# IDHAYA COLLEGE FOR WOMEN

# KUMBAKONAM – 612 001

## PG & RESEARCH

## DEPARTMENT OF COMPUTER SCIENCE

| | | |
|---|---|---|
| ACADEMIC YEAR | : | 2019 – 2020 |
| SEMESTER | : | IV |
| CLASS | : | II – M.Sc (CS) |
| SUBJECT IN-CHARGE | : | A. FAIROSEBANU |
| SUBJECT NAME | : | BIG DATA ANALYTICS |
| SUBJECT CODE | : | P16CSE5A |

# UNIT – V

## HADOOP MAPREDUCE & YARN FRAMEWORK

**Introduction to MapReduce, Processing with Hadoop using MapReduce, Introduction to YARN, Components, Need and Challenges of YARN, Dissecting YARN, MapReduce Application, Data serialization and Working with common serialization formats, Big Data serialization formats**

# UNIT – V

# HADOOP MAPREDUCE & YARN FRAMEWORK

## MAPREDUCE

- MapReduce is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster.
- A MapReduce program is composed of a map procedure, which performs filtering and sorting, which performs a summary operation.
- Mapping means grouping the data.
- Reducing means minimize the data.

## PHASES OF MAPREDUCE

- In MapReduce programming, jobs (applications) are split into a set of map tasks and reduce tasks.
- These tasks are executed in a distributed fashion on Hadoop Cluster.

## Map Tasks

- Map Task takes care of loading, parsing, transforming and filtering.
- Each map task is broken into the following phases:
    1. Record Reader
    2. Mapper
    3. Combiner
    4. Partitioner

## 1. Record Reader

- Record Reader converts byte – oriented task into record – oriented task.

**2. Mapper**

- A Mapper converts normal key – value into set of intermediate key – value.

**3. Combiner**

- To combining the two tasks.
- It is an optional function, but provides higher performance.

**4. Partitioner**

- To partition the user – specific code on data in the local disk.
- To separate the storage block.

**Reduce Tasks**

- The responsibility of reduce task is grouping and aggregating data, that is produced by map tasks to generate final output.
- The reduce tasks are broken into the following phases:

1. Shuffle
2. Sort
3. Reducer
4. Output Format

**1. Shuffle**

- To shuffle the data.

**2. Sort**

- To sort the data according to the conditions.

### 3. Reducer

- ❖ It provides various operations such as aggregation, filtering and combining.

### 4. Output Format

- ❖ To produce the output by using the record writer.

## PROCESSING DATA WITH HADOOP USING MAPREDUCE

- ❖ To process data by using MapReduce in Hadoop.
- ❖ It involves the following terms:
    1. MapReduce Daemons
    2. MapReduce Workflow
    3. MapReduce Example

