleOut
16/12/13 10:54:52 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in java classes where applicable
16/12/13 10:54:55 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
16/12/13 10:54:57 INFO input.FileInputFormat: Total input paths to process : 1
16/12/13 10:54:58 INFO mapreduce.JobSubmitter: number of splits:1
16/12/13 10:54:58 INFO Configuration.deprecation: user.name is deprecated. Instead, use mapreduce.job.user.name
16/12/13 10:54:58 INFO Configuration.deprecation: mapred.jar is deprecated. Instead, use mapreduce.job.jar
16/12/13 10:54:58 INFO Configuration.deprecation: mapred.output.value.class is deprecated. Instead, use mapreduce.job.output.value.class
16/12/13 10:54:58 INFO Configuration.deprecation: mapreduce.combine.class is deprecated. Instead, use mapreduce.job.output.key.class
16/12/13 10:54:58 INFO Configuration.deprecation: mapred.working.dir is deprecated. Instead, use mapreduce.working.dir
16/12/13 10:54:58 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1481601033699_0002
16/12/13 10:55:00 INFO impl.YarnClientImpl: Submitted application application_1481601033699_0002 to ResourceManager at /0.0.0.0:8032
16/12/13 10:55:01 INFO mapreduce.Job: The url to track the job: http://localhost:8088/proxy/application_1481601033699_0002/
16/12/13 10:55:01 INFO mapreduce.Job: Running job: job_1481601033699_0002
16/12/13 10:55:57 INFO mapreduce.Job: Job job_1481601033699_0002 running in uber mode : false
16/12/13 10:55:57 INFO mapreduce.Job: map 0% reduce 0%
```

Execution

```
File Edit View Search Terminal Help  
[edureka@localhost Desktop]$ hadoop jar abc.jar WordCount /text /output
```

I

```
File Edit View Search Terminal Help  
Job Counters  
    Launched map tasks=1  
    Launched reduce tasks=1  
    Data-local map tasks=1  
    Total time spent by all maps in occupied slots (ms)=10030  
    Total time spent by all reduces in occupied slots (ms)=11901  
    Total time spent by all map tasks (ms)=10030  
    Total time spent by all reduce tasks (ms)=11901  
    Total vcore-milliseconds taken by all map tasks=10030  
    Total vcore-milliseconds taken by all reduce tasks=11901  
    Total megabyte-milliseconds taken by all map tasks=10270720  
    Total megabyte-milliseconds taken by all reduce tasks=12186624  
Map -Reduce Framework  
    Map input records=3  
    Map output records=9  
    Map output bytes=81  
    Map output materialized bytes=105  
    Input split bytes=91  
    Combine input records=0  
    Combine output records=0
```

Ack

File Edit View Search Terminal Help

```
[edureka@localhost mapreduce]$ hadoop dfs -ls /firstExampleOut
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

16/12/13 10:58:00 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in java classes where applicable
Found 2 items
-rw-r--r-- 1 edureka supergroup          0 2016-12-13 10:57 /firstExampleOut/_SUCCESS
-rw-r--r-- 1 edureka supergroup          28 2016-12-13 10:57 /firstExampleOut/part-r-00000
[edureka@localhost mapreduce]$ █
```

```
[edureka@localhost mapreduce]$ hadoop dfs -ls /firstExampleOut
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

16/12/13 10:58:00 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in java classes where applicable
Found 2 items
-rw-r--r-- 1 edureka supergroup          0 2016-12-13 10:57 /firstExampleOut/_SUCCESS
-rw-r--r-- 1 edureka supergroup          28 2016-12-13 10:57 /firstExampleOut/part-r-00000
[edureka@localhost mapreduce]$ hadoop dfs -cat /firstExampleOut/part-r-00000
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

16/12/13 10:58:38 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using built-in java classes where applicable
Bear    2
Car     3   █
Deer    2
River   2
[edureka@localhost mapreduce]$ █
```

Result

Browsing HDFS × +

localhost:50070/explorer.html#/

BROWSE DIRECTORY

/

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	root	supergroup	498.2 KB	Dec 31 15:01	1	128 MB	olympix_data.csv
-rw-r--r--	edureka	supergroup	45 B	Jan 03 16:26	1	128 MB	text
drwxr-xr-x	edureka	supergroup	0 B	Jan 03 16:34	0	0 B	output
drwxr-xr-x	edureka	supergroup	0 B	Oct 17 09:53	0	0 B	
drwxrwxrwx	root	supergroup	0 B	Dec 31 14:58	0	0 B	tmp

Browsing HDFS × +

localhost:50070/explorer.html#/output

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

Browse Directory

/output

Show 25 entries Search:

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	edureka	supergroup	0 B	Jan 03 16:34	1	128 MB	SUCCESS
-rw-r--r--	edureka	supergroup	28 B	Jan 03 16:34	1	128 MB	part-r-00000

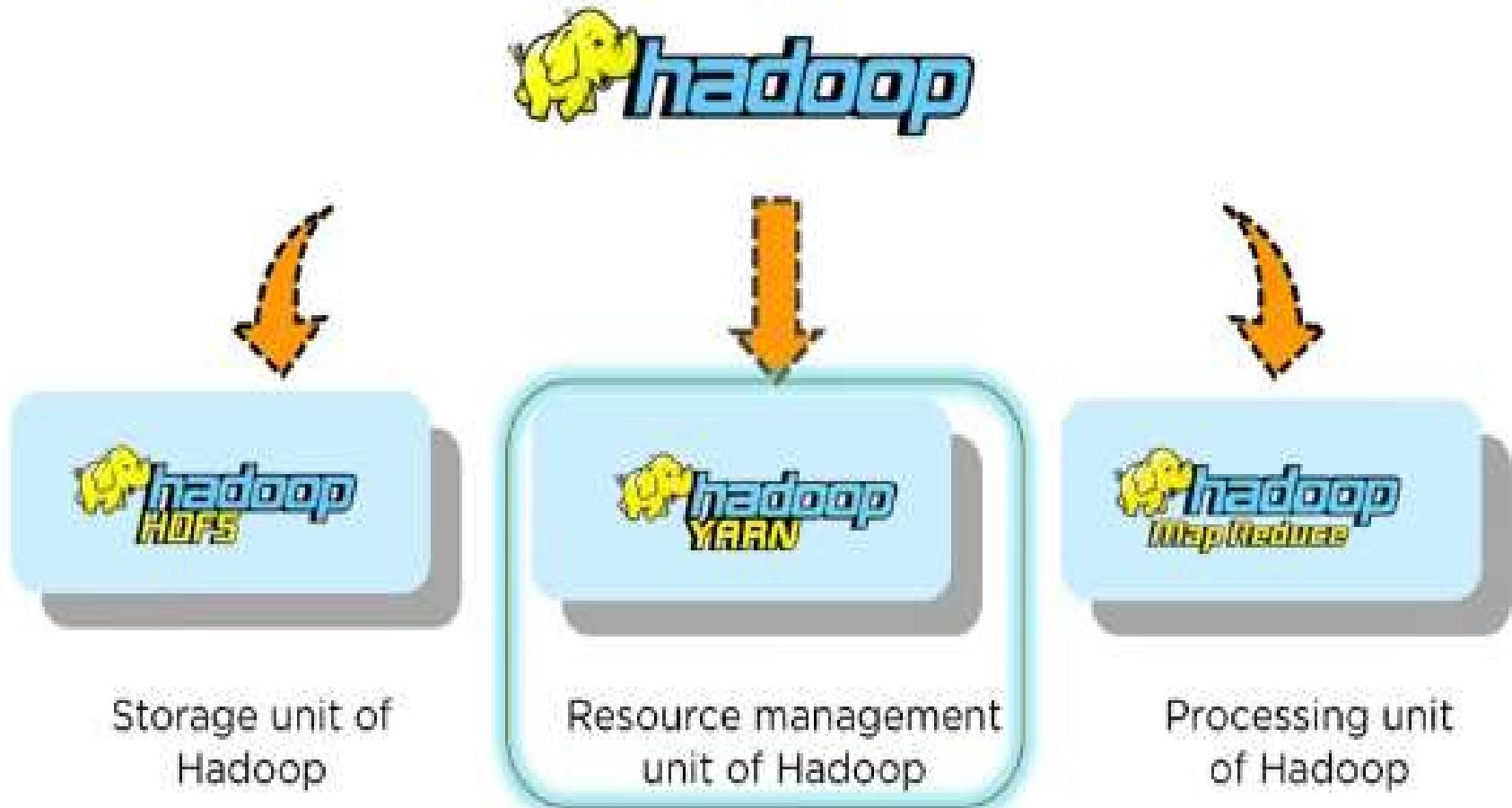
Result-Contd

```
File Edit View Search Terminal Help  
[edureka@localhost Desktop]$ hadoop fs -cat /output/part-r-00000  
Bear 1  
Car 3  
Dear 2  
River 3  
[edureka@localhost Desktop]$
```

Better Alternative

Property	Apache Spark	Hadoop MapReduce
Performance	Lightning-fast cluster computing (100 times faster)	Slower than Spark
Read/Write Cycle	Low	High
Usability	Easy	Requires code for every operation
Realtime Analysis	Supported	Not Supported
Latency	Low	High
Fault Tolerance	Supported	Supported
Security	Low	High
Cost	High (requires a lot of RAM)	Low
Developing Language	Scala	Java
Licenses	Free	Free
SQL Support	Spark SQL	Hive
Scalability	High	High
Machine Learning	MLLib	Apache Mahout
Data Caching	Supported	Not Supported
Hardware Requirements	High-Level hardware	Commodity hardware

Basic Hadoop-2 Architecture



YET
ANOTHER
RESOURCE
NEGOTIATOR

Coming Up.....



