**Hadoop MapReduce Interview Questions**

Looking out for Hadoop MapReduce Interview Questions that are frequently asked by employers?

I hope you have not missed the previous blog in this interview questions blog series that contains the most frequesntly asked [***Top 50 Hadoop Interview Questions***](http://www.edureka.co/blog/interview-questions/top-50-hadoop-interview-questions-2016/?utm_source=blog&utm_medium=content-link&utm_term=hadoop-interview-questions-mapreduce) by the employers. Now, before moving ahead in this Hadoop MapReduce Interview Questions blog, let us have a brief understanding of MapReduce framework and its working:

* ***Definition:***

MapReduce is a programming framework that allows us to perform distributed and parallel processing on large data sets in a distributed environment.

* ***MapReduce 2.0 or YARN Architecture:***
  + MapReduce framework also follows Master/Slave Topology where the master node (Resource Manager) manages and tracks various MapReduce jobs being executed on the slave nodes (Node Mangers).
  + Resource Manager consists of two main components:
    - **Application Manager:** It accepts job-submissions, negotiates the container for ApplicationMaster and handles failures while executing MapReduce jobs.
    - **Scheduler:** Scheduler allocates resources that is required by various MapReduce application running on the Hadoop cluster.
* ***How MapReduce job works:***
  + As the name MapReduce suggests, reducer phase takes place after mapper phase has been completed.
  + So, the first is the map job, where a block of data is read and processed to produce key-value pairs as intermediate outputs.
  + The reducer receives the key-value pair from multiple map jobs.
  + Then, the reducer aggregates those intermediate data tuples (intermediate key-value pair) into a smaller set of tuples or key-value pairs which is the final output.

*“A mind troubled by doubt cannot focus on the course to victory.”*

*– Arthur Golden*

The above quote reflects the importance of having your fundamentals clear before appearing for an interview as well as while going through this Hadoop MapReduce Interview Question blog. Therefore, I would suggest you to go through [***MapReduce Tutorial***](http://www.edureka.co/blog/mapreduce-tutorial?utm_source=blog&utm_medium=content-link&utm_term=hadoop-interview-questions-mapreduce) blog to brush up your basics.

[Learn MapReduce from Industry Experts](http://www.edureka.co/big-data-and-hadoop?utm_source=blog&utm_medium=content-link&utm_term=hadoop-interview-questions-mapreduce)

Here, is the list of *Hadoop MapReduce Interview Questions* that will help you to stand up to the expectation of the employers.

**Hadoop Interview Questions and Answers | Edureka**

**1. What do you mean by data locality?**

[***Data locality***](http://www.edureka.co/blog/mapreduce-tutorial/?utm_source=blog&utm_medium=content-link&utm_campaign=hadoop-interview-questions-mapreduce/#data_locality) talks about moving computation unit to data rather data to the computation unit. MapReduce framework achieves data locality by processing data locally i.e. processing of the data happens in the very node by Node Manager where data blocks are present.

**2. Is it mandatory to set input and output type/format in MapReduce?**

No, it is not mandatory to set the input and output type/format in MapReduce. By default, the cluster takes the input and the output type as ‘text’.

**3. Can we rename the output file?**

Yes, we can rename the output file by implementing *multiple format output class*.

**4. What do you mean by shuffling and sorting in MapReduce?**

Shuffling and sorting takes place after the completion of map task where the input to the every reducer is sorted according to the keys. Basically, the process by which the system sorts the key-value output of the map tasks and transfer it to the reducer is called shuffle.

**5. Explain the process of spilling in MapReduce?**

The output of a map task is written into a circular memory buffer (RAM). The default size of buffer is set to 100 MB  which can be tuned by using mapreduce.task.io.sort.mb property. Now, spilling is a process of copying the data from memory buffer to disc when the content of the buffer reaches a certain threshold size. By default, a background thread starts spilling the contents from memory to disc after 80% of the buffer size is filled. Therefore, for a 100 MB size buffer the spilling will start after the content of the buffer reach a size of 80 MB.

**Note:** One can change this spilling threshold using mapreduce.map.sort.spill.percent which is set to 0.8 or 80% by default.

**6. What is a distributed cache in MapReduce Framework?**

Distributed Cache can be explained as, a facility provided by the MapReduce framework to cache files needed by applications. Once you have cached a file for your job, Hadoop framework will make it available on each and every data nodes where you map/reduce tasks are running. Therefore, one can access the cache file as a local file in your Mapper or Reducer job.

**7. What is a combiner and where you should use it?**

Combiner is like a mini reducer function that allow us to perform a local aggregation of map output before it is transferred to reducer phase. Basically, it is used to optimize the network bandwidth usage during a MapReduce task by cutting down the amount of data that is transferred from a mapper to the reducer.

**8. Why the output of map tasks are stored (spilled ) into local disc and not in HDFS?**

The outputs of map task are the intermediate key-value pairs which is then processed by reducer to produce the final aggregated result. Once a MapReduce job is completed, there is no need of the intermediate output produced by map tasks. Therefore, storing these intermediate output into HDFS and replicate it will create unnecessary overhead.

**9. What happens when the node running the map task fails before the map output has been sent to the reducer?**

In this case, map task will be assigned a new node and whole task will be run again to re-create the map output.

