Interview Questions - Introduction to Hadoop and MapReduce Programming

1. **Can you modify files in Hadoop Distributed File System (HDFS)?**

No. Files in HDFS are immutable, which means that they cannot be modified. HDFS only allows files to be appended.

2. **What is a Secondary NameNode and why is it required?**

A Secondary NameNode is created such that it acts as a NameNode in case the actual NameNode fails. The Secondary NameNode maintains a copy of the namespace image by periodically merging it with the edit log. When the primary node fails and the data files are lost, you copy the NameNode metadata files to the secondary NameNode and run it as the new primary NameNode.

3. **When should you use a combiner in MapReduce jobs?**

A Combiner is an optional class that acts as a Reducer for the output key–value pairs that are sent by a particular Mapper. A Combiner helps in minimising the data transferred between the Map and Reduce tasks. Thus, it helps in reducing the size of the intermediate data, thereby saving disk and network I/O. The Reducer class can itself be used as the Combiner if the Reduce function is commutative and associative like integer addition and multiplication. However, if the Reduce function is an average function which is not commutative and associative, then you need to write a separate class for the Combiner.

4. **Explain the shuffle and sort phases in MapReduce**

MapReduce makes sure that the input provided to every Reducer is sorted by key. Shuffle is the phase in which the system performs the sort and then transfers the Map outputs to the Reducers as input. In a MapReduce job, Shuffle and Sort happen as follows:

Map Side

- The output of the Map task is not simply rewritten to the disk; rather, it is first pre-sorted by writing it into a buffer memory.
- Before writing the Map output to the disk, the data is first partitioned corresponding to the Reducers to which they will ultimately be sent.

- Within each partition, an in-memory sort by key is performed first and then the Combiner function (if provided) is performed.
- As different Mappers can finish at different times, sometimes shuffling can start even before the Map phase is complete to save time.

Reduce Side

- The Reducer starts by copying the corresponding partitions into a buffer memory (copy phase).
- Then, these sorted partitions are merged into a single sorted file before the Reduce phase starts.
- Each Reduce task has to group/Reduce values by key. This step becomes easy if the input data is already sorted.

5. **Where is the output of a Mapper written?**

The output of a Mapper is written on the local disk. MapReduce writes its final output to the HDFS block, but the intermediate output is written on the disk, because writing an intermediate output, which is temporary, to HDFS would be inefficient.

6. **What is data locality in Hadoop?**

By providing data locality, Hadoop tries to ensure that the minimum geographical distance between the data and the compute nodes is maintained so that data access is fast. Data locality helps in achieving good performance by optimising the overhead of the network I/O.

7. **Can two clients write to an HDFS file simultaneously?**

No. When two clients try to write on a file simultaneously, the second client has to wait until the first client has completed its job. This does not apply to reading a file, that is, multiple clients can access a file simultaneously. Therefore, Hadoop is built to work on the write once read many (WORM) functionality.

8. **What happens if a DataNode fails?**

In a Hadoop cluster, the NameNode periodically receives a heartbeat and a Block report, which consists of the details of the data block stored in a particular DataNode. If the NameNode does not receive heartbeat messages from a particular DataNode within a certain period of time, then it marks the DataNode as dead.

To recover the data lost in that DataNode, it begins replicating the blocks that were stored in the dead DataNode on an alive DataNode according to the block report of the dead DataNode. This replication data transfer happens directly between the two DataNodes and never passes through the NameNode.

9. Explain the small files problem in Hadoop.

HDFS is designed for processing/storing big data. So, in case of small files, it is not prepared to efficiently process/store numerous small files. These files generate a lot of overhead to the NameNode and the DataNodes. Reading through small files normally causes a lot of seeks and hopping from one DataNode to another to retrieve each small file. All of this adds up to inefficient data read/write operations.

10. What is the difference between Data Block and Input Split?

Data Block: HDFS stores data by first splitting it into smaller chunks. HDFS splits a large file into smaller chunks known as blocks. Thus, it stores each file as a set of data blocks. These data blocks are replicated and distributed across multiple DataNodes.

Input Split: An input split represents the amount of data that is processed by an individual Mapper at a time. In MapReduce, the number of input splits is equal to that of Map tasks. Hence, it is used to configure the number of Map tasks which is equal to the number of Input Splits.

11. What is HA in a NameNode?

The implementation of Standby and Secondary NameNodes from Hadoop 2.x ensures High Availability in Hadoop clusters, which was not present in Hadoop 1.x. In the case of Hadoop 1.x clusters (one NameNode, multiple DataNodes), a NameNode was a single point of failure. If a NameNode went down owing to lack of backup, then the entire cluster would become unavailable. Hadoop 2.x solved this problem of a single point of failure by including an additional Standby/Secondary NameNode to the cluster. In Hadoop, a pair of NameNodes is in an active-standby configuration. The standby NameNode acts as a backup for the NameNode metadata. The standby NameNode also receives block reports from the DataNodes and maintains a synced copy of edit logs with the active NameNode, and in case the NameNode is down, the standby NameNode takes charge and ensures cluster availability.

12. What is Hadoop Streaming?

Hadoop Streaming is an API that allows writing Mappers and Reduces in any language.
It uses Unix standard streams as the interface between Hadoop and the user application.
Streaming is naturally suited for text processing. The data view is line-oriented and processed as a key-value pair separated by a 'tab' character. The Reduce function reads lines from the standard input, which is sorted by key, and writes its results to the standard output.

Given below is a link to the official Hadoop Documentation on Hadoop Streaming - https://hadoop.apache.org/docs/r1.2.1/streaming.html

13. What is HA in YARN ResourceManager?

The YARN Resource Manager is responsible for managing the resources in a cluster and scheduling applications.Prior to Hadoop 2.4, the Resource Manager was a single point of failure in a YARN cluster.
The Resource Manager provides High Availability (HA) by implementing an active-standby Resource Manager pair to remove this single point of failure. When the active Resource Manager Node fails, the control switches to the standby Resource Manager, and all halted applications resume from the last state saved in the state store. This allows handling failover without any performance degradation in the following situations:

- Unplanned events such as machine crashes
- Planned maintenance events of software or hardware upgrades to the machine running the ResourceManager
- ResourceManager HA requires the ZooKeeper and HDFS services to be running

Given below is a link to the official Hadoop Documentation on Resource Manager HA concept - https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/ResourceManagerHA.html