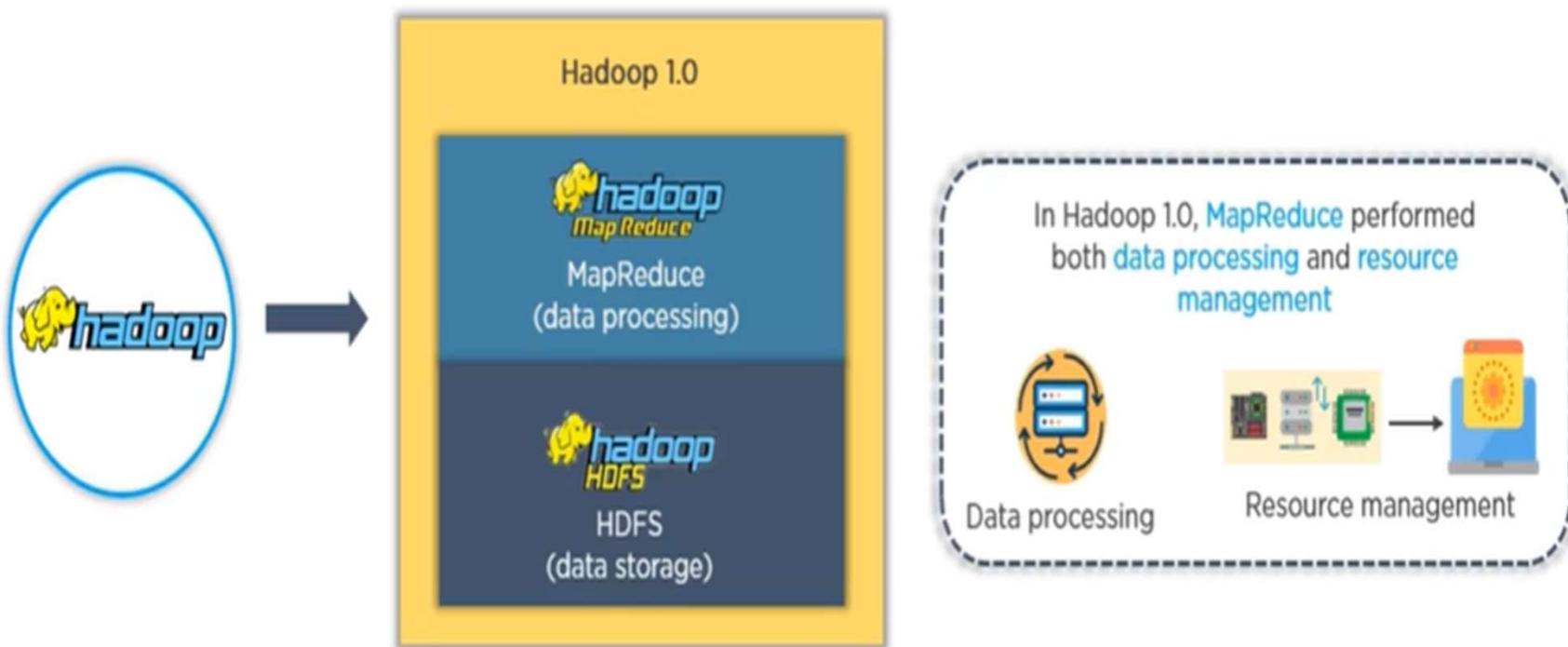
That is all for the day

Thank you

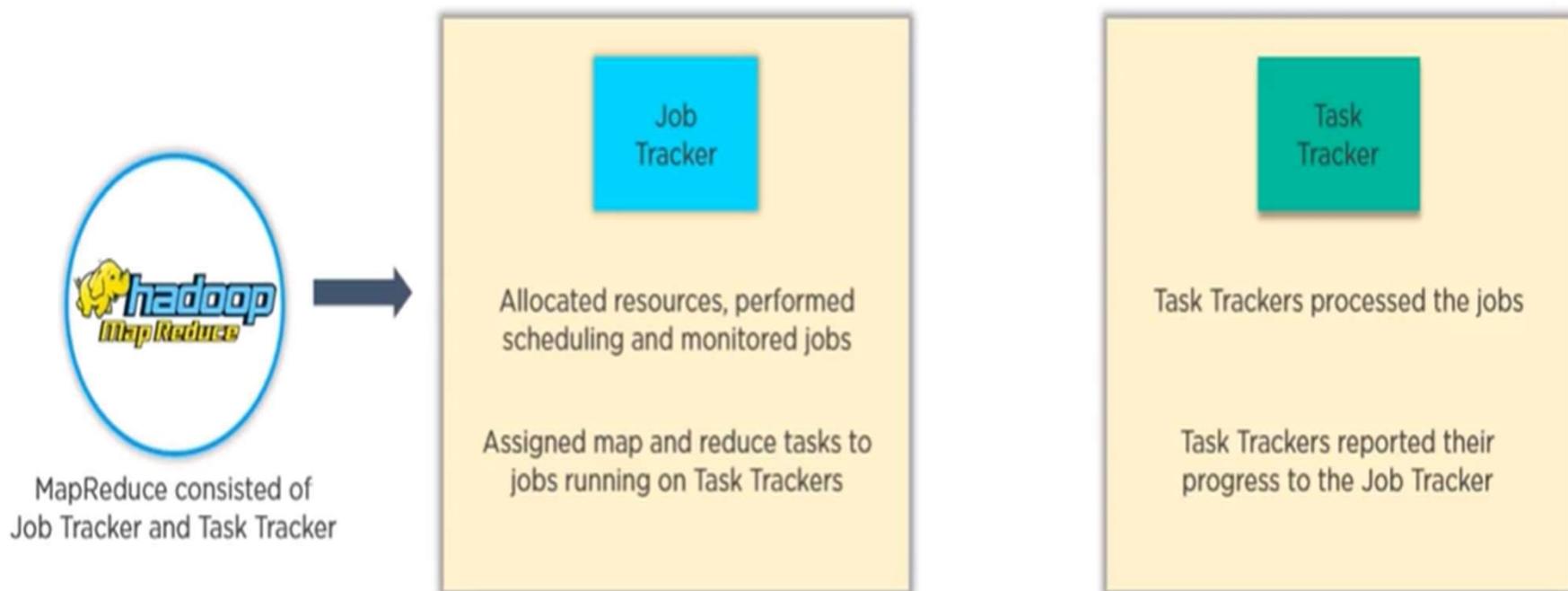
Todays Agenda

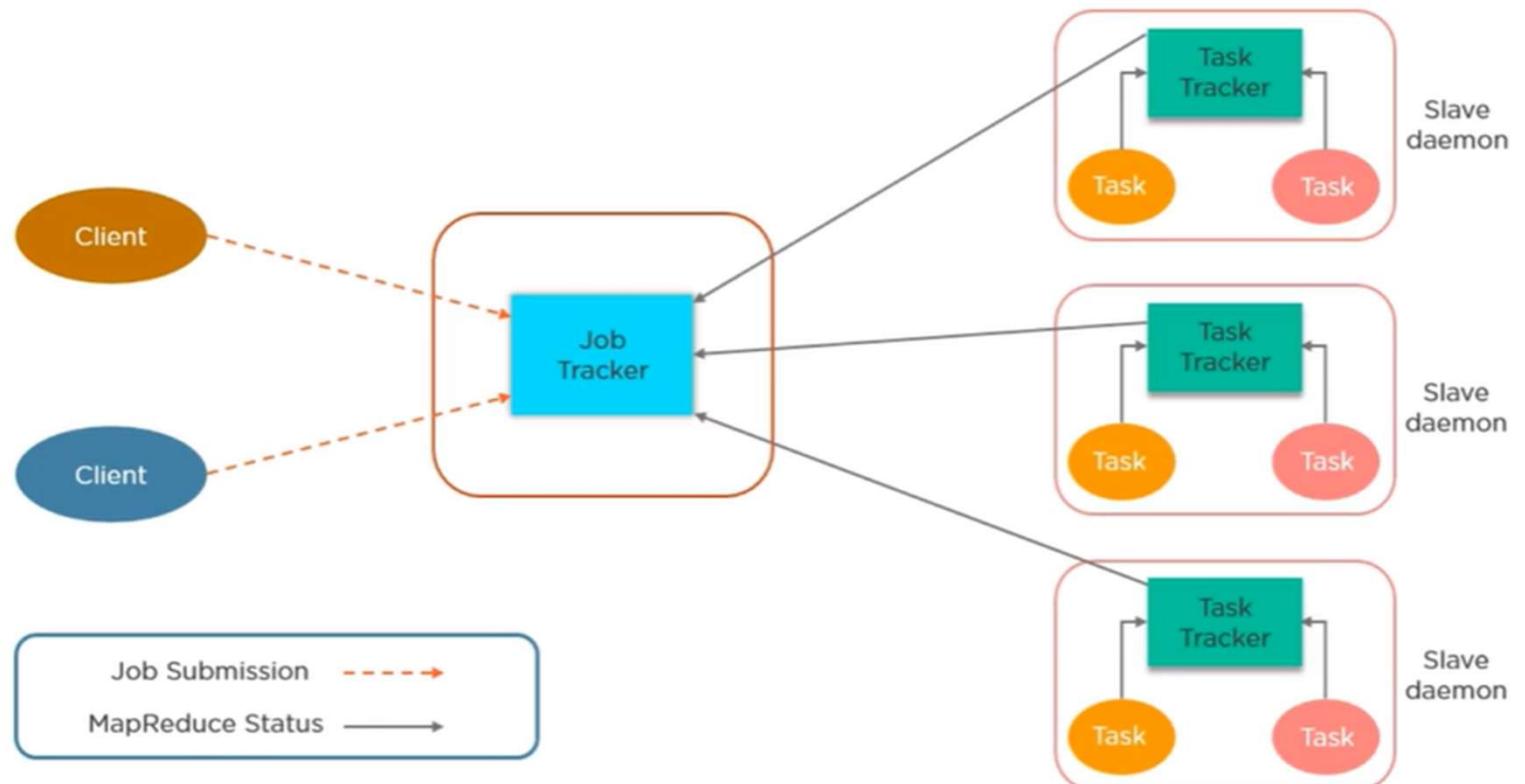


Hadoop-1



Hadoop-1





Author: All rights reserved

Managing jobs using a single job tracker and utilization of computational resources was inefficient in MR 1

Limitations

1 Scalability

Due to a **single JobTracker**, scalability became a bottleneck.
Cannot have a cluster size of more than **4000 nodes** and cannot run more than **40000 concurrent tasks**

2 Availability issue

JobTracker is **single point of failure**. Any failure kills all queued and running jobs. Jobs need to be resubmitted by users

3 Resource Utilization

Due to predefined number of **map** and **reduce slots** for each TaskTracker, **resource utilization** issues occur

4 Limitations in running non-MapReduce applications

Problem in performing **real-time analysis** and running **Ad-hoc query** as MapReduce is batch driven

Solution

Scalability



Can have a cluster size of more than 10,000 nodes and can run more than 1,00,000 concurrent tasks

Compatibility



Applications developed for Hadoop 1 runs on YARN without any disruption or availability issues

Resource utilization



Allows dynamic allocation of cluster resources to improve resource utilization

Multitenancy

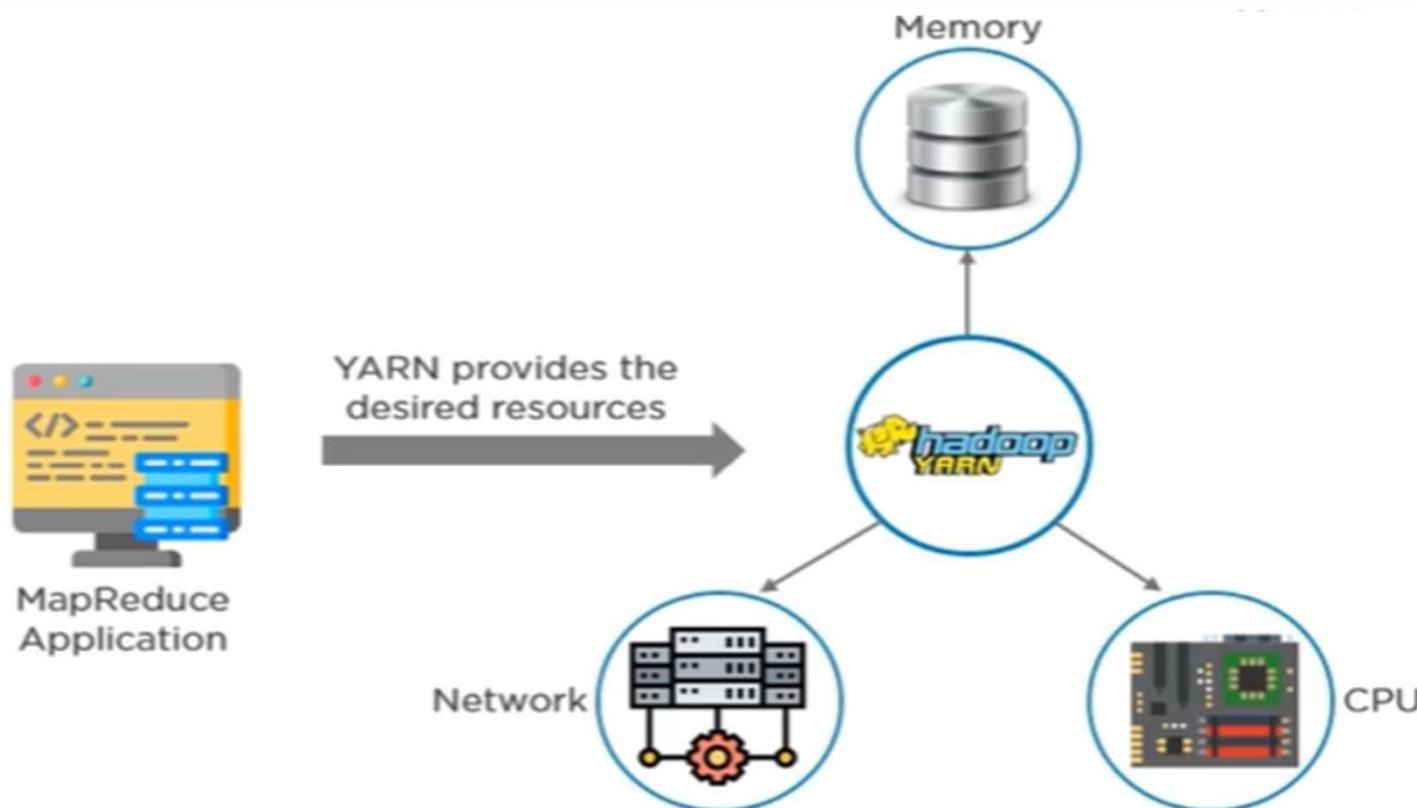


Can use open-source and proprietary data access engines and perform real-time analysis and running ad-hoc query

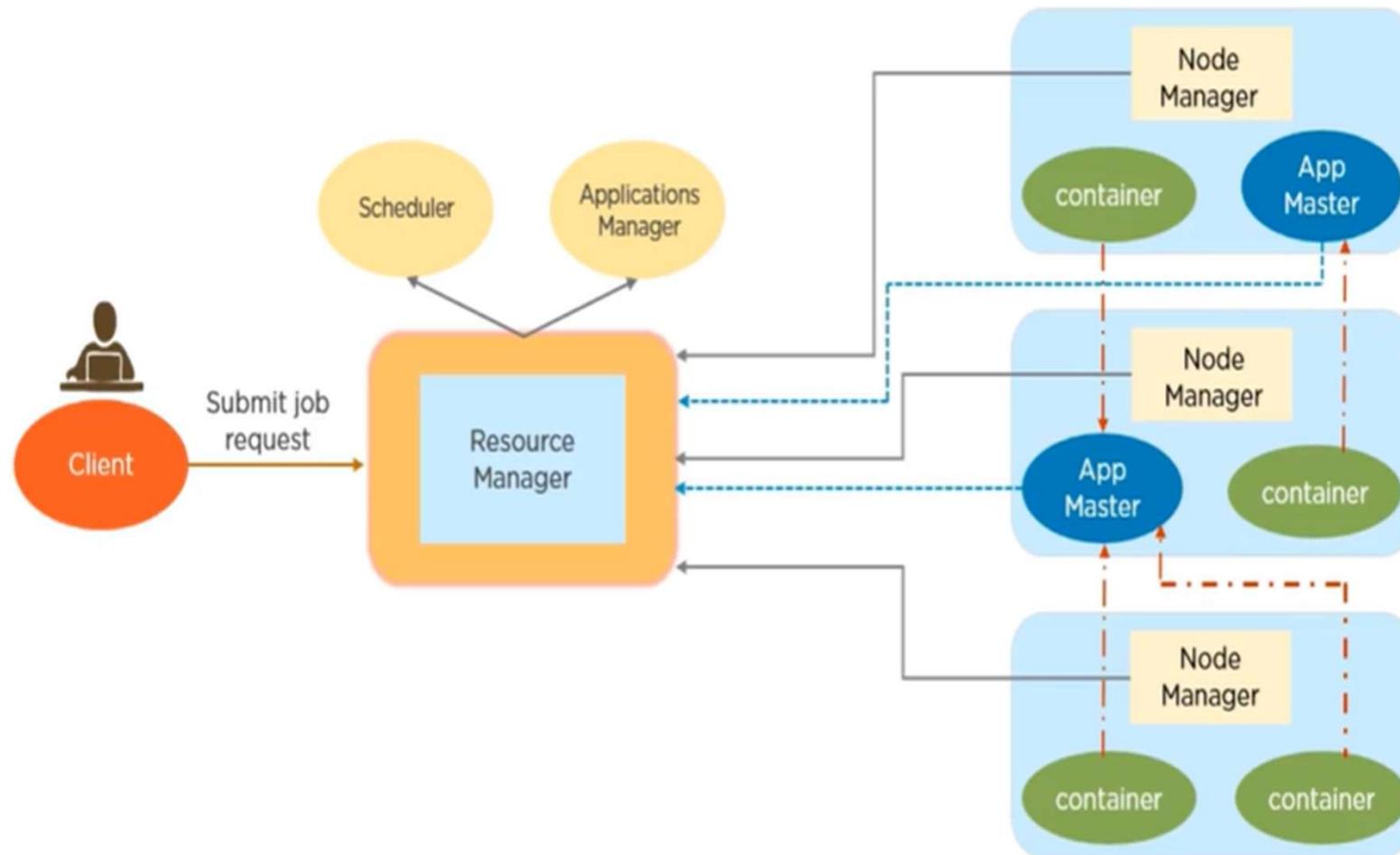
What is YARN ?