**10. Define Speculative Execution**

If a node appears to be executing a task slower than expected, the master node can redundantly execute another instance of the same task on another node. Then, the task which finishes first will be accepted whereas other task will be killed. This process is called speculative execution.

**11. What is the role of a MapReduce Partitioner?**

A partitioner divides the intermediate key-value pairs produced by map tasks into partition. The total number of partition is equal to the number of reducers where each partition is processed by the corresponding reducer. The partitioning is done using the hash function based on a single key or group of keys. The default partitioner available in Hadoop is *HashPartitioner*.

**12. How can we assure that the values regarding a particular key goes to the same reducer?**

By using a partitioner we can control that a particular key – value goes to the same reducer for processing.

**13. What is the difference between Input Split and HDFS block?**

HDFS block defines how the data is physically divided in HDFS whereas input split defines the logical boundary of the records required for processing it.

**14. What do you mean by InputFormat?**

InputFormat describes the input-specification for a MapReduce job.The MapReduce framework relies on the InputFormat of the job to:

* Validate the input-specification of the job.
* Split-up the input file(s) into logical InputSplit instances, each of which is then assigned to an individual Mapper.
* Provide the RecordReader implementation used to read records from the logical InputSplit for processing by the Mapper.

**15. What is the purpose of TextInputFormat?**

TextInputFormat is the default input format present in the MapReduce framework. In TextInputFormat, an input file is produced as keys of type LongWritable (byte offset of the beginning of the line in the file) and values of type Text (content of the line).

**16. What is the role of RecordReader in Hadoop MapReduce?**

InputSplit defines a slice of work, but does not describe how to access it. The “RecordReader” class loads the data from its source and converts it into (key, value) pairs suitable for reading by the “Mapper” task. The “RecordReader” instance is defined by the “Input Format”.

**17. What are the various configuration parameters required to run a MapReduce job?**

The main [***configuration parameters***](http://www.edureka.co/blog/mapreduce-tutorial/?utm_source=blog&utm_medium=content-link&utm_campaign=hadoop-interview-questions-mapreduce/#explanation_of_mapreduce_program) which users need to specify in “MapReduce” framework are:

* Job’s input locations in the distributed file system
* Job’s output location in the distributed file system
* Input format of data
* Output format of data
* Class containing the map function
* Class containing the reduce function
* JAR file containing the mapper, reducer and driver classes

**18. When should you use SequenceFileInputFormat?**

SequenceFileInputFormat is an input format for reading within sequence files. It is a specific compressed binary file format which is optimized for passing the data between the outputs of one “MapReduce” job to the input of some other “MapReduce” job.

Sequence files can be generated as the output of other MapReduce tasks and are an efficient intermediate representation for data that is passing from one MapReduce job to another.

**19. What is an identity Mapper and Identity Reducer?**

Identity mapper is the default mapper provided by the Hadoop framework. It runs when no mapper class has been defined in the MapReduce program where it simply passes the input key – value pair for the reducer phase.

Like Identity Mapper, Identity Reducer is also the default reducer class provided by the Hadoop, which is automatically executed if no reducer class has been defined. It also performs no computation or process, rather it just simply write the input key – value pair into the specified output directory.

**20. What is a map side join?**

Map side join is a process where two data sets are joined by the mapper.

**21. What are the advantages of using map side join in MapReduce?**

The advantages of using map side join in MapReduce are as follows:

* Map-side join helps in minimizing the cost that is incurred for sorting and merging in the shuffle and reduce stages.
* Map-side join also helps in improving the performance of the task by decreasing the time to finish the task.

**22. What is reduce side join in MapReduce?**

As the name suggests, in the reduce side join, the reducer is responsible for performing the join operation. It is comparatively simple and easier to implement than the map side join as the sorting and shuffling phase sends the values having identical keys to the same reducer and therefore, by default, the data is organized for us.

♣**Tip**: I would suggest you to go through a dedicated blog on [***reduce side join***](http://www.edureka.co/blog/mapreduce-example-reduce-side-join/?utm_source=blog&utm_medium=content-link&utm_term=hadoop-interview-questions-mapreduce) in MapReduce where the whole process of reduce side join is explained in detail with an example.

**23. What do you know about NLineInputFormat?**

NLineInputFormat splits ‘n’ lines of input as one split.

**24. Is it legal to set the number of reducer task to zero? Where the output will be stored in this case?**

Yes, It is legal to set the number of reduce-tasks to *zero* if there is no need for a reducer. In this case the outputs of the map task is directly stored into the HDFS which is specified in the setOutputPath(Path).

**25. Is it necessary to write a MapReduce job in Java?**

No, MapReduce framework supports multiple languages like Python, Ruby etc.

**26. How do you stop a running job gracefully?**

One can gracefully stop a MapReduce job by using the command: *hadoop job -kill JOBID*

**27. How will you submit extra files or data ( like jars, static files, etc. ) for a MapReduce job during runtime?**

The distributed cache is used to distribute large read-only files that are needed by map/reduce jobs to the cluster. The framework will copy the necessary files from a URL on to the slave node before any tasks for the job are executed on that node. The files are only copied once per job and so should not be modified by the application.

**28. How does inputsplit in MapReduce determines the record boundaries correctly?**

RecordReader is responsible for providing the information regarding record boundaries in an input split.

**29. How do reducers communicate with each other?**

This is a tricky question. The “MapReduce” programming model does not allow “reducers” to communicate with each other. “Reducers” run in isolation.