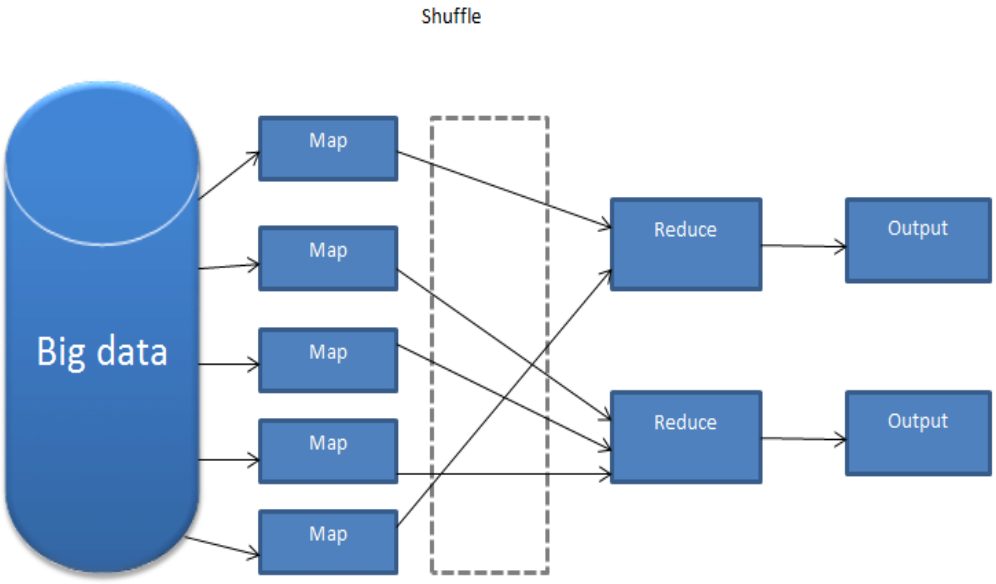
### 1. MapReduce Daemons

- ❖ It describes the components of the MapReduce Architecture
- ❖ Components are the following:
    - Job Tracker: It is a master daemon responsible for executing overall MapReduce job. It provides connectivity between Hadoop and user's application.
    - Task Tracker: There is a single task tracker per slave and spawns multiple Java Virtual Machince (JVM) to handle multiple maps in parallel. It continuously sends heartbeat message to job tracker.

### 2. MapReduce Works Flow

- ❖ MapReduce divides a data analysis task into two parts: Map and Reduce
- ❖ There are two mappers and one reducer.

- Each mapper works on the partial dataset that is stored on that node and the reducer combines the output from the mapper to produce the reduced result set.
- To describe the working model of mapreduce programming.
- The following steps describe how MapReduce performs its task.
    1. First, the input dataset is split into multiple pieces of data.
    2. Next, the framework creates a master and several workers processes and executes the worker processes remotely.
    3. Several map tasks work simultaneously and read pieces of data that were uses the map function to extract only those data that are present on their server and generates key – value pair for the extracted data.
    4. Map worker uses partitioner function to divide the data into regions. Partitioner decides which reducer should get the output of the specified mapper.
    5. When the map workers complete their work, the master instructs the reduce workers to begin their work. The reduce workers in turn contact the map workers to get the key – value data for their partition. The data thus received is shuffled and sorted as per keys.
    6. Then it calls reduce function for every unique key. This function writes the output to the file.
    7. When all the reduce workers complete their work, the master transfers the control to the user program.




**MapReduce Work Flow**

## 3. MapReduce Example

- The famous example for MapReduce programming is word count.
- To count the occurrences of similar words across 50 files.
- Word count MapReduce programming by using java.
- The MapReduce requires three things:
    1. Driver Class: This class specifies job configuration details.
    2. Mapper Class: This class overrides the map function based on the problem statement.
    3. Reduce Class: This class overrides the reduce function based on the problem statement.

## APPLICATIONS OF MAPREDUCE

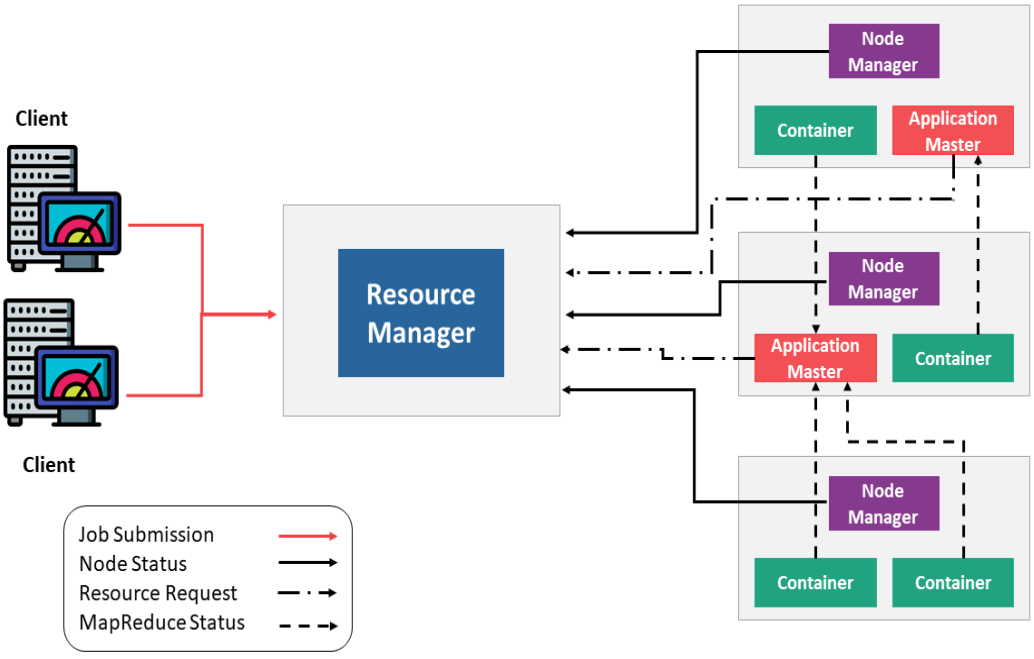
- MapReduce applications are:
  - Distributed Grep
  - Word Count
  - Tera Sort
  - Inverted & Ranked Inverted Index
  - Team Vector
  - Random Forest
  - Spark
  - Extreme Learning Machine
  - DNA Fragment
  - Mobile Sensor Data
  - Social Networks
  - Algorithms