YARN – Yet Another Resource Negotiator

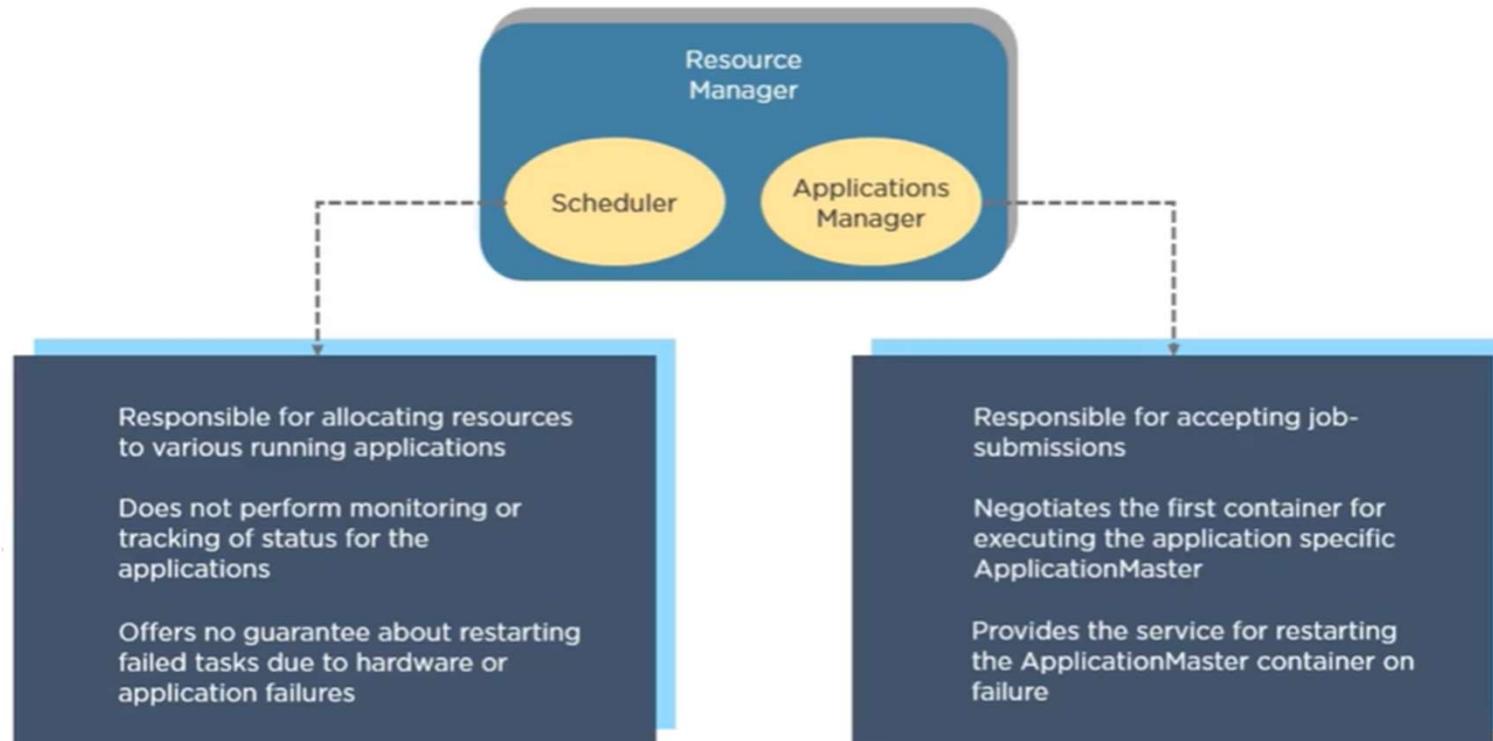
YARN is the cluster resource management layer of the Apache Hadoop Ecosystem, which schedules jobs and assigns resources



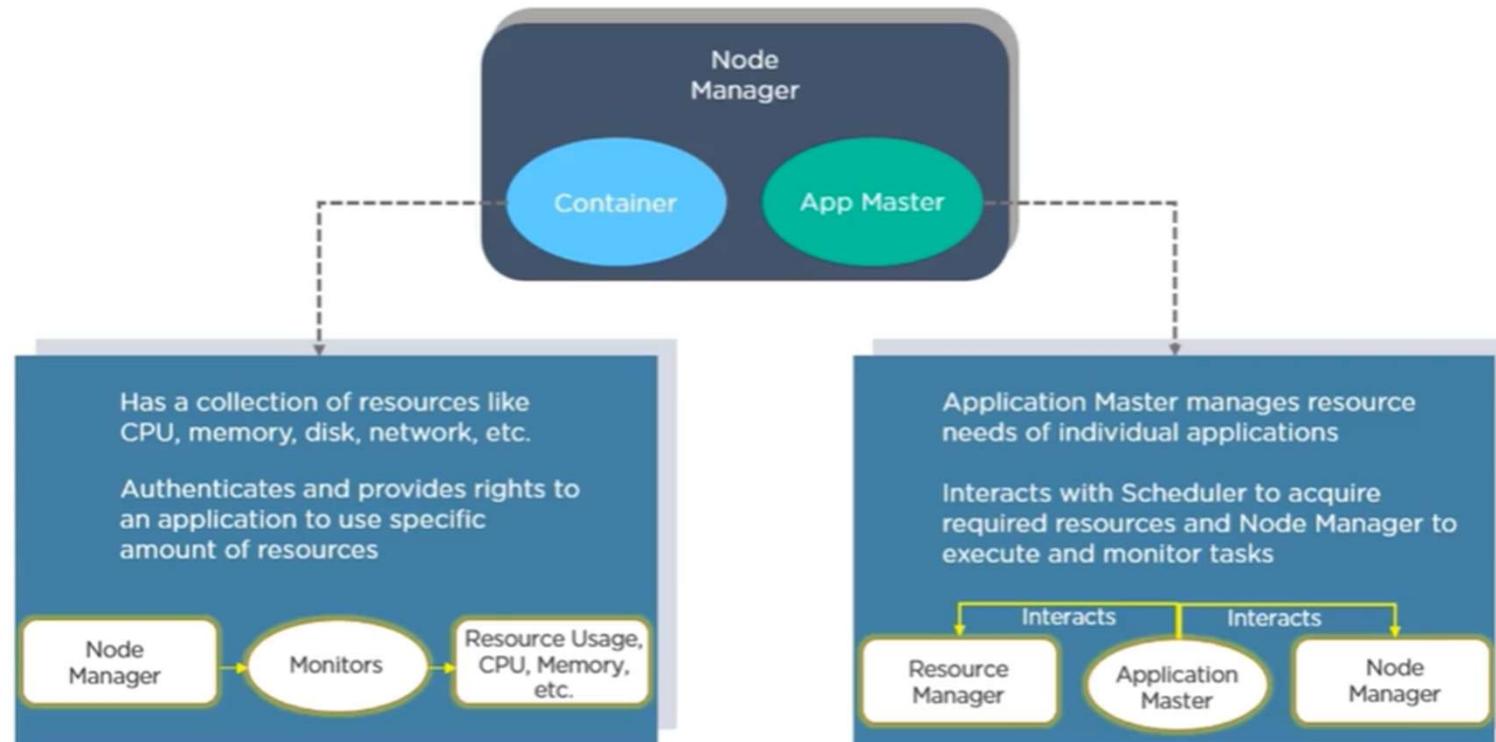
Architecture



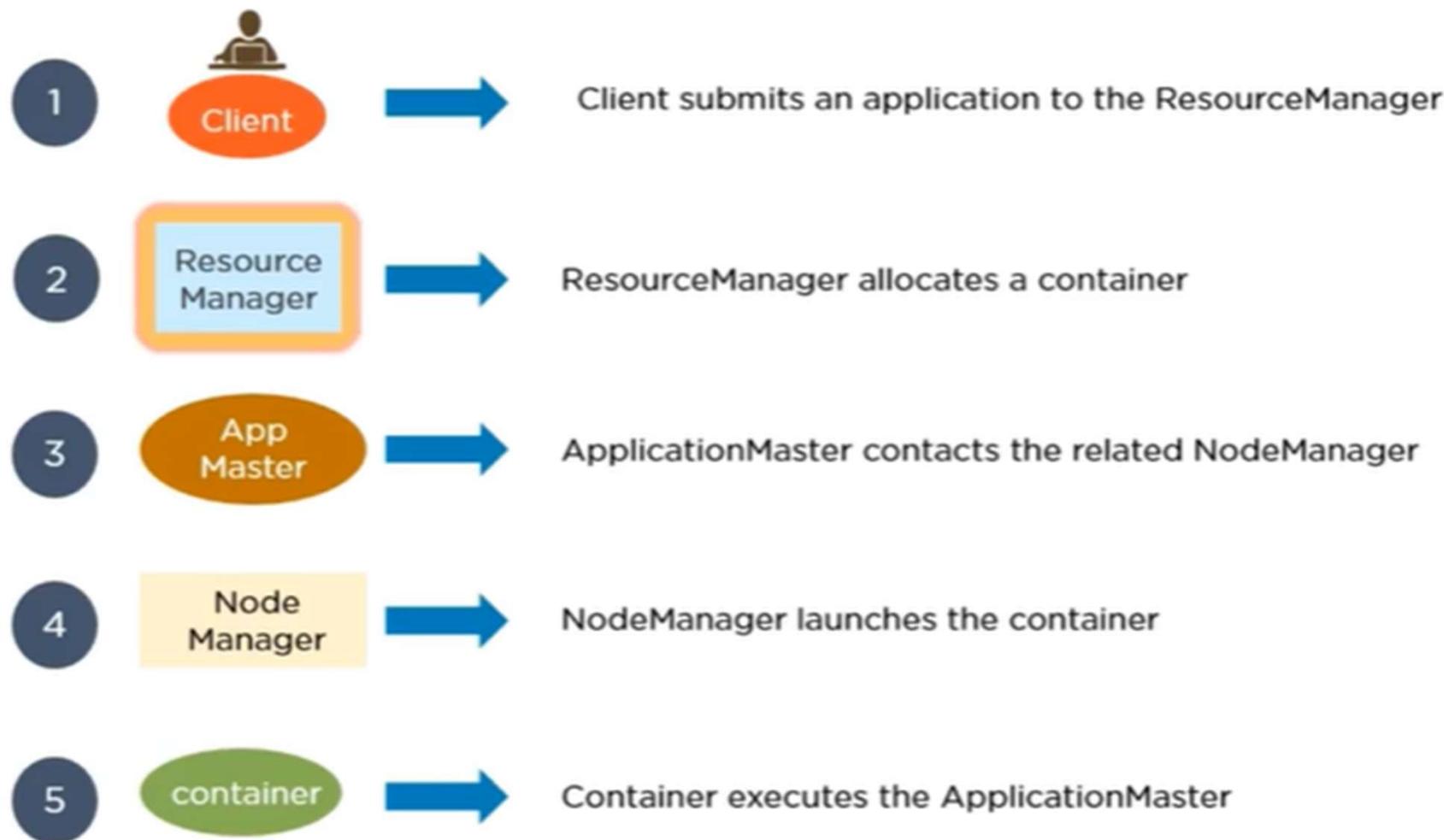
Resource Manager



Node Manager

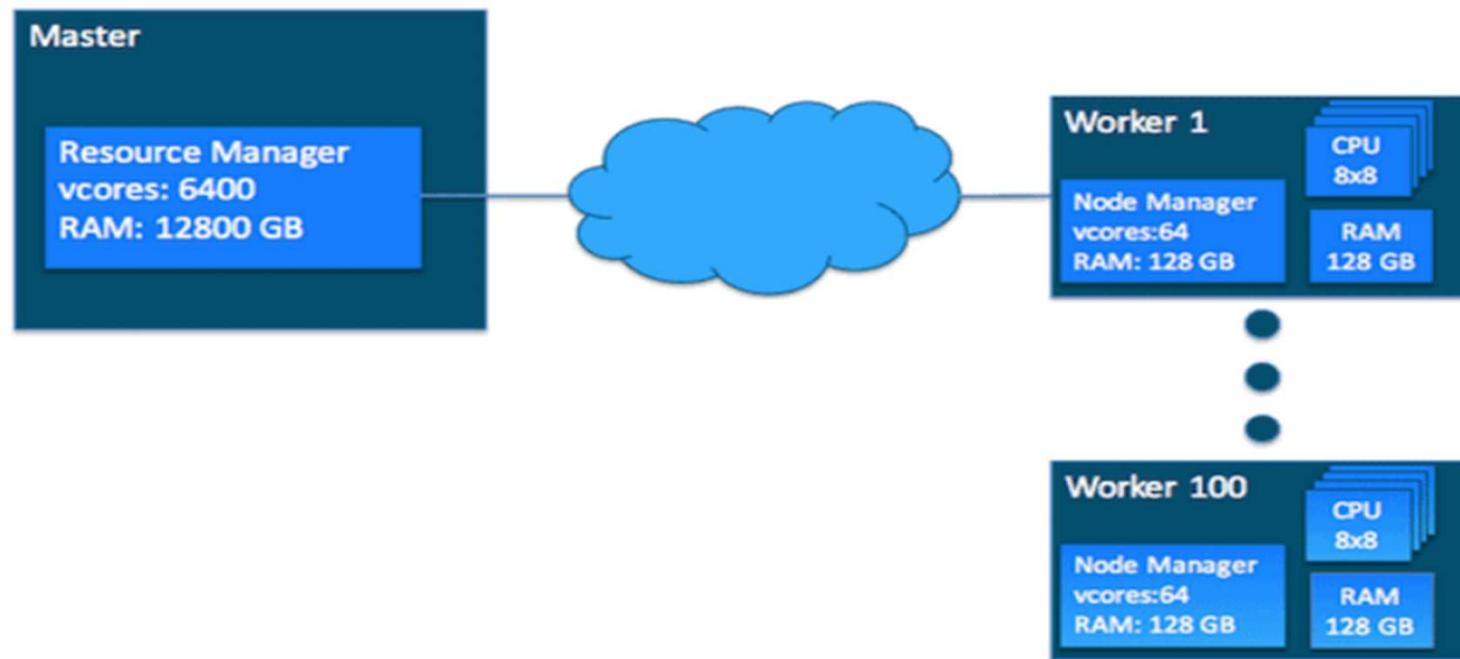


Team work



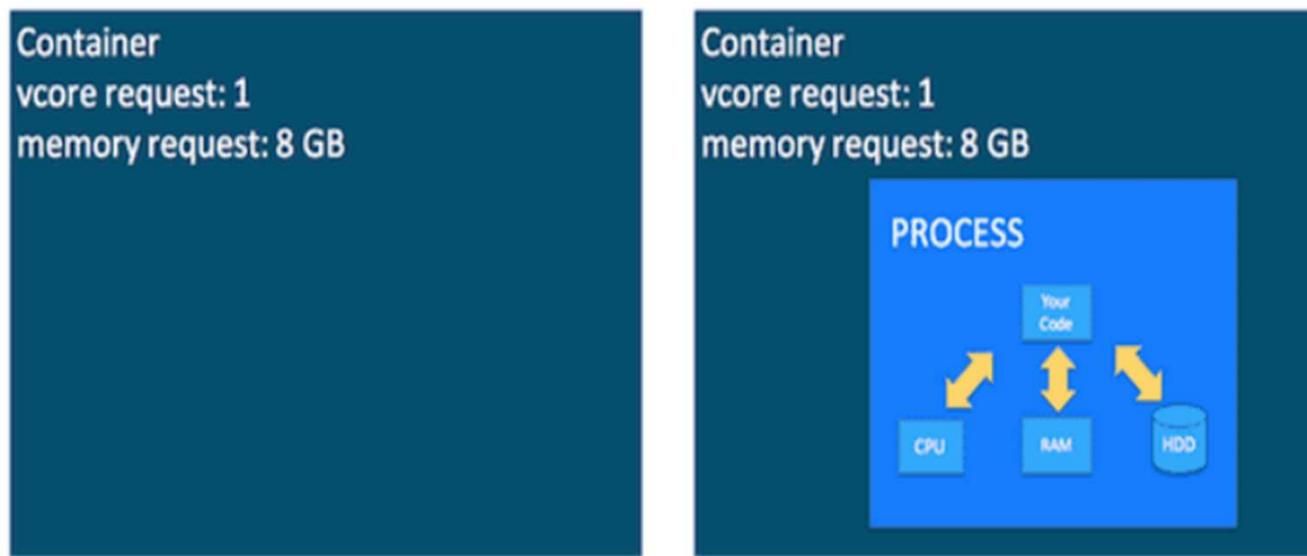
Global View

YARN currently defines two resources, *vcores* and *memory*. Each NodeManager tracks its own local resources and communicates its resource configuration to the ResourceManager, which keeps a running total of the cluster's available resources. By keeping track of the total, the ResourceManager knows how to allocate resources as they are requested. (Vcore has a special meaning in YARN. You can think of it simply as a "usage share of a CPU core." If you expect your tasks to be less CPU-intensive (sometimes called I/O-intensive), you can set the ratio of vcores to physical cores higher than 1 to maximize your use of hardware resources.)



Container

Containers are an important YARN concept. You can think of a container as a request to hold resources on the YARN cluster. Currently, a container hold request consists of vcore and memory, shown in Figure



Container as a hold (left), and container as a running process (right)

Once a hold has been granted on a host, the NodeManager launches a process called a *task*. The right side of Figure shows the task running as a process inside a container.

Running Process/App Master

1. The application starts and talks to the ResourceManager for the cluster:

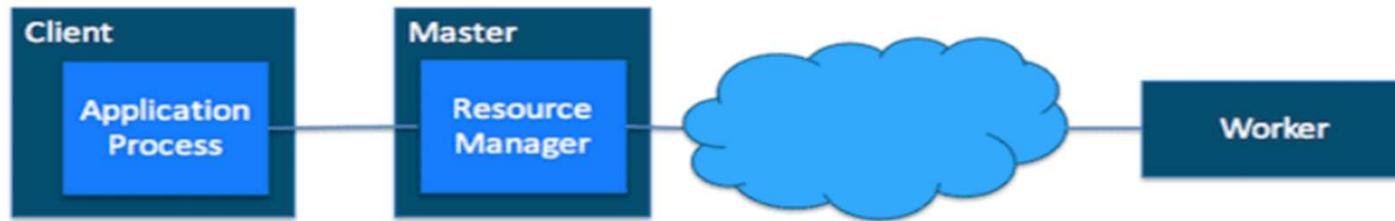


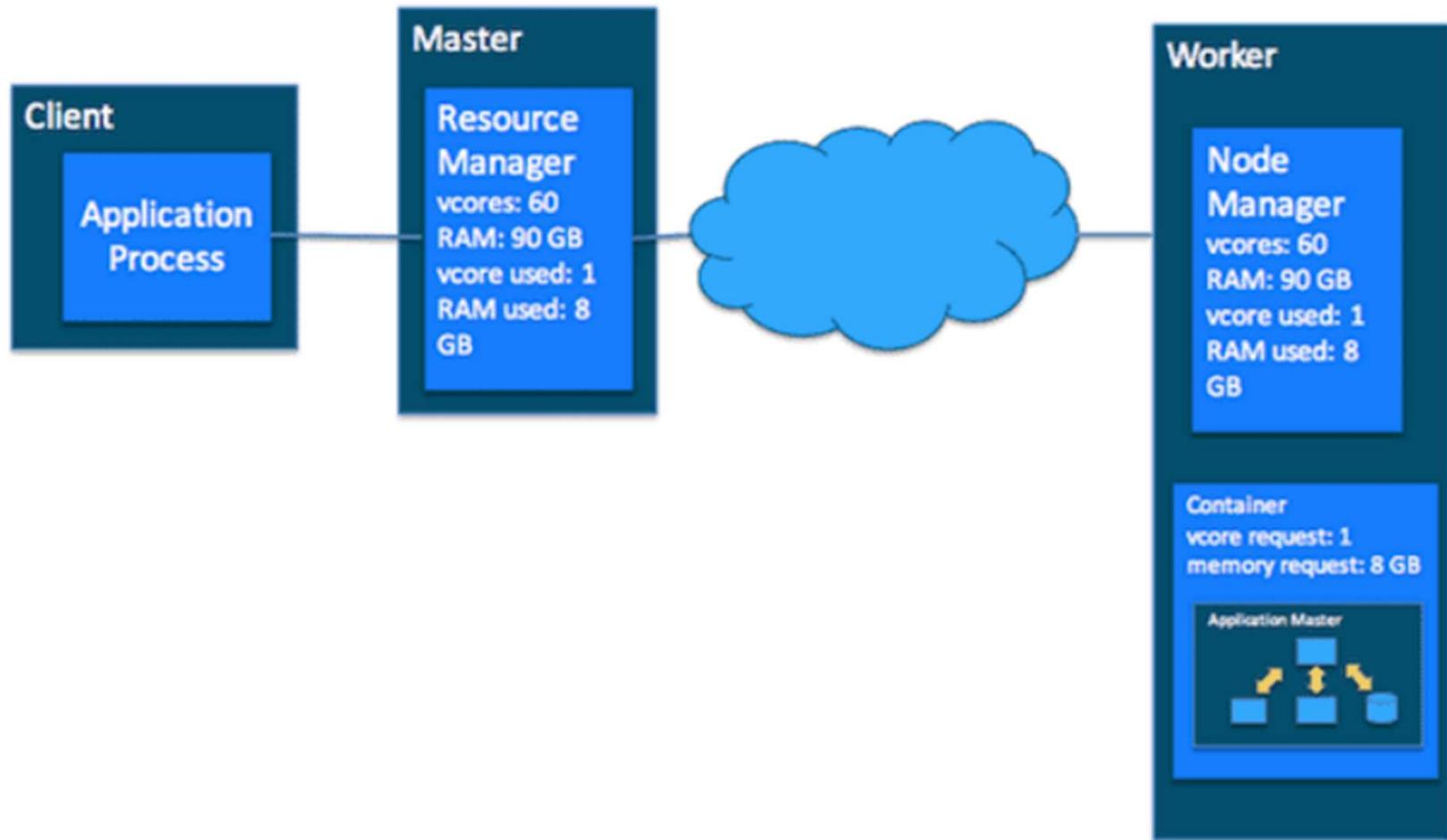
Figure 5: Application starting up before tasks are assigned to the cluster

2. The ResourceManager makes a single container request on behalf of the application:

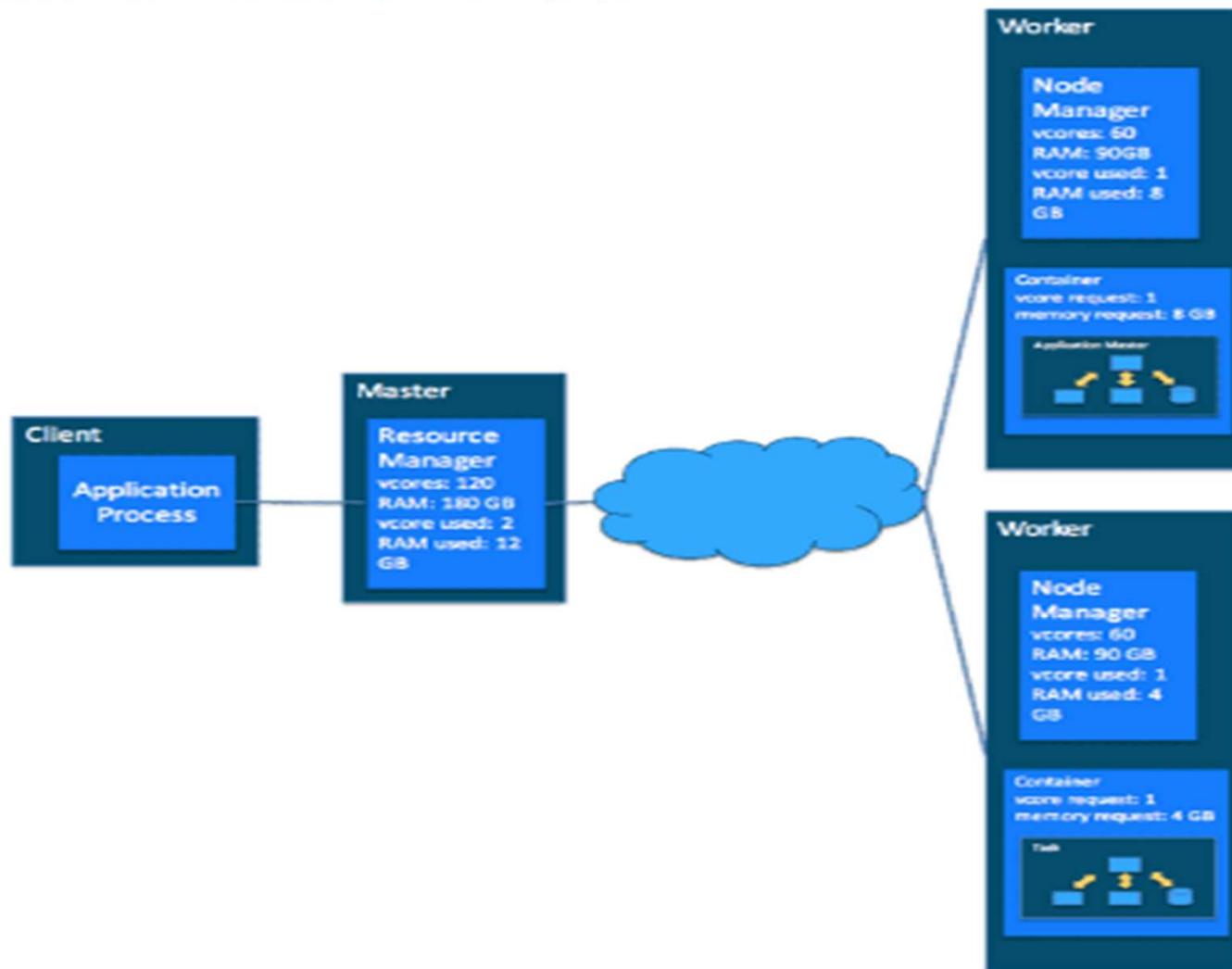


Step 3

3. The ApplicationMaster starts running within that container:



4. The ApplicationMaster requests subsequent containers from the ResourceManager that are allocated to run tasks for the application. Those tasks do most of the status communication with the ApplicationMaster allocated in Step 3):



