**Problem Statement**

List the Components of Hadoop 2.x and explain each component in detail.

**Solution:**

**Hadoop 2.x:**

Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. A Hadoop frame-worked application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

Hadoop 2.x has some common Hadoop API which can easily be integrated with any third party applications to work with Hadoop

It has some new Java APIs and features in HDFS and MapReduce which are known as HDFS2 and MR2 respectively

New architecture has added the architectural features like HDFS High Availability and HDFS Federation

Hadoop 2.x not using Job Tracker and Task Tracker daemons for resource management now on-wards, it is using YARN (Yet Another Resource Negotiator) for Resource Management.

Hadoop2 Architecture has mainly 2 set of daemons

**HDFS 2.x Daemons:** *Name Node, Secondary Name Node and Data Nodes*

MapReduce 2.x Daemons (YARN): Resource Manager, Node Manager

HDFS 2.x Daemons

Components Description of Hadoop 2.x:

**1. Name Node**

There is only single instance of this process runs on a cluster and that is on a master node. It is responsible for manage metadata about files distributed across the cluster. It manages information like location of file blocks across cluster and its permission. This process reads all the metadata from a file named fsimage and keeps it in memory. After this process is started, it updates metadata for newly added or removed files in RAM It periodically writes the changes in one file called edits as edit logs. This process is a heart of HDFS, if it is down HDFS is not accessible any more.

**2. Secondary Name Node**

For this also, only single instance of this process runs on a cluster. This process can run on a master node (for smaller clusters) or can run on a separate node (in larger clusters) depends on the size of the cluster One misinterpretation from name is “This is a backup Name Node” but IT IS NOT!!!!!

It manages the metadata for the Name Node. In the sense, it reads the information written in edit logs (by Name Node) and creates an updated file of current cluster metadata

Than it transfers that file back to Name Node so that fsimage file can be updated

So, whenever Name Node daemon is restarted it can always find updated information in fsimage file.

**3. Data Node**

There are many instances of this process running on various slave nodes (referred as Data nodes)

It is responsible for storing the individual file blocks on the slave nodes in Hadoop cluster

Based on the replication factor, a single block is replicated in multiple slave nodes (only if replication factor is > 1) to prevent the data loss

Whenever required, this process handles the access to a data block by communicating with Name Node

This process periodically sends heart bits to Name Node to make Name Node aware that slave process is running

**4. Resource Manager**

This daemon process runs on master node (may run on the same machine as name node for smaller clusters)

It is responsible for getting job submitted from client and schedule it on cluster, monitoring running jobs on cluster and allocating proper resources on the slave node

It communicates with Node Manager daemon process on the slave node to track the resource utilization

It uses two other processes named Application Manager and Scheduler for MapReduce task and resource management

**5. Node Manager**

This daemon process runs on slave nodes (normally on HDFS Data node machines)

It is responsible for coordinating with Resource Manager for task scheduling and tracking the resource utilization on the slave node

It also reports the resource utilization back to the Resource Manager

It uses other daemon process like Application Master and Container for MapReduce task scheduling and execution on the slave node