**Explain the core changes made in Hadoop 2.x**

**Ans :**

* The progress from a Hadoop 1's more restricted processing model of batch oriented MapReduce jobs, to more interactive and specialized processing models of Hadoop 2 will only further position the Hadoop ecosystem as the dominant big data analysis platform.
* MapReduce, as implemented in Hadoop 1, can be I/O intensive, not suitable for interactive analysis, and constrained in support for graph, machine learning and on other memory intensive algorithms. Hadoop developers rewrote major components of the file system to produce Hadoop 2.
* Two of the most important advances in Hadoop 2 are the introduction of **HDFS** federation and the resource manager **YARN.**
* **HDFS**
* HDFS is the Hadoop file system and comprises two major components: namespaces and blocks storage service.
* The namespace service manages operations on files and directories, such as creating and modifying files and directories. The block storage service implements data node cluster management, block operations and replication.
* In Hadoop 1, a single Namenode managed the entire namespace for a Hadoop cluster.
* HDFS federation, multiple Namenode servers manage namespaces and this allows for horizontal scaling, performance improvements, and multiple namespaces.
* The implementation of HDFS federation allows existing Namenode configurations to run without changes. For Hadoop administrators, moving to HDFS federation requires formatting Namenodes, updating to use the latest Hadoop cluster software, and adding additional Namenodes to the cluster.
* **YARN**
* YARN, the other major advance in Hadoop 2, brings significant performance improvements for some applications, supports additional processing models, and implements a more flexible execution engine.
* YARN is a resource manager that was created by separating the processing engine and resource management capabilities of MapReduce as it was implemented in Hadoop 1.
* YARN is often called the operating system of Hadoop because it is responsible for managing and monitoring workloads, maintaining a multi-tenant environment, implementing security controls, and managing high availability features of Hadoop.
* YARN supports multiple processing models in addition to MapReduce. One of the most significant benefits of this is that we are no longer limited to working the often I/O intensive, high latency MapReduce framework.

**Explain the difference between MapReduce 1 and MapReduce 2 / Yarn**

**Ans :**

**MapReduce 1.0**

In a typical Hadoop cluster, racks are interconnected via core switches. Core switches should connect to top-of-rack switches Enterprises using Hadoop should consider using **10GbE**, bonded Ethernet and redundant top-of-rack switches to mitigate risk in the event of failure. A file is broken into **64MB** chunks by default and distributed across Data Nodes. Each chunk has a default replication factor of **3**, meaning there will be **3 copies** of the data at any given time. Hadoop is “Rack Aware” and **HDFS** has replicated chunks on nodes on different racks. JobTracker assign tasks to nodes closest to the data depending on the location of nodes and helps the NameNode determine the **‘closest’** chunk to a client during reads. The administrator supplies a script which tells Hadoop which rack the node is in, for example: **/enterprisedatacenter/rack2.**

**Limitations of MapReduce 1.0 –** Hadoop can scale up to 4,000 nodes. When it exceeds that limit, it raises unpredictable behavior such as cascading failures and serious deterioration of overall cluster. Another issue being multi-tenancy – it is impossible to run other frameworks than MapReduce 1.0 on a Hadoop cluster.

**MapReduce 2.0**

**MapReduce 2.0 has two components** – YARN that has cluster resource management capabilities and MapReduce.

In MapReduce 2.0, the JobTracker is divided into three services:

**ResourceManager,** a persistent **YARN** service that receives and runs applications on the cluster. A MapReduce job is an application. JobHistoryServer, to provide information about completed jobs Application Master, to manage each MapReduce job and is terminated when the job completes. Also, the **TaskTracker** has been replaced with the NodeManager, a **YARN** service that manages resources and deployment on a node. NodeManager is responsible for launching containers that could either be a map or reduce task.

This new architecture breaks **JobTracker** model by allowing a new ResourceManager to manage resource usage across applications, with ApplicationMasters taking the responsibility of managing the execution of jobs. This change removes a bottleneck and lets Hadoop clusters scale up to larger configurations than **4000 nodes**. This architecture also allows simultaneous execution of a variety of programming models such as graph processing, iterative processing, machine learning, and general cluster computing, including the traditional MapReduce.