Putting It Together

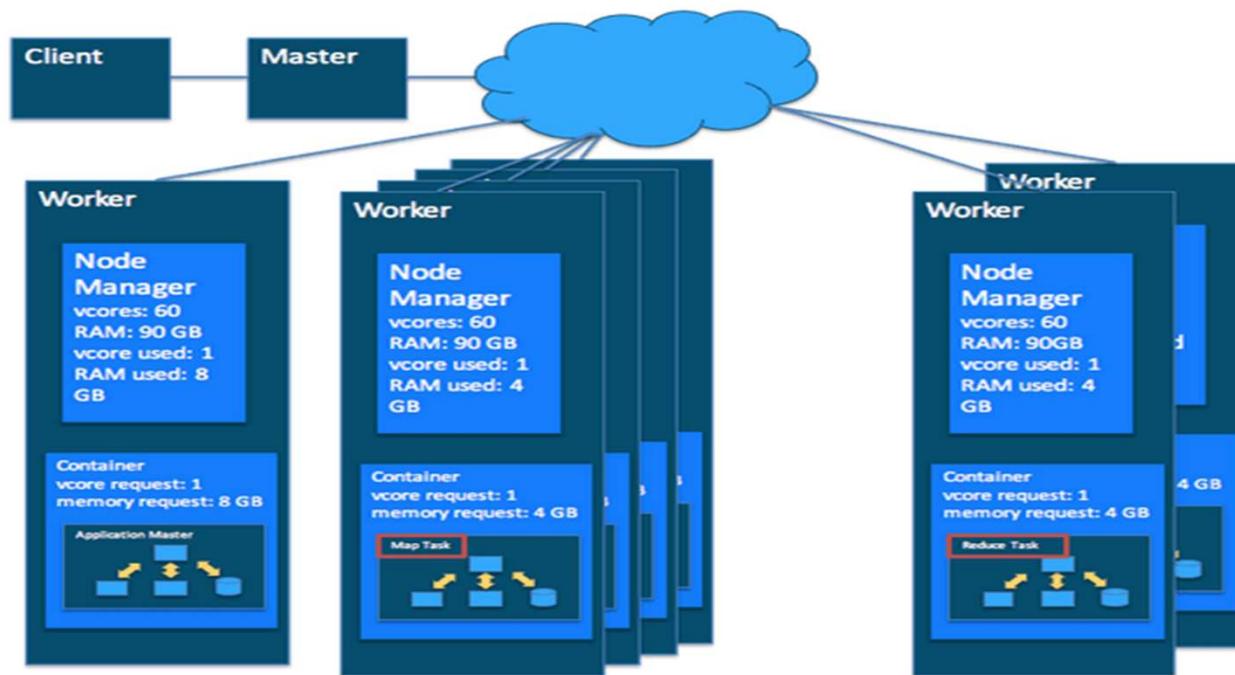


Figure 10: Merged MapReduce/YARN Application Running on a Cluster

In a MapReduce application, there are multiple map tasks, each running in a container on a worker host somewhere in the cluster. Similarly, there are multiple reduce tasks, also each running in a container on a worker host.

Simultaneously on the YARN side, the ResourceManager, NodeManager, and ApplicationMaster work together to manage the cluster's resources and ensure that the tasks, as well as the corresponding application, finish cleanly.

YARN Installation facts

In reality, there are two reasons why the full set of resources on a node cannot be allocated to YARN:

1. Non-Apache Hadoop services are also required to be running on a node (overhead).
2. Other Hadoop-related components require dedicated resources and cannot be shared with YARN (such as when running CDH).

The Intel-backed group develops CDH, a distribution of Hadoop that includes several other open-source projects, such as Impala and Search. It also offers security and integration features. The Impala framework is an interactive SQL query engine that allows direct queries of data stored in HDFS, Apache HBase, or AWS S3.

YARN in RM UI

As mentioned before, the ResourceManager has a snapshot of the available resources on the YARN cluster.

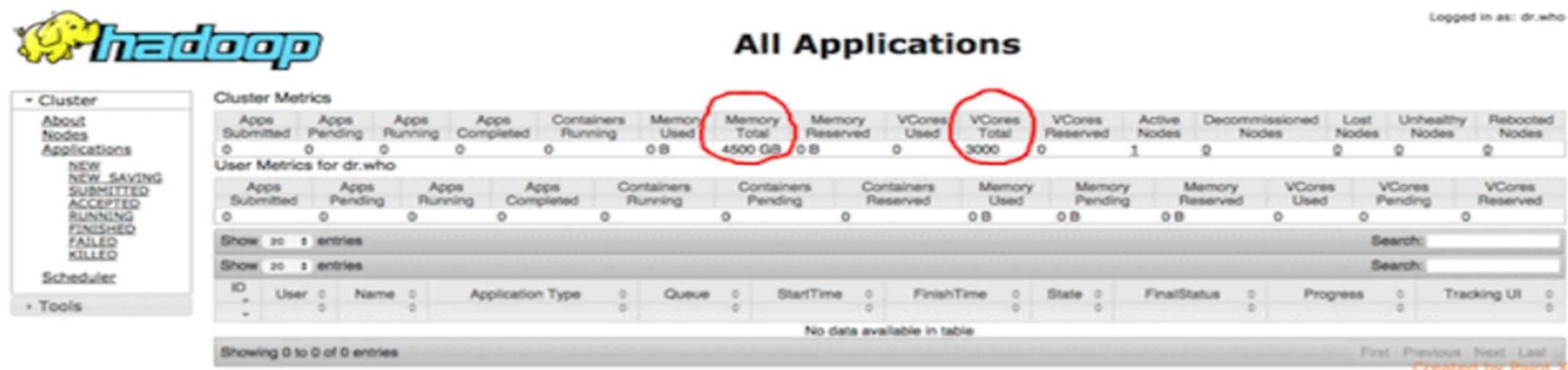
Example: Assume you have the following configuration on your 50 Worker nodes:

1. `yarn.nodemanager.resource.memory-mb = 90000`
2. `yarn.nodemanager.resource.vcores = 60`

Doing the math, your cluster totals should be:

1. memory: $50 * 90\text{GB} = 4500\text{GB} = 4.5\text{TB}$
2. vcores: $50 * 60 \text{ vcores} = 3000 \text{ vcores}$

On the ResourceManager Web UI page, the cluster metrics table shows the total memory and total vcores for the cluster, as seen in Figure 2 below.



