1.

Explain the importance of below 4 demons in job execution with minimum of 5 points

•

Name node

•

Data node

•

Resource Manager

•

Node manager

**Solution:**

Namenode: The NameNode is the centerpiece of an HDFS file system. It keeps the directory tree of all files in the file system, and tracks where across the cluster the file data is kept. It does not store the data of these files itself.

Client applications talk to the NameNode whenever they wish to locate a file, or when they want to add/copy/move/delete a file. The NameNode responds the successful requests by returning a list of relevant [DataNode](https://wiki.apache.org/hadoop/DataNode) servers where the data lives.

The NameNode is a [Single Point of Failure](https://wiki.apache.org/hadoop/Single%20Point%20of%20Failure) for the HDFS Cluster. HDFS is not currently a High Availability system. When the NameNode goes down, the file system goes offline. There is an optional [SecondaryNameNode](https://wiki.apache.org/hadoop/SecondaryNameNode) that can be hosted on a separate machine. It only creates checkpoints of the namespace by merging the edits file into the fsimage file and does not provide any real redundancy. Hadoop 0.21+ has a [BackupNameNode](https://wiki.apache.org/hadoop/BackupNameNode) that is part of a plan to have an HA name service, but it needs active contributions from the people who want it (i.e. you) to make it Highly Available.

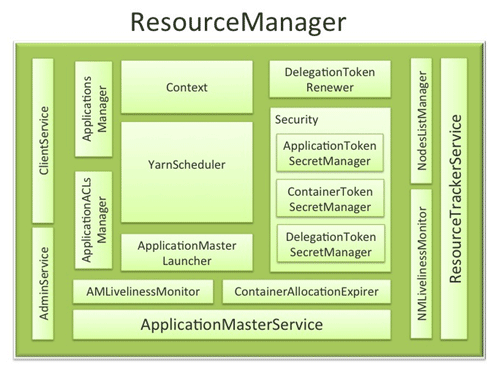
Datanode: A DataNode stores data in the [[HadoopFileSystem](https://wiki.apache.org/hadoop/HadoopFileSystem)]. A functional filesystem has more than one DataNode, with data replicated across them.

On startup, a DataNode connects to the [NameNode](https://wiki.apache.org/hadoop/NameNode); spinning until that service comes up. It then responds to requests from the [NameNode](https://wiki.apache.org/hadoop/NameNode) for filesystem operations.

Client applications can talk directly to a DataNode, once the [NameNode](https://wiki.apache.org/hadoop/NameNode) has provided the location of the data. Similarly, [MapReduce](https://wiki.apache.org/hadoop/MapReduce) operations farmed out to [TaskTracker](https://wiki.apache.org/hadoop/TaskTracker) instances near a DataNode, talk directly to the DataNode to access the files. [TaskTracker](https://wiki.apache.org/hadoop/TaskTracker) instances can, indeed should, be deployed on the same servers that host DataNode instances, so that [MapReduce](https://wiki.apache.org/hadoop/MapReduce) operations are performed close to the data.

Resource Manager: **ResourceManager (RM)** is the master that arbitrates all the available cluster resources and thus helps manage the distributed applications running on the YARN system. It works together with the per-node **NodeManagers (NMs)** and the per-application **ApplicationMasters (AMs)**.

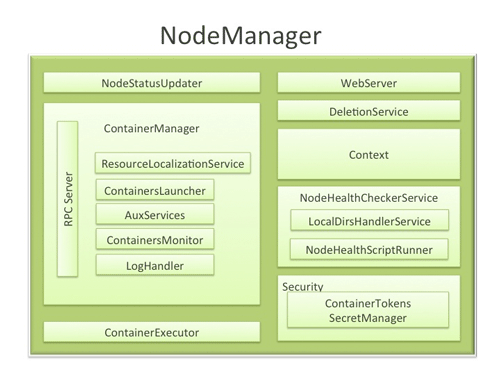
1. **NodeManagers** take instructions from the ResourceManager and manage resources available on a single node.
2. **ApplicationMasters** are responsible for negotiating resources with the ResourceManager and for working with the NodeManagers to start the containers.

[](http://hortonworks.com/wp-content/uploads/2012/08/resource_manager.png)

Node manager:

The NodeManager (NM) is YARN’s per-node agent, and takes care of the individual compute nodes in a Hadoop cluster. This includes keeping up-to date with the ResourceManager (RM), overseeing containers’ life-cycle management; monitoring resource usage (memory, CPU) of individual containers, tracking node-health, log’s management and auxiliary services which may be exploited by different YARN applications.

**NodeManager Components**

[](http://hortonworks.com/wp-content/uploads/2012/09/Node-Manager-Diagram.png)