## YARN

- Expansion of YARN is **Y**et **A**nother **R**esource **N**egotiator.
- YARN is a cluster resource management system for Hadoop.
- It acts as a resource manager by using node manager.
- It is used to managing and monitoring workloads.
- It is an operating system for Hadoop.
- It handles the following resources:
    - Memory
    - Devices
    - Process
    - File

## YARN ARCHITECTURE

- Diagram for YARN architecture.



- YARN architecture are as follows:

    - A client program submits the application which includes the necessary specifications to launch the application – specific Application Master itself.
    - The Resource Manager launches the Application Master by assigning some container.
    - The Application Master, on boot – up, registers with the Resource Manager. This helps the client program to query the Resource Manager directly for the details.
    - During the normal course, Application Master negotiates appropriate resource containers via the resource – request protocol.
    - On successful container allocations, the Application Master launches the container by providing the container launch specification to the Node Manager.
    - The Node Manager executes the application code and provides necessary information such as progress, status, etc., to its Application Master via an application – specific protocol.
    - During an application execution, the client that submitted the job directly communicates with the Application Master to get via an application – specific protocol.
    - Once the application has been processed completely. Application Master deregisters with the Resource Manager and shuts down, allowing its own container to be repurposed.

## YARN COMPONENTS

- YARN architecture has the following components:
    - Global Resource Manager
    - Scheduler
    - Application Manager
    - Node Manager
    - Application Master

### 1. Global Resource Manager

- Its main responsibility is to distribute resources among various applications in the system.

### 2. Scheduler

- The Scheduler is just that, a pure scheduler, meaning it does not monitor or track the status of the application.

### 3. Application Manager

- Application Manager does the following:
    - Accepting job submissions
    - Negotiating resources for executing the application specific Application Master.
    - Restarting the Application Master in case of failure.

### 4. Node Manager

- This is a per – machine slave daemon.
- Node Manager Responsibility is launching the application containers for application execution.

- ❖ It monitors the resource usage such as memory, CPU, disk, network, etc.,
- ❖ It reports the usage of resources to the Global Resource Manager.

**5. Application Master**

- ❖ This is an application – specific entity.
- ❖ Its responsibility is to negotiate required resources for execution from the Resource Manager.
- ❖ It works along with the Node Manager for executing and monitoring component tasks.

**ADVANTAGES OF YARN**

- ❖ YARN advantages are the following:
    - Multi – tenancy
    - Scalability
    - Cluster utilization
    - Compatability

**DISADVANTAGES OF YARN**

- ❖ YARN disadvantages are the following:
    - Availability
    - Limited Scalability
    - Less Resource Utilization
    - Failure in cascading

## DATA SERIALIZATION

- Data Serialization is the process of converting structured data to a format, that allows sharing or storing of the data in a form that allows recovery of its original structure.

## WORKING WITH COMMON SERIALIZATION FORMATS

- There are some common serialization formats like,
    - CSV (Comma – Separated Values)
    - XML (eXtensible Markup Language)
    - JSON (Java Script Object Notation)
    - YAML (Yet Another Markup Language)
    - Message Pack
    - Protocol Buffer
    - BSON (Binary JSON)
    - Oracle

## BIG DATA SERIALIZATION FORMATS

- Big data serialization formats are same as normal data serialization.
- It discussed above.
- Like, XML, JSON, BSON, etc.,

***Stay Home, Stay Reading…..***