The screenshot shows the "All Applications" page of the Hadoop ResourceManager Web UI. The top navigation bar includes the Hadoop logo and the text "Logged in as: dr.who". On the left, there's a sidebar with sections for Cluster (About, Nodes, Applications, Scheduler), Applications (NEW, NEW_SAVING, SUBMITTED, ACCEPTED, RUNNING, FINISHED, FAILED, KILLED), and Tools. The main content area is titled "All Applications" and contains two tables under "Cluster Metrics". The first table shows overall cluster statistics: Apps Submitted (0), Apps Pending (0), Apps Running (0), Apps Completed (0), Containers Running (0), Memory Used (0 B), Memory Total (4500 GB), Memory Reserved (0 B), Vcores Used (0), Vcores Total (3000), Vcores Reserved (0), Active Nodes (1), Decommissioned Nodes (0), Lost Nodes (0), Unhealthy Nodes (0), and Rebooted Nodes (0). The second table provides user-specific metrics for "dr.who": Apps Submitted (0), Apps Pending (0), Apps Running (0), Apps Completed (0), Containers Running (0), Containers Pending (0), Containers Reserved (0), Memory Used (0 B), Memory Pending (0 B), Memory Reserved (0 B), Vcores Used (0), Vcores Pending (0), and Vcores Reserved (0). Below these tables, there are search fields and links for "Show 20 entries" and "Search". A note at the bottom states "No data available in table". At the very bottom, it says "Showing 0 to 0 of 0 entries" and "First Previous Next Last". The footer indicates the page was "Created by Pinal X".

Container Config

At this point, the YARN Cluster is properly set up in terms of Resources. YARN uses these resource limits for allocation, and enforces those limits on the cluster.

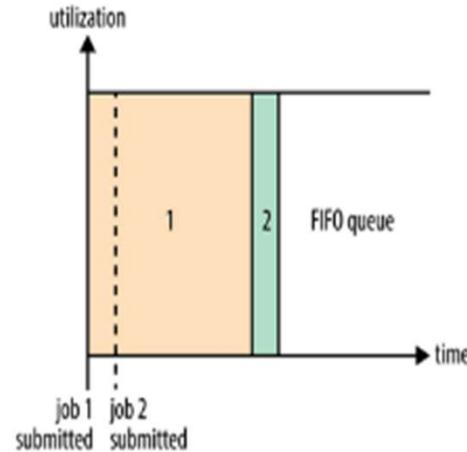
- YARN Container Memory Sizing
 - Minimum: `yarn.scheduler.minimum-allocation-mb`
 - Maximum: `yarn.scheduler.maximum-allocation-mb`
- YARN Container VCore Sizing
 - Minimum: `yarn.scheduler.minimum-allocation-vcores`
 - Maximum: `yarn.scheduler.maximum-allocation-vcores`
- YARN Container Allocation Size Increments
 - Memory Increment: `yarn.scheduler.increment-allocation-mb`
 - VCore Increment: `yarn.scheduler.increment-allocation-vcores`

Restrictions

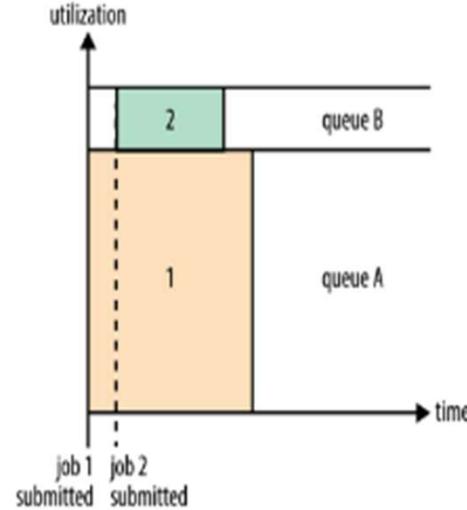
- Memory properties:
 - Minimum required value of 0 for `yarn.scheduler.minimum-allocation-mb`.
 - Any of the memory sizing properties must be less than or equal to `yarn.nodemanager.resource.memory-mb`.
 - Maximum value must be greater than or equal to the minimum value.
- VCore properties:
 - Minimum required value of 0 for `yarn.scheduler.minimum-allocation-vcores`.
 - Any of the vcore sizing properties must be less than or equal to `yarn.nodemanager.resource.vcores`.
 - Maximum value must be greater than or equal to the minimum value.
 - Recommended value of 1 for `yarn.scheduler.increment-allocation-vcores`. Higher values will likely be wasteful.

Scheduling YARN

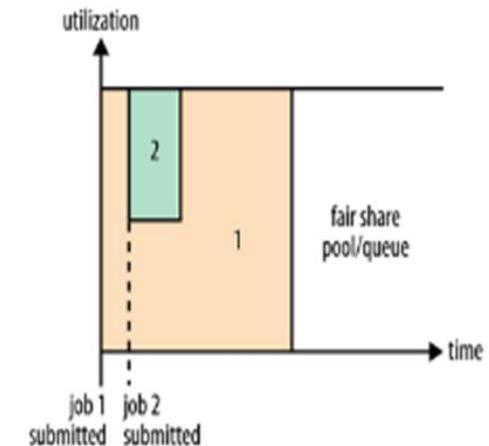
i. FIFO Scheduler



ii. Capacity Scheduler



iii. Fair Scheduler



Jump To Next PPT

Add YARN Service

Home

Status All Health Issues Configuration  All Recent Commands

Cluster 1 (CDH 5.12.1, Parcels)

-  Hosts
-  HDFS
-  CloudBees

Actions

-  Add Service
- Start
- Stop
- Restart
- Rolling Restart
- Deploy Client Configuration

Charts

Cluster CPU



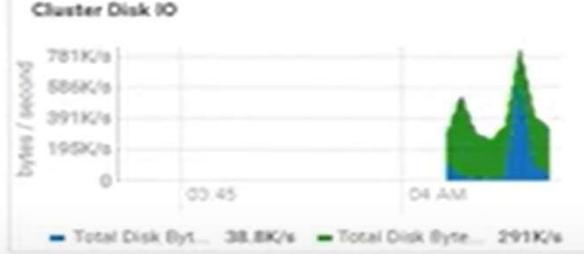
Percent

0%, 50%, 100%

03:45 04 AM

Cluster 1, Host CPU Usage Across Hosts 5.1%

Cluster Disk IO



bytes / second

0, 199K/s, 391K/s, 585K/s, 781K/s

03:45 04 AM

Total Disk Byte... 38.8K/s Total Disk Byte... 291K/s

without directly providing AWS credentials, subject to having the proper permissions defined via Sentry protocol.

 Sentry	Sentry service stores authorization policy metadata and provides clients concurrent and secure access to data.
 Solr	Solr is a distributed service for indexing and searching data stored in HDFS.
 Spark	Apache Spark is an open source cluster computing system. This service runs Spark as an application on the cluster.
 Spark (Standalone)	Apache Spark is an open source cluster computing system. This is the standalone version of the service management. Cloudera recommends using Spark on YARN instead of this standalone version.
 Sqoop 1 Client	Configuration and connector management for Sqoop 1.
 Sqoop 2	Sqoop is a tool designed for efficiently transferring bulk data between Apache Hadoop and structured data stores. The version supported by Cloudera Manager is Sqoop 2.
 YARN (MR2 Included)	Apache Hadoop MapReduce 2.0 (MRv2), or YARN, is a data computation framework that supports MapReduce and streaming processing. It includes support for HDFS and other storage systems.
 ZooKeeper	Apache ZooKeeper is a centralized service for maintaining and synchronizing configuration data.

Add YARN (MR2 Included) Service to Cluster 1

Customize Role Assignments for YARN (MR2 Included)

You can customize the role assignments for your new service here, but note that if assignments are made incorrectly, such as assigning too many roles, it may suffer.

You can also view the role assignments by host.

[View By Host](#)

RM ResourceManager × 1 New

ip-172-31-9-240.ap-southeast-1.compute.internal

JHS JobHistory Server × 1 New

ip-172-31-0-223.ap-southeast-1.compute.internal

NM NodeManager × 3 New

ip-172-31-0-223.ap-southeast-1.compute.internal

Select hosts for a new or existing role. The host list is filtered to remove hosts that are not valid candidates; these include hosts that are unhealthy, members of other clusters, or have incompatible version of CDH installed on them.

Enter hostnames: host01, host[01-10], IP addresses or rack.

[Search](#)

Hostname	IP Address	Rack	Cores	Physical Memory	Existing Roles	Added Role
<input type="checkbox"/> ip-172-31-0-223.ap-southeast-1.compute.internal	172.31.0.223	/default	2	7.7 GiB	B DN AP SM	NM
<input type="checkbox"/> ip-172-31-14-33.ap-southeast-1.compute.internal	172.31.14.33	/default	2	7.7 GiB	DN HM	NM
<input type="checkbox"/> ip-172-31-5-165.ap-southeast-1.compute.internal	172.31.5.165	/default	2	7.7 GiB	DN SNN RM	NM
<input checked="" type="checkbox"/> ip-172-31-9-240.ap-southeast-1.compute.internal	172.31.9.240	/default	2	7.7 GiB	NN ES	RM JHS

Default config

Add YARN (MR2 Included) Service to Cluster 1

Review Changes

Default Configuration

NodeManager Local Directories

yarn.nodemanager.local-dirs

NodeManager Default Group

/yarn/nm

Enable Container Usage Metrics Collection

YARN (MR2 Included) (Service-Wide)

Container Usage MapReduce Job User

YARN (MR2 Included) (Service-Wide)

Cloudera Manager Container Usage Metrics Directory

YARN (MR2 Included) (Service-Wide)

/tmp/cmYarnContainerMetrics

Container Usage Output Directory

YARN (MR2 Included) (Service-Wide)

/tmp/cmYarnContainerMetricsAggregate

Adding Services

Add YARN (MR2 Included) Service to Cluster 1

First Run Command

Status Running Oct 29, 4:13:06 AM Abort **Adding the service...it may take some time**

Completed 3 of 4 step(s). Show All Steps Show Only Failed Steps Show Only Running Steps

> <input checked="" type="radio"/> Ensuring that the expected software releases are installed on hosts. Successfully completed 1 steps.
> <input checked="" type="radio"/> Deploying Client Configuration Cluster 1 Successfully deployed all client configurations.
> <input checked="" type="radio"/> Creating DFS directories required for YARN Successfully completed 2 steps.
> <input type="radio"/> Start YARN (MR2 Included) YARN (MR2 Included)

Contd..

```
[root@ip-172-31-0-223 ~]# service ntpd start
Starting ntpd: [ OK ]
[root@ip-172-31-0-223 ~]# service ntpd status
ntpd (pid 5061) is running...
[root@ip-172-31-0-223 ~]# hdfs dfs -ls /user/
ls: `/user/': No such file or directory
You have new mail in /var/spool/mail/root
[root@ip-172-31-0-223 ~]# hdfs dfs -ls /
Found 1 items
drwxrwxrwt - hdfs supergroup 0 2017-10-29 04:02 /tmp
[root@ip-172-31-0-223 ~]# hdfs dfs -ls /
Found 2 items
drwxrwxrwt - hdfs supergroup 0 2017-10-29 04:13 /tmp
drwxr-xr-x - hdfs supergroup 0 2017-10-29 04:13 /user
```

Directory created by YARN

Service added Successfully

Cluster 1 (CDH 5.12.1, Parcels)

- Hosts: 3
- HDFS: 1
- YARN (MR2 Included)** (YARN SERVICE ADDED)

Cloudera Management Service

- Cloudera M...: 2

Charts

30m 1h 2h 6h

Cluster CPU



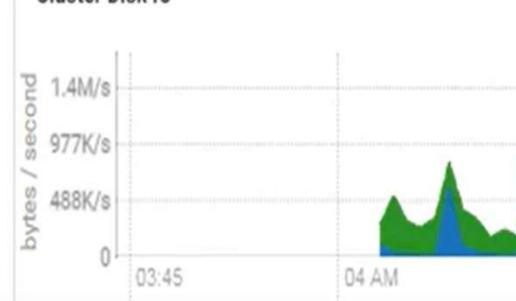
percent

0%, 50%, 100%

03:45 04 AM

Cluster 1, Host CPU Usage Across Hosts 9.9%

Cluster Disk IO



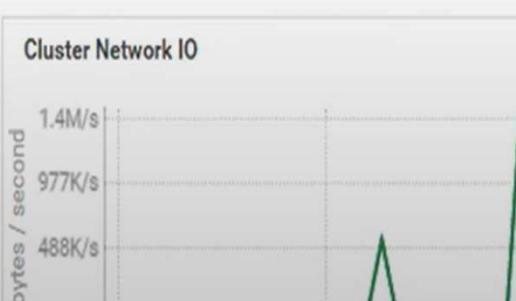
bytes / second

0, 488K/s, 977K/s, 1.4M/s

03:45 04 AM

Total Disk Byte... 1.3M/s Total Disk Byte... 471K/s

Cluster Network IO



bytes / second

0, 488K/s, 977K/s, 1.4M/s

HDFS IO



bytes / second

0, 2b/s, 4b/s

Home Status All Health Issues 04 Configuration 5 All Recent Commands

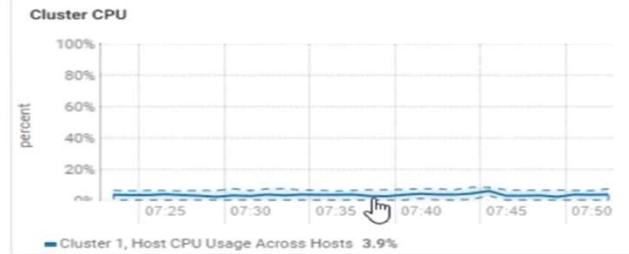
You are running Cloudera Manager in non-production mode, which uses an embedded PostgreSQL database. Switch to using a supported external database before moving into production. [More Details](#)

Cluster 1 (CDH 5.14.0, Parcels)

- 9 Hosts 5
- Flume
- HBase
- HDFS
- Hive
- Hue
- Impala
- Kafka
- Key-Value St...
- Impala
- Kafka
- Key-Value St...
- Oozie
- Solr
- Spark
- Spark 2
- Spoon 2
- YARN (MR2 I...)**
- ZooKeeper

Charts

Cluster CPU



percent

Cluster 1, Host CPU Usage Across Hosts 3.9%

Cluster Disk IO



bytes / second

Total Disk Bytes Read... 0 Total Disk Byte... 1.9M/s

Cluster Network IO



bytes / second

Cluster 1, Host CPU Usage Across Hosts 3.9%

HDFS IO



bytes / second

Total Disk Bytes Read... 0 Total Disk Byte... 1.9M/s

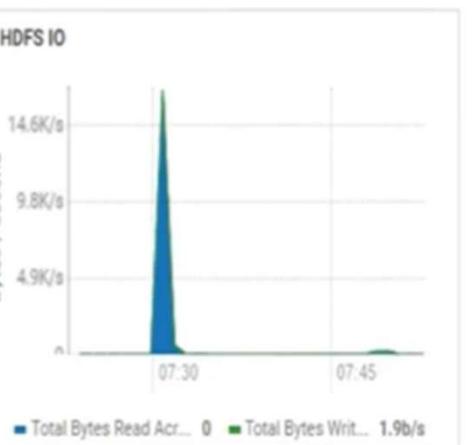
Cluster Network IO



bytes / second

Total Bytes Rec... 1.2M/s Total Bytes Tra... 1.2M/s

HDFS IO



bytes / second

Total Bytes Read Acr... 0 Total Bytes Writ... 1.9b/s

Completed Impala Queries

Ind

physical memory

55.9G

Status page

Status instances Configuration Commands Applications Resource Pools Charts Library Audits Web UI ▾ Quick Links ▾

Health Tests [Create Trigger](#)

 Show 3 **good**

 YARN Container Usage Aggregation [Suppress...](#)

This health test is disabled because container usage metric collection is disabled for YARN.

Status Summary

JobHistory Server	 1 Good Health
NodeManager	 3 Good Health
ResourceManager	 1 Good Health
Hosts	 4 Good Health

Charts

Applications Running (Cumulative)

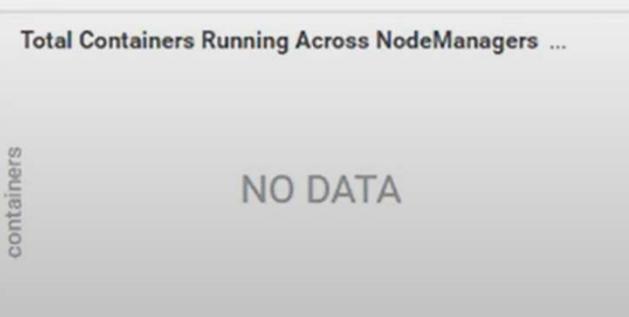


applications

03:45 04 AM

root (YARN (MR2 Included)), Applications Running (... 0

Total Containers Running Across NodeManagers ...



containers

NO DATA

Instances/Roles

 YARN (MR2 Included) (Cluster 1) Actions ▾ Apr 14, 7:53 PM UTC

Status Instances Configuration Commands Applications Resource Pools Charts Library Web UI ▾ Quick Links ▾

Search 

Role Type	State	Host	Commission State	Role Group
JobHistory Server	Started	ip-10-0-1-20.ec2.internal	Commissioned	JobHistory Server Default Group
NodeManager	Started	ip-10-0-1-10.ec2.internal	Commissioned	NodeManager Default Group
NodeManager	Started	ip-10-0-2-12.ec2.internal	Commissioned	NodeManager Group 1
NodeManager	Stopped	ip-10-0-2-15.ec2.internal	Decommissioned	NodeManager Default Group
NodeManager	Started	ip-10-0-2-11.ec2.internal	Commissioned	NodeManager Group 1
NodeManager	Stopped	ip-10-0-2-14.ec2.internal	Decommissioned	NodeManager Group 2
NodeManager	Started	ip-10-0-2-13.ec2.internal	Commissioned	NodeManager Default Group
ResourceManager (Active)	Started	ip-10-0-1-20.ec2.internal	Commissioned	ResourceManager Default Group

Filters

- STATUS**
 - Stopped 2
 - Good Health 6
- COMMISSION STATE**
- MAINTENANCE MODE**
- RACK**
- ROLE GROUP**
- ROLE TYPE**
- STATE**

Feedback

Configuration

 YARN (MR2 Included) (Cluster 1) Actions ▾ Apr 14, 7:53 PM UTC

Status Instances Configuration Commands Applications Resource Pools Charts Library Web UI ▾ Quick Links ▾

Search I Switch to the classic layout Role Groups

Filters Read-only: Requires additional authorization Show All Descriptions

SCOPE

Service	Scope	Description
HDFS Service	YARN (MR2 Included) (Service-Wide)	hdfs
ZooKeeper Service	YARN (MR2 Included) (Service-Wide)	zookeeper

CATEGORY

Category	Setting	Description
Advanced	yarn.acl.enable	Enable ResourceManager ACLs
Compression		
Logs		
Main		
Monitoring		

Admin API VADM / YARN (MR2 Included) / Overview Metrics

Resource Manager UI



Logged in as: dr

All Applications

Cluster	
About	
Nodes	
Applications	
NEW	
NEW SAVING	
SUBMITTED	
ACCEPTED	
RUNNING	
FINISHED	
FAILED	
KILLED	
Scheduler	
Tools	

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores Total	Vcores Reserved
0	0	0	0	0	0 B	3 GB	0 B	0	6	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
3	0	0	0	0	0

User Metrics for dr.who

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	Memory Pending	Memory Reserved	Vcores Used	Vcores Pending	Vcore Reserved
0	0	0	0	0	0	0	0 B	0 B	0 B	0	0	0

Show 20 ▾ entries

Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores	Allocated Memory MB	Progress	Tracking UI

Job History UI



The screenshot shows the Hadoop JobHistory user interface. At the top left is the Hadoop logo. On the right, there is a status bar with "Logged in as" followed by a placeholder. Below the logo, the title "JobHistory" is displayed.

The main area features a table titled "Retired Jobs". The table has a header row with columns: Submit Time, Start Time, Finish Time, Job ID, Name, User, Queue, State, Maps Total, Maps Completed, Reduces Total, and Reduces Completed. A dropdown menu above the table allows selecting the number of entries to show (20). A search input field is also present.

Below the table, a message says "No data available in table". The bottom part of the table structure shows a repeating header row and a message "Showing 0 to 0 of 0 entries" along with navigation links for First, Previous, and Next.

YARN Tuning



Why Cloudera Products Services & Support



This is the documentation for Cloudera Enterprise 54.x. Documentation for other versions is available at [Cloudera Docs](#)

Documentation > [Cloudera Installation and Upgrade](#) > [Installing Cloudera Manager and CDH](#) > [Installing and Deploying CDH Using the Command Line](#)

[View All Categories](#)

- ▶ Cloudera Introduction
 - Cloudera Release Notes
- ▶ Cloudera QuickStart
- ▼ Cloudera Installation and Upgrade
 - ▶ Installation Requirements for Cloudera Manager, Cloudera Navigator, and CDH 5
- ▼ Installing Cloudera Manager and CDH
 - Java Development Kit Installation
 - ▶ Installing Cloudera Manager, CDH,

Tuning YARN

This topic applies to YARN clusters only, and describes how to tune and optimize YARN for yo

Note: Download the Cloudera [YARN tuning spreadsheet](#) to help calculate YARN configura
short video overview, see [Tuning YARN Applications](#).

Overview

This overview provides an abstract description of a YARN cluster and the goals of YARN tunir

Machine Configuration

STEP 1: Worker Host Configuration

Enter your likely machine configuration in the input boxes below. If you are uncertain what machines you plan on buying, put in some minimum values that will suit what you expect to buy.

Host Components	Quantity	Description
RAM	8	Gigabytes
CPU	2	3 CPUs: 6 cores, 3.5 GHz, 15MB cache
HDD (Hard Disk Drive)	36	.2x3TB SATA III Hard Drives in JBOD Configuration
Ethernet	2	. Gigabit Ethernet

STEP 2: Worker Host Planning



to allocate resources, mainly CPU and memory, to the various software components that run on the host.

	Service	Category	CPU (cores)	Memory (M)	Notes
16	Operating System	Overhead	1	1024	Most operating systems use 4-8GB minimum.
17	Task overhead	Overhead	0	0	Allow additional memory overhead for task buffers such as
18	Cloudera Manager agent	Overhead	0	1024	Allocate 1GB for Cloudera Manager agents, which track res
19	Other services	Overhead	0	0	Enter the required cores or memory for services not listed a
20	HDFS DataNode	CDH	0	1024	Allocate 1GB for the HDFS DataNode.
21	Impala daemon	CDH	0	0	(Optional Service) Suggestion: Allocate at least 16GB mem
22	Hbase RegionServer	CDH	0	0	(Optional Service) Suggestion: Allocate no more than 12-16
23	Solr Server	CDH	0	0	(Optional Service) Suggestion: Minimum 1GB for Solr server
24	YARN NodeManager	CDH	0	1024	Allocate 1GB for the YARN NodeManager.
25	Available Resources		1	4096	
26	Physical Cores to Vcores Multiplier		4		Set this ratio based on the expected number of concurrent t
27	YARN Available Vcores		4		This value will be used in STEP 4 for YARN Configuration
28	YARN Available Memory			4096	This value will be used in STEP 4 for YARN Configuration
29					
30					
31					
32					
33					
34	STEP 3: Cluster Size				
	Cluster Configuration	YARN Configuration	MapReduce Configuration		

Details of worker nodes

Apache RegionServer	CDH	0	0 (Optional Service) Suggestion:
HDFS Server	CDH	0	0 (Optional Service) Suggestion:
YARN NodeManager	CDH	0	1024 Allocate 1GB for the YARN Node Manager
Available Resources		1	4096
Physical Cores to Vcores Multiplier		4	Set this ratio based on the experience
YARN Available Vcores		4	This value will be used in STEP 2
YARN Available Memory		4096	This value will be used in STEP 3

STEP 3: Cluster Size

Enter the number of nodes you have (or expect to have) in the cluster

		Quantity			
Number of Worker Hosts in the cluster		3			

Make note of node statistics

cloudera MANAGER

- Clusters ▾
- Hosts ▾**
- Diagnostics ▾
- Audits
- Charts ▾
- Backup ▾
- Administration ▾

Search
Support ▾
admin ▾

All Hosts

Configuration
Add New Hosts to Cluster
Re-run Upgrade Wizard
Inspect All Hosts

Filters

Actions for Selected ▾
Columns: 11 Selected ▾

	IP	Roles	Commission State	Last Heartbeat	Load Average	Cores	Disk Usage	Physical Memory	Swap Space
0-	172.31.0.223	5 Role(s)	Commissioned	9.62s ago	0.06 0.11 0.10	2	9 GiB / 47.4 GiB	1.8 GiB / 7.7 GiB	0 B / 816 MiB
14-	172.31.14.33	3 Role(s)	Commissioned	9.75s ago	0.00 0.01 0.07	2	8.9 GiB / 47.4 GiB	1.5 GiB / 7.7 GiB	0 B / 816 MiB
5-	172.31.5.165	4 Role(s)	Commissioned	9.51s ago	0.21 0.07 0.09	2	9.4 GiB / 47.4 GiB	1.8 GiB / 7.7 GiB	0 B / 816 MiB
9-	172.31.9.240	4 Role(s)	Commissioned	9.18s ago	0.06 0.04 0.01	2	11 GiB / 47.4 GiB	3.9 GiB / 7.7 GiB	0 B / 816 MiB



YARN Configuration

STEP 4: YARN Configuration on Cluster

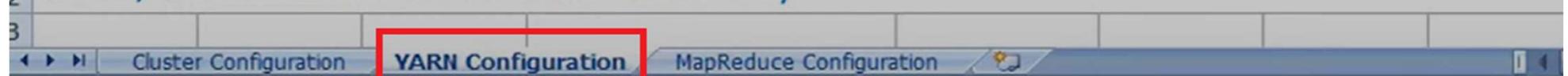
These are the first set of configuration values for your cluster. You can set these values in YARN > Configuration

Go To These Locations and Get the Values

YARN Configuration Property	Value
yarn.nodemanager.resource.cpu-vcores	4 Copied from STEP 2 "Av
yarn.nodemanager.resource.memory-mb	4096 Copied from STEP 2 "Av

STEP 5: Verify YARN Settings on Cluster

Go to the Resource Manager Web UI (usually <http://<ResourceManagerIP>:8088/>) and verify that "Memory Total" and "Vcores Total" matches the values above. If your machine has no bad nodes, then the numbers should match exactly.



Memory Available update

cloudera MANAGER Clusters ▾ Hosts ▾ Diagnostics ▾ Audits Charts ▾ Backup ▾ Administration ▾

YARN (MR2 Included) (Cluster 1) Actions ▾

Status Instances Configuration Commands Applications Resource Pools Charts Library Audits Web UI ▾ Quick Links ▾

Switch to 1

Filters

SCOPE

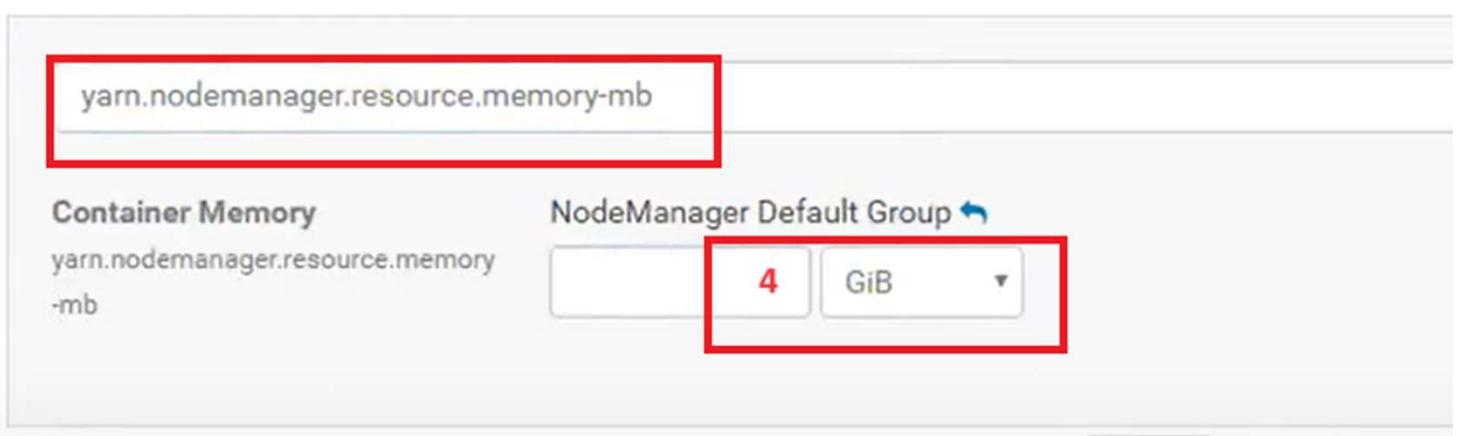
YARN (MR2 Included) (Service-Wide)	0
Gateway	0
JobHistory Server	0
NodeManager	1
ResourceManager	0

yarn.nodemanager.resource.memory-mb

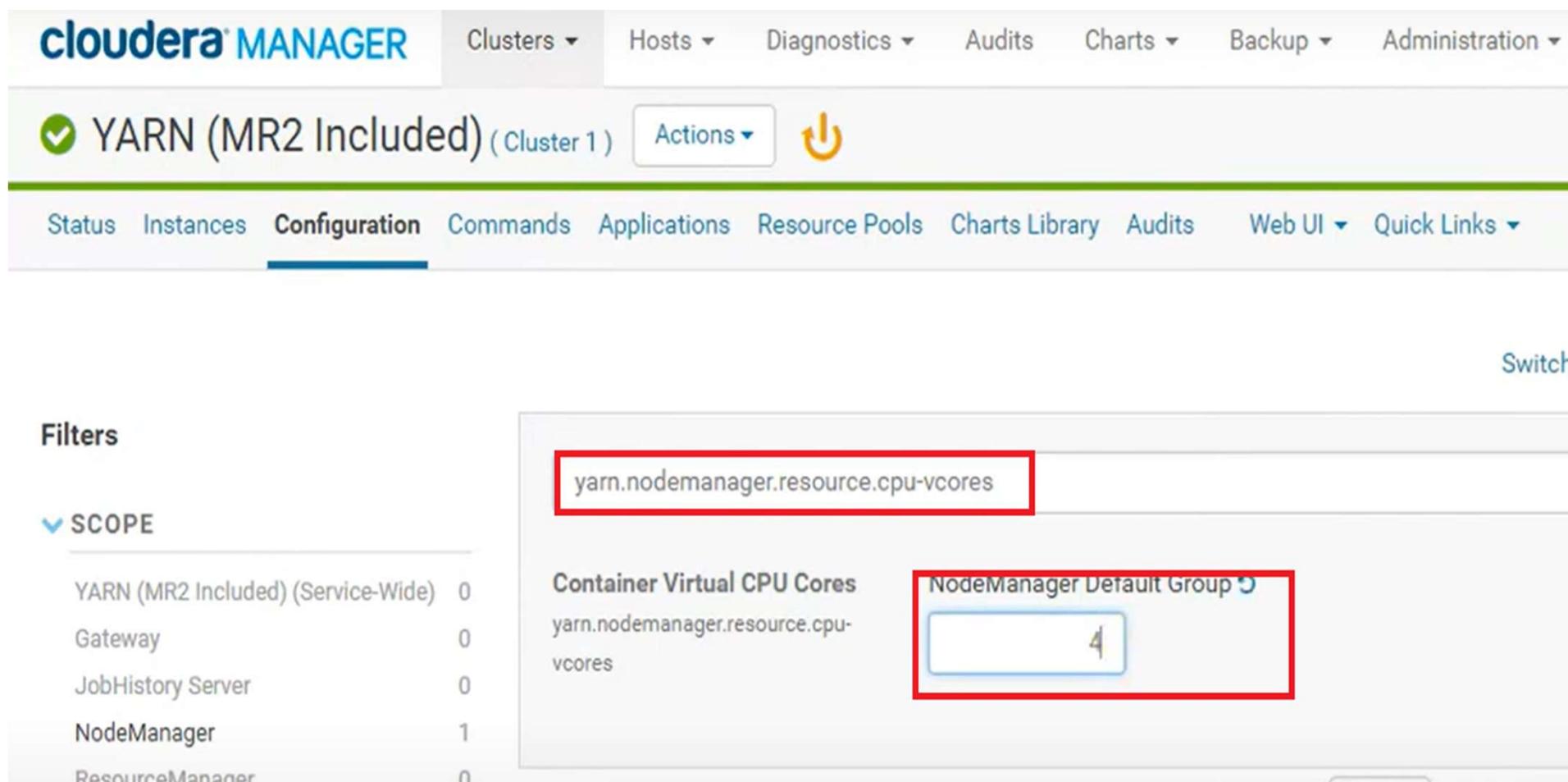
Container Memory NodeManager Default Group ↵

yarn.nodemanager.resource.memory-mb

4 GiB ▾



CPU Cores Available Update



cloudera MANAGER

Clusters ▾ Hosts ▾ Diagnostics ▾ Audits Charts ▾ Backup ▾ Administration ▾

YARN (MR2 Included) (Cluster 1) Actions ▾ 

Status Instances Configuration Commands Applications Resource Pools Charts Library Audits Web UI ▾ Quick Links ▾

Switch

Filters

SCOPE

Scope	Value
YARN (MR2 Included) (Service-Wide)	0
Gateway	0
JobHistory Server	0
NodeManager	1
ResourceManager	0

yarn.nodemanager.resource.cpu-vcores

Container Virtual CPU Cores

yarn.nodemanager.resource.cpu-vcores

NodeManager Default Group

4

Few More Details

STEP 5: Verify YARN Settings on Cluster

Go to the Resource Manager Web UI (usually <http://<ResourceManagerIP>:8088/>) and verify the "Memory Total" and "Vcores Total" matches the values above. If your machine has no bad nodes, then the numbers should match exactly.

Resource Manager Property to Check	Value	Note
Expected Value for "Vcores Total"	12	Calculated from STEP 2 "YARN Available Vcores"
Expected Value for "Memory Total" (in GB)	12	Calculated from STEP 2 "YARN Available Memory"

STEP 6: Verify Container Settings on Cluster

In order to have YARN jobs run cleanly, you need to configure the container properties.

YARN Container Configuration Property (Vcores)	Value	Description
yarn.scheduler.minimum-allocation-vcores	1	Minimum vcore reservation for a container
yarn-scheduler.maximum-allocation-vcores	1	Maximum vcore reservation for a container
yarn.scheduler.increment-allocation-vcores	1	Vcore allocations must be a multiple of this value
YARN Container Configuration Property (Memory)	Value	

Scheduler Info

STEP 6: Verify Container Settings on Cluster

In order to have YARN jobs run cleanly, you need to configure the container properties.

YARN Container Configuration Property (Vcores)	Value	Description
yarn.scheduler.minimum-allocation-vcores	1	Minimum vcore reservation for a container
yarn.scheduler.maximum-allocation-vcores	1	Maximum vcore reservation for a container
yarn.scheduler.increment-allocation-vcores	1	Vcore allocations must be a multiple of
YARN Container Configuration Property (Memory)	Value	
yarn.scheduler.minimum-allocation-mb	1024	Minimum memory reservation for a container
yarn.scheduler.maximum-allocation-mb	8192	Maximum memory reservation for a container
yarn.scheduler.increment-allocation-mb	512	Memory allocations must be a multiple of

Check

41 STEP 6B: Container Sanity Checking

42 This section will do some basic checking of your container parameters in STEP 6 against the hosts.

43

44 Sanity Check

45 Vcore Max >= Vcore Min

46 Memory Max >= Memory Min

47 VCoreMin >= 0

48 VCoreMin <= HostsVCores

49 VCoreMax >= 1

50 VCoreMax <= HostsVcores

51 Memory Min < 1024 MB

52 Memory Max <= HostsMemory

53

54

Means GOOD TO GO

Check Status	Description
GOOD	yarn.scheduler.maximum-allocation-vcores
GOOD	yarn.scheduler.maximum-allocation-mb
GOOD	yarn.scheduler.minimum-allocation-vcores
GOOD	yarn.scheduler.minimum-allocation-mb
GOOD	yarn.scheduler.maximum-allocation-vcores
GOOD	yarn.scheduler.maximum-allocation-mb
GOOD	yarn.scheduler.minimum-allocation-vcores
GOOD	yarn.scheduler.maximum-allocation-mb

Map-Reduce Config

MapReduce Configuration

STEP 7: MapReduce Configuration

Property	Property Type	Component	Value	Description
yarn.app.mapreduce.am.resource.cpu-vcores	Config	Application Master	1	AM cores
yarn.app.mapreduce.am.resource.mb	Config	Application Master	1024	AM memory
ApplicationMaster Java Maximum Heap Size (available in CM)	Java VM Heap	Application Master	1024	AM Java heap
mapreduce.map.cpu.vcores	Config	Map Task	1	Map cores
mapreduce.map.memory.mb	Config	Map Task	1024	Map memory
mapreduce.map.java.opts.max.heap	Java VM Heap	Map Task	1024	Map Java heap
mapreduce.reduce.cpu.vcores	Config	Reduce Task	1	Reduce cores
mapreduce.reduce.memory.mb	Config	Reduce Task	1024	Reduce memory
mapreduce.reduce.java.opts	Java VM Heap	Reduce Task	1024	Reduce Java heap
mapreduce.task.io.sort.mb	Config	Spill/Sort (Map Task)	256	Spill/Sort memory

STEP 7A: MapReduce Sanity Checking

Sanity check MapReduce settings against container minimum/maximum properties.

Application Master Sanity Checks				Value	Description		
yarn.app.mapreduce.am.resource.cpu-vcores >= container min				GOOD	Make sure ApplicationMaster vcore request fits within container limits		
yarn.app.mapreduce.am.resource.cpu-vcores <= container max				GOOD	Ditto		
yarn.app.mapreduce.am.resource.mb >= container min				GOOD	Make sure ApplicationMaster memory request fits within container lim		
yarn.app.mapreduce.am.resource.mb <= container max				GOOD	Ditto		
ApplicationMaster Java Heap "close" to memory request				GOOD	Make sure ApplicationMaster Java Heap is within 90% to 100% of yarn.		
Map Task Sanity Checks				Value	Description		
mapreduce.map.cpu.vcores >= container min				GOOD	Make sure Map Task vcore request fits within container limits		
mapreduce.map.cpu.vcores <= container max				GOOD	Ditto		
mapreduce.map.cpu.memory.mb >= container min				GOOD	Make sure Map Task memory request fits within container limits		
mapreduce.map.cpu.memory.mb <= container max				GOOD	Ditto		
Map Task Java Heap "close" to memory request				GOOD	Make sure that Map Task Java Heap is within 90% to 100% of mapredu		
mapreduce.task.io.sort.mb << Map Task Java Heap				GOOD	Make sure that Spill/Sort memory reservation leaves enough "room" in		
Reduce Task Sanity Checks				Value	Description		
mapreduce.reduce.cpu.vcores >= container min				GOOD	Make sure Reduce Task vcore request fits within container limits		
mapreduce.reduce.cpu.vcores <= container max				GOOD	Ditto		
mapreduce.reduce.cpu.memory.mb >= container min				GOOD	Make sure Reduce Task memory request fits within container limits		
mapreduce.reduce.cpu.memory.mb <= container max				GOOD	Ditto		
Reduce Task Java Heap "close" to memory request				GOOD	Make sure that Reduce Task Java Heap is within 90% to 100% of mapre		

Restart Services

Cloudera MANAGER

Clusters ▾ Hosts ▾ Diagnostics ▾ Audits Charts ▾ Backup ▾ A

Home

Status All Health Issues Configuration ⚡ 6 All Recent Commands

Cluster 1 (CDH 5.12.1, Parcels)

Hosts 3

HDFS

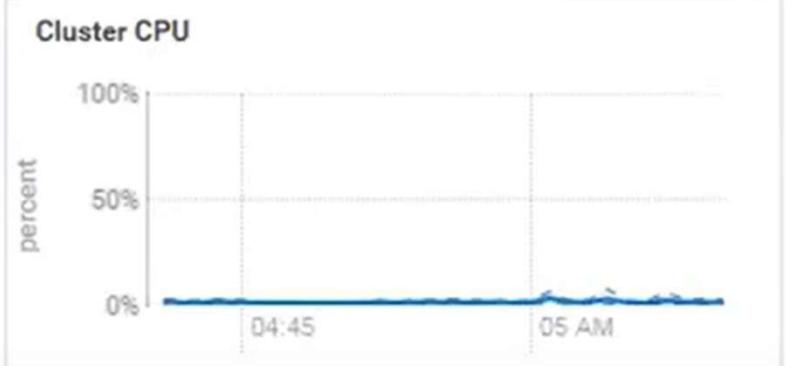
YARN (MR2...)

Restart Services

Stale Configuration: Client configuration redeployment needed

Charts

Cluster CPU



Time	Usage (%)
04:45	~5%
05 AM	~5%

Verify

Logged in as: dr.



All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Vcores Used	Vcores Total	Vcores Reserved
0	0	0	0	0	0 B	12 GB	0 B	0	12	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
3	0	0	0	0	0

User Metrics for dr.who

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Containers Pending	Containers Reserved	Memory Used	Memory Pending	Memory Reserved	Vcores Used	Vcores Pending	Vcores Reserved
0	0	0	0	0	0	0	0 B	0 B	0 B	0	0	0

Show 20 ▾ entries

Search:

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU Vcores	Allocated Memory MB	Progress	Tracking UI
----	------	------	------------------	-------	-----------	------------	-------	-------------	--------------------	----------------------	---------------------	----